İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

STOKASTİK TALEP ALTINDA KALICI İNDİRİM ENİYİLEMESİ

DOKTORA TEZİ

Özlem COŞGUN

Anabilim Dalı : Endüstri Mühendisliği

Programı : Endüstri Mühendisliği

MART 2011

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

STOKASTİK TALEP ALTINDA KALICI İNDİRİM ENİYİLEMESİ

DOKTORA TEZİ
Özlem COŞGUN
(507062108)

Tezin Enstitüye Verildiği Tarih : 28 Aralık 2010
Tezin Savunulduğu Tarih : 18 Mart 2011

Tez Danışmanı : Prof. Dr. Cengiz KAHRAMAN (İTÜ)
Eş Danışman : Yrd. Doç. Dr. Ufuk KULA (SÜ)
Diğer Jüri Üyeleri : Prof. Dr. M.Nahit SERARSLAN (İTÜ)
Prof. Dr. Ziya ULUKAN (GSÜ)
Doç. Dr. Tufan Vehbi KOÇ (İTÜ)
Yrd. Doç. Dr. C.Erhan BOZDAĞ(İTÜ)
Yrd. Doç. Dr. Tankut ATAN (IÜ)

MART 2011

Eşime ve kızım Duru'ya,

ÖNSÖZ

Bu tez çalışmasının tüm aşamalarında değerli bilgi ve tecrübelerine başvurduğum tez danışmanım Prof. Dr. Cengiz KAHRAMAN ve tez eş danışmanım Yrd. Doç. Dr. Ufuk KULA'ya gösterdikleri her türlü ilgi, destek ve anlayış için en içten teşekkkürlerimi sunarım. Ayrıca tez jürimde yer alan sayın Prof. Dr. M. Nahit SERARSLAN, Prof. Dr. Ziya ULUKAN ve Yrd. Doç. Dr. Cafer Erhan BOZDAĞ'a değerli katkıları için teşekkür ederim.

Bu tez çalışması 107M257 numaralı TÜBİTAK projesi kapsamında gerçekleşmiştir. TÜBİTAK kurumuna bu proje kapsamında sağladığı destek ve doktora eğitimim süresince sağladığı burs için, ayrıca çalışmamız için gerekli verileri sağlayan L.C.Waikiki firmasına teşekkür ederim. Bu proje kapsamında yer alan sayın Doç. Dr. Ayhan DEMİRİZ, Yrd. Doç. Dr. Tankut ATAN ve Yrd. Doç. Dr. Gürdal ERTEK'e katkıları için teşekkürlerimi sunarım.

Eğitim hayatım boyunca, maddi ve manevi desteklerini esirgemeyen sevgili anne ve babama, bu tez çalışmasının başarıya ulaşmasında gösterdiği her türlü destek ve anlayış için sevgili eşim Arş. Gör. Cumhur COŞGUN'a ve bir nebze de olsa ilgimi esirgediğim kızım Duru'ya sonsuz teşekkürlerimi sunarım.

Aralık 2010

Özlem COŞGUN

Endüstri Y. Mühendisi

İÇİNDEKİLER

Sayfa

KISALTMALAR

ÖÖ	: Ödüllü Öğrenme
YDP	: Yaklaşık Dinamik Programlama
MNL	: Multinomial Logit
DP	: Dinamik Programlama
MDP	: Markov Karar Süreçleri
TD	: Zamansal Fark
MC	: Monte Carlo
GPI	: Genelleştirilmiş Politika Yineleme
YDY	: Yaklaşık Değer Yineleme
iia	: Farklı Alternatiflerin Bağımsızlık Özelliği
MSE	: Hata Karelerinin Ortalaması
VM	: Veri Madenciliği
std	: Standart
dk	: Dakika
iter	: iterasyon

ÇİZELGE LİSTESİ

Sayfa

ŞEKİL LİSTESİ

Sayfa

SEMBOL LİSTESİ

u_j : j ürününün fayda fonksiyonu

v_j : j ürününün temsili fayda fonksiyonu

ε : rassal hata fonksiyonu

P_j : j ürününün logit (seçilme) olasılığı

x_j : j ürününe ait gözlenebilen değişkenler vektörü

O_j : j ürününün kendi fiyat esnekliği

E_j : j ürününün çapraz fiyat esnekliği

T_j : j ürününün zaman esnekliği

D_{jt} : t anında j ürününe ait ilk talep miktarı

d_{jt} : t anında j ürününün esneklik etkilerinden sonraki talep miktarı

β_{jt} : t anında j ürününe ait fayda fonksiyonunun parametre katsayıları

s_j : j ürününün envanter düzeyi

S_t : t anındaki sistem durumu

a_{jt} : t anında j ürününün fiyat kararı

a_{jt}^* : t anında j ürününün en iyi fiyat kararı

$A(S_t)$: S_t'nin olası karar kümesi

$r_t(S_t, a_t)$: a_t kararı alındığında S_t' nin gelir fonksiyonu

$p_t(S_{t+1} \mid S_t, a_t)$: a_t kararı alındığında S_t durumundan S_{t+1} durumuna geçiş olasılığı

π^* : optimal fiyat politikası

$V(S_t)$: S_t durumunun değer fonksiyonu

$\bar{V}(S_t)$: S_t durumunun yaklaşık değer fonksiyonu

$\hat{v}(S_t)$: S_t durumunın tahmini değer fonksiyonu

$Q(S_t, a_t)$: durum-karar değer fonksiyonu

α : adım büyüklüğü parametresi

γ : indirim oranı

$C(S_t, a_t)$: a_t kararı alındığında S_t' nin maliyet (veya gelir) fonksiyonu

$W_t(w^n)$: n. iterasyonda t anında oluşan rassal talep bilgisi (örnek patika)

(g) : kümeleme indeksi

ϕ : bağımsız değişken (fonksiyon parametreleri)

θ : ϕ bağımsız değişkenlerinin katsayıları

h : birim elde tutma maliyeti

μ_j : j ürününe ait beklenen talep ortalaması

σ : standart sapma

U : güven aralığı üst sınırı

L : güven aralığı alt sınırı

STOKASTİK TALEP ATINDA KALICI İNDİRİM ENİYİLEMESİ

ÖZET

Sezon ürünleri kısa satış periyotlarına, fiyata bağlı belirsiz talep fonksiyonlarına ve dönem sonlarında da çok az değere sahiptirler. Perakendeciliğin bir alt dalı olan hazır giyimde de satış mevsimleri kısaldığından bu durum firmaların dönem sonunda ellerinde mümkün olduğunca az ürün stoğu kalmasını sağlayacak tedbirler almaya zorlamaktadır. Dönem sonu stoklarını enazlayarak geliri enbüyüklemeye yönelik önemli araçlardan biri kalıcı indirim (markdown) eniyilemesidir. Kalıcı indirimlerin optimal zamanları ve büyüklüğü hakkında kararlar vermek, talepteki belirsizlikler ve envanter miktarı ile ilişkili maliyetlerden dolayı oldukça zor bir problemdir. Tezimizde bir hazır giyim perakende zincirinde karşılaşılan kalıcı indirim eniyileme problemi ele alınmıştır.

Literatürde yer alan kalıcı indirim eniyilemesi ile ilgili çalışmalar ve pazarda var olan ticari yazılımlar, kalıcı indirim eniyilemesinde ürünlerin taleplerinin birbirinden bağımsız olduğunu varsaymakta ve ürün talepleri arasındaki fiyata bağlı tamamlayıcı ve ikame etkilerini göz ardı etmektedir. Oysa ürünler arası korelasyon ve etkileşimler de kalıcı indirim eniyilemesinde dikkate alınması gereken önemli bir noktadır.

250 mağazaya sahip L.C. Waikiki' de uygulamasını yaptığımız tezimizin temel odak noktası çok ürünlü ortam için kalıcı indirim fiyatlandırması olup özgün yönlerinden biri ürünler arasındaki çapraz fiyat esnekliği ilişkilerinin kalıcı indirim eniyilemesinde dikkate alınmasıdır. Her ürün rassal talebe sahiptir ve ürünler arasındaki ikame etkisinden dolayı bir ürünün kalıcı indirim politikası diğerlerini etkilemektedir. Bu yüzden çapraz esnekliğe sahip ürün gruplarının optimal indirim politikalarına karar verirken hepsi birlikte düşünülmelidir.

Bu çalışmada öncelikle elde edilen satış verilerinden faydalanılarak, ekonomi ve pazarlama literatüründe sıklıkla kullanılan kesikli seçim modellerinden MNL Modeli geliştirilmiş ve ürün satışlarının etkileri fiyat indiriminden dolayı ikame etkisi ve zaman etkisi olmak üzere 2 gruba ayrıştırılmıştır. Daha sonra bu indirim eniyileme problemi bir Markov Karar Süreci olarak formüle edilmiştir. İncelenen ürün gruplarında birden fazla ürün olduğundan ve optimal politikalarına birlikte karar verilmesi gerektiğinden durum uzayı boyutu büyümektedir. Bu tarz problemleri değer yineleme ve politika yineleme gibi klasik yöntemlerle çözmek imkansız olduğundan Ödüllü Öğrenme (Reinforcement Learning) algoritmalarından SARSA ve Yaklaşık Değer Yineleme algoritmaları kullanılmıştır. Tüm durumların öğrenilmesi imkansız olduğundan kümeleme (aggregation) tekniği kullanılmıştır. Birbirine yakın durumlar artık aynı kümeye dahil olduğundan her birinin öğrenilmesine gerek kalmamış ve hesaplama süresinden büyük kazanç sağlanmıştır. Diğer bir çözüm olarak da fonksiyon uydurma (function fitting) tekniği kullanılmıştır. Her periyot için tahmin edilen fonksiyonlarla, incelenen durumların, dahil olduğu küme değerleri değil de, durumların kendi değerlerine doğrudan daha hızlı ulaşılmıştır. Sonuçta her ürünün her durumuna ait, incelenen her dönem için

yaklaşık bir kalıcı indirim politikası oluşmuş, her ürünün fiyatının oluşan kalıcı indirim politikası üzerinde ne gibi etkileri olduğuna ilişkin görüşler her iki algoritma için de elde edilmiştir. Son olarak algoritmaların performansını ölçmek için determistik model geliştirilmiş ve kıyaslamaları yapılmıştır.

MARKDOWN OPTIMIZATION UNDER STOCHASTIC DEMAND

SUMMARY

Fashion (or seasonal-style) goods have short selling seasons, uncertain and nonstationary demand (price response) functions and little salvage value at the end of their selling seasons. The ever increasingly shortening selling seasons for fashion (apparel) products pressure the firms to eliminate or minimize distressed inventories to maximize revenues. One of the frequently used mechanisms to achieve this goal is markdowns. Markdowns are permanent price reductions used to clear inventory before products become obsolete. Determining the optimal timing and magnitude of the permanent markdowns is complicated by uncertain and nonstationary demand functions and the costs associated with selling out the inventory too quickly or too late.

To the best of our knowledge, prior work on markdown optimization has focused only on single-product markdowns assuming that each product has an independent demand process. However, we believe that it is crucially important to consider the correlations and the interactions among products in markdown optimization.

Thus, the main focus of our thesis is to develop methodologies for multi-product markdowns and one of the original ideas is considering the effects of the cross price elasticities of the items in the markdown optimization. The developed model was applied on a leading textile firm, L.C. Waikiki with 250 stores. The retailer faces a random demand for each product, and because of substitution among products the markdown policy of one product affects the sales of other products. Therefore, markdown policies for product groups having a significant crossprice elasticity among each other should be jointly determined.

In this study, we first use point of sales data to decompose observed product sales into two components by using MNL model which is one of the discrete choice models and commonly used in economics and marketing literature. Then, we formulate the markdown optimization problem as an MDP. Since the state space of the problem makes it impossible to solve it by classical methods such as value and policy iteration, we use SARSA and Approximate Value Iteration Method which are the Reinforcement Learning Algorithms, as viable ways to solve MDPs for stochastic inventory management. Since the state space is multidimensional and most states encountered will never have been experienced exactly before, we construct an aggregation structure to get a markdown policy for the products. Since the least-distance states are in the same clusters, it is not required to learn each state individually and therefore we can save more runtime. Then function fitting which is another solution methodology is developed and insights on the behavior of how each product price affects the resulting markdown policy for each algorithm are provided. To measure the performance of the algorithms we develop a deterministic model and compare with the developed methods.

1. GİRİŞ

Tekstil sektörü ülkemiz ekonomisinde önemli bir yer tutmaktadır. Türk tekstil sektörünün gelecekteki rekabet gücünü belirleyecek önemli faktörlerden biri sektörde bilgi teknolojisi temelli yönetim ve karar destek sistemleri kullanımının yaygınlaşması olacaktır. Tekstil sektörünün önemli bir dalı olan hazır giyim sektöründeki eğilim üretici firmaların aynı zamanda perakendeci olarak da faaliyet göstermeleridir (L.C.Waikiki, Mavi Jeans, v.b.). Dolayısıyla ülkemiz hazır giyim sanayinin rekabet edebilirliği gelecekte hazır giyim perakendecilerimizin yabancı rakiplerine karşı gösterdiği performansa da bağlı olacaktır.

Yerli hazır giyim perakendecilerimizin yabancı rakipleri ile rekabet edebilmek ve onların önüne geçebilmek için bilgi teknolojilerinden yararlanarak elde ettikleri verileri karar vermede en iyi biçimde kullanabilmeleri gerekmektedir. Geçtiğimiz on yıl içerisinde, özellikle A.B.D' de perakende sektöründeki karar vericilere yardımcı olmak amacı ile geliştirilen ve perakende analitiği yazılımları olarak adlandırılan yazılımlar yaygın olarak kullanılmaya başlanmıştır. Bu çok fonksiyonlu yazılımlar kestirim, ürün gamı planlanması, fiyatlandırma, raporlama ve benzeri işlevleri yerine getirmektedirler. Perakende analitiği yazılımlarının belirgin iki özelliği, büyük hacimlerde günlük, haftalık ve aylık veriler üzerinde işlemler gerçekleştirmeleri ve eyleme dönüştürülebilecek kararlar üretebilmeleridir. Perakende analitiği yazılımları temel olarak yönetim biliminin iki alanına dönük çalışırlar: Tedarik zinciri planlaması ve gelir yönetimi. Gelir yönetiminde amaç, firmanın gelirlerini en çoklayacak şekilde ürünlerinin zaman içinde fiyatları veya kapasiteyi nasıl kontrol etmesi gerektiği problemine çözüm üretmektir. Perakende sektöründeki gelir yönetimi uygulamalarında talebi yönlendirmek için kullanılan temel araç fiyat kontrolüdür. Perakendeciler ürünlerine indirim yaparken farklı yöntemler kullanabilirler. Bunlar genel olarak promosyonlar ve kalıcı indirimler (markdown) olarak sınıflandırılabilir. Promosyonlar belirli dönemlerde yapılan geçici indirimlerdir. Anneler Gününde bazı ürünlere yapılan indirimler, indirim kuponları, buzdolabının fiyatının kışın düşürülmesi gibi örnekler promosyonlar için verilebilir.

Kalıcı indirimler ise ürünler demode olmadan yada sezonu geçmeden envanteri temizlemek için yapılan indirimlerdir. Kalıcı indirim planlamada amaç fiyatların kademeli olarak nasıl düşürüleceğine karar vererek sezon sonu yaklaştıkça düşüş gösteren ürün talebini artırmak ve ürünlerin elde kalmadan en yüksek karla satışını gerçekleştirmektir. Bir ürüne kalıcı indirim uygulandığında bilinir ki bir daha fiyatı yükselmeyecektir. Kalıcı indirimlerde önemli olan karı maksimize ederken kalıcı indirimlerin ne zaman ve ne büyüklükte yapılacağıdır. Giderek önem kazanan kalıcı indirim uygulamasının görüldüğü ürünlerin temel özelliği; (1) uzun imalat ön süreleri nedeniyle aynı sezon içinde siparişin tekrar edilememesi, (2) belirli tarihten sonra ürün değerinin büyük oranda azalmasıdır. Bu nedenle kalıcı indirimler sezon ürünlerine, modası hemen geçen ürünlere, çabuk bozulabilecek ürünlere, tiyatro biletlerine, otomobillere, vb. uygulanmaktadır.

Sezon ürünleri kısa satış periyotlarına, belirsiz talep fonksiyonlarına ve dönem sonunda çok az hurda değerine sahiptirler. Satıcılar gözlemlenen talebi karşılamak için bu gibi ürünlerin stok siparişlerini önceden verirler. Fakat çoğu durumda uzun teslim temin süreleri ve diğer tedarik kısıtları, yeniden sipariş ve hızlı envanter yenileme olanaklarını ortadan kaldırmaktadır. Hızlı tedarik etme eksikliğinden dolayı satıcılar sezon sonu ellerinde kalan ürünlerle talebi dengeleyebilmek için iyimser talep tahminlerine göre ürün stoğu oluştururlar. Bazen, tahminler kısa süreli tahminleri yansıtabileceğinden satıcıların ellerinde stok kalabilir. Bu yüzden karlarını maksimize etmek için stoklarını temizlemek isterler, bunun için de kalıcı indirimleri kullanırlar. (Mantrala ve diğ., 2001).

Tezimizde hazır giyim perakende sektörünün bilinen firmalarından L.C.Waikiki'nin karşılaştığı, sezon sonu ürünleri için kalıcı indirim eniyilemesi problemini ele aldık. Kalıcı indirimler hazır giyim perakende sektöründe ilk olarak, 1950'li yıllarda A.B.D'de görülmeye başlanmış ancak yaygın olarak kullanılmamıştır. 1960'lı yıllardan itibaren ise kalıcı indirimlerin sıklığı ve indirim miktarları artış göstermeye başlamıştır. Hazır giyim ürünleri satan mağazalarda 1967 yılında %6 olan ortalama indirim oranı, 1997 yılında %28'e yükselmiştir (Phillips, 2005). Önümüzdeki yıllarda bu artış eğiliminin devam etmesi beklenmektedir. Bu artış eğilimin nedenleri arasında; (1) müşterilerin farklı yerlerde bulunan mağazalara erişimlerinin kolaylaşması, (2) sürekli indirimli satış yapan mağaza (outlet) sayısının artması, (3) tüketicilerin giderek daha fazla çeşit istemeleri sayılabilir. A.B.D.'de yapılan bir

çalışmaya göre, bu eğilim sonucunda 2003'te perakendecilerin yüzde 12'si kalıcı indirim eniyilemesi kullanmakta, yüzde 53'ü ise 2 sene içinde benzer sistemlere geçiş yapmayı planladıklarını ifade etmektedir (Reda, 2003).

Literatürü incelediğimizde kalıcı indirim eniyilemesi ile ilgili çalışmalarda sadece tek ürün ve talebin bağımsız olduğu durumlar ele alınmıştır. Ancak kalıcı indirim eniyilemesinde ürünler arası korelasyon ve etkileşimler dikkate alınması gereken önemli noktalardan biridir. Örneğin, ikame mallar (biri diğerinden daha pahalı iki tişört gibi) için, envanter seviyesi düşük olan bir malın ikamesi var ise bu ürünün fiyatını düşürmek gerekmeyebilir. Yani, pahalı tişörtün envanteri tükenmek üzereyken daha ucuz olan tişörtte kalıcı indirime gitmeye gerek olmayabilir. Bu örnekteki etkileşimin kalıcı indirim politikalarını nasıl etkileyeceği açık değildir.

Belirsizlik altında çok aşamalı karar problemlerine endüstride çok sık rastlanılmaktadır. Bu tarz problemlerin çözümünde Markov Karar Süreçleri (MDP) kullanılır. MDP problemlerini çözmek için stokastik dinamik programlama kullanılmaktadır; ancak gerçek uygulamalarda hesaplama gereksinimleri, bu yöntemlerin kullanışlılığını kısıtlamaktadır (Lee ve diğ., 2006). Dinamik programlama sıralı eniyileme problemlerini çözen genel bir hesaplama tekniğidir. Fakat problem boyutu (durum, karar uzayı, vb.) büyüdükçe çözüm için ihtiyaç duyulan bilgisayar kaynağı artmaktadır. Son yıllarda büyük ölçekli problemleri çözmek için Ödüllü Öğrenme (ÖÖ) gibi simülasyon ve fonksiyon yaklaşımlarını birlikte kullanan bazı yaklaşım algoritmaları geliştirilmiştir (Chapados ve diğ., 2007). Yaklaşık Dinamik Programlama (YDP) olarak da bilinen ÖÖ geleneksel dinamik programlamada karşımıza çıkan boyut problemini ortadan kaldırmaktadır (Lee ve diğ., 2006).

Tezimizde birbirinin ikamesi ve tamamlayıcısı olan birden fazla ürünün olduğu ürün grupları incelenmiştir. Bu ürünler arasında çapraz fiyat esnekliği bulunmaktadır. Yani, bir ürünün fiyatındaki değişim diğer ürünlerin talebini etkilemektedir. Bu nedenle ürünlerin optimal politikalarına karar verirken, bu ürünlerin birlikte değerlendirilmeleri gerekmektedir. Ancak sistem durumu çok boyutlu hale geldiğinden ve dolayısıyla durum sayısı arttığından bu problemi klasik dinamik programlama yöntemleri ile çözmek imkansızdır. ÖÖ algoritmaları ile bu problem ortadan kaldırılmış, ürünlerin yaklaşık optimal kalıcı indirim politikalarına karar

verilmiştir. Ayrıca ürünler arası etkileşimlerin (ikame etkisi) ve içinde bulunulan dönemin (zaman etkisi) kalıcı indirimler üzerindeki etkileri incelenmiştir. Tüm durumların öğrenilmesi çok fazla iterasyon gerektirdiğinden, kümeleme tekniği kullanılmıştır. Birbirine yakın durumlar artık aynı kümeye dahil olduğundan her durumun değerinin öğrenilmesine gerek kalmamış, karşılaşılan durumların değerleri, içerisinde bulunan küme değerine eşit olarak alınmıştır. Diğer bir çözüm yöntemi olarak fonksiyon uydurma tekniği kullanılmıştır. Her satış periyodu için tahmin edilen fonksiyonlarla, incelenen durumların, kümeleme tekniğinde olduğu gibi dahil olduğu küme değerleri değil de, kendi değerlerine direkt olarak daha hızlı ulaşılmış ve koşum süresinden büyük kazanç sağlanmıştır.

Tüketiciler ihtiyaçları olan bir ürünü alacakları zaman birbirine ikame olabilecek, renk, kalite, model gibi özellikleri kıyaslayarak aynı ürün grubunda yer alan ürünler arasından seçim yapmak durumundadırlar. Satın alma aşamasındaki en önemli etkenlerden biri ürünün fiyatıdır. Renk, kalite gibi özelliklerden dolayı bir ürünü tercih eden müşteri, indirime girmiş diğer ikame bir ürünü gördüğünde, bu ürünü tercih edebilir.

Tüketiciler kısıtlı bütçelerinin olması ve çok çeşit olmasından dolayı satın alma sırasında kararsız kalabilirler. Karar vermek için bütçe kısıtlarıyla tercih kısıtları birlikte düşünülmelidir. Eğer müşteri iki ürün arasında kararsız ise iki üründen aynı memnuniyeti yani aynı faydayı sağlıyor demektir. Dolayısıyla tüketici davranışları talep tahmininde dikkate alınması gereken önemli bir konudur (Shen ve diğ., 2007). Tüketici tercihleri hem olasılıklı ikame davranışlarını hem de tamamlayıcı ürün etkilerini içermektedir. Biz bu çalışmada tüketici davranışlarının fiyattan kaynaklandığını yani fiyatın ikame ürünleri etkilediği bir tüketici davranış modeli geliştirdik. Tüketici tercihlerini modellemede sıklıkla kullanılan kesikli seçim modellerinden Multinomial Logit Modelini kullandık.

Bu bölümde tüketici tercih modelleri, dinamik fiyatlandırma ve ÖÖ ile ilgili literatür özeti verildikten sonra, ikinci bölümde Tüketici Tercih Modellerinden Logit Modeli anlatılmıştır. Üçüncü bölümde Ödüllü Öğrenme Algoritmalarına yer verilmiş, neden ÖÖ algoritmalarının tercih edildiği, Dinamik Programlamadan farkları, karşımıza çıkan sorunları nasıl çözebileceğimize dair detaylı bilgiler verilmiştir. Bu bölümü Kalıcı İndirim Eniyileme Modelleri izlemektedir. İlk olarak, stokastik talep altında

tek ürün kalıcı indirim eniyileme modeli üzerinde durulmuştur. Durum sayısı çok fazla olmadığından ürünün optimal politikasına karar verirken klasik stokastik dinamik programlama kullanılmış ve sonuçları değerlendirilmiştir. Beşinci bölümde ise stokastik talep altında çoklu ürün grupları için kalıcı indirim eniyileme problemine değinilmiştir. Bu bölümde de ele alınan problemlerin durum uzayı çok büyük olduğundan, ÖÖ algoritmalarından SARSA ve Yaklaşık Değer Yineleme (YDY) algoritmaları ile yaklaşık optimal politikalar elde edilmiş, ikame ve zamanın optimal politikalar üzerindeki etkilerine detaylı bir şekilde değinilmiştir. Tezimiz sonuçlar bölümü ile sonlanmıştır. İkame ve zaman etkilerinin yaklaşık optimal politikaları nasıl etkilediği değerlendirilmiştir.

1.1 Literatür Taraması

Doğru ürünü, doğru zamanda, doğru yere, istenilen miktarda ve uygun fiyatla sağlamak çok zor bir iştir (Gupta ve diğ., 2006). Amaç, müşteri memnuniyetini sağlayarak kar elde etmektir. Müşteri talebini karşılarken perakendecinin karşılaştığı bazı zorluklar vardır. Bunlardan biri kalıcı indirimlerin optimal zamanları ve büyüklüğü hakkında kararlar vermektir (Vinod, 2004). Bu karar, talepteki belirsizlikler ve envanter miktarı ile ilişkili maliyetlerden dolayı oldukça zor bir problemdir.

Kalıcı indirim eniyilemesi ile ilgili literatür çalışması üç kısımda incelenmiştir. Çalışmada daha çok perakende sektörü ile doğrudan ilgili çalışmalara yer verilmiştir. İlk kısımda kalıcı indirim ve dinamik fiyatlandırma ile ilgili çalışmalar incelenmiş olup daha sonra pazarlama faktörlerinin (fiyat, indirim, promosyon, vb.) ve etkilerinin ayrıştırılması üzerinde durulmuştur. Ayrıca tüketici tercih modelleri ile ilgili literatüre değinilmiştir. Son olarak "boyutun laneti" olarak bilinen durum uzayı büyüdüğünde karşımıza çıkan problemlere karşı önerilen çözüm algoritmalarından Ödüllü Öğrenme ile yapılmış çalışmalar ele alınmıştır.

1.1.1 Dinamik fiyatlandırma ve kalıcı indirimler

Dinamik fiyatlandırma konusunda literatürde karşılaşılan ilk çalışmalar pazarlama alanındadır. Bu çalışmalarda hangi dinamik fiyatlandırma stratejilerinin hangi durumlarda kullanılabileceği hakkında çıkarımlarda bulunmak amaçlanmış, ancak

pratikte uygulanabilecek operasyonel dinamik fiyatlandırma politikaları ele alınmamıştır. Pazarlama alanında yapılan ilk dinamik fiyatlandırma çalışmalarına Monroe ve Bitta (1978), Rao (1984)'nun çalışmalarında rastlanmaktadır. Tezimizin konusu perakende sektörü için bir kalıcı indirim politikası geliştirmek olduğundan literatür çalışmamızda sadece kalıcı indirimlerle doğrudan ilgisi olan dinamik fiyatlandırma literatürünü ele aldık. Elmaghraby ve Keskinocak (2003), Bitran ve Caldentey (2003), ve Chan ve diğ. (2004) kalıcı indirimlerin gösterdiği özelliklerden farklı özellikler taşıyan dinamik fiyatlandırma çalışmaları ile ilgili literatürü geniş bir biçimde incelemektedirler.

Perakende sektöründe dinamik fiyatlandırma problemini ilk ele alan çalışmalardan biri Lazear'dır (1986). Bu çalışmada rezervasyon fiyatı (bir müşterinin ürün için ödemeyi kabul edeceği en yüksek fiyat) belirli bir olasılık dağılım fonksiyonuna uyan N adet müşteri (N biliniyor) ilk ve ikinci dönemlerde mağazaya gelmekte ve P olasılığı ile ürüne perakendecinin rezervasyon fiyatından daha düşük değer biçmektedirler. Lazear (1986), rezervasyon fiyatı değişkenliğinin fiyat üzerindeki etkisini göstermektedir. Pashigian (1988) ise, Lazear (1986)'da ele alınan modeli pazar dengesini belirleyecek biçimde uyarlayarak kalıcı indirimlerin nedenlerinden birinin de artan ürün çeşitliliği olduğu yargısına varan analitik ve gözlemsel sonuçlar sunmaktadır. Bu çalışmaların amacı, pazarda gözlemlenen fiyatlandırma stratejilerinin daha iyi anlaşılması olduğundan perakendecilere ürünleri fiyatlandırmalarında kullanabilecekleri bir karar aracı olabilmekten uzaktır.

Sipariş yenilemenin olmadığı durumlarda fiyatlandırma kararlarının verilmesinde kullanılabilecek modellerin tamamı, tek bir ürün için indirim kararlarının nasıl alınması gerektiğini ele almakta; ürünler arasındaki olası korelasyonu ve ikame etkisini göz ardı etmektedir. Rajan ve diğ. (1992), ürün talebinin deterministik olduğu ve ürün değerinin zaman içerisinde azaldığı durumda en iyi envanter ve fiyatlandırma politikalarını incelemiştir. Talebin deterministik olduğu durumda kalıcı indirimleri inceleyen diğer bir çalışma ise Smith ve Achabal'dır (1998). Smith ve Achabal (1998), talebi fiyata bağlı bir satış hızı biçiminde tanımlayarak doğrusal olmayan bir matematiksel programlama modeli geliştirmiştir. Söz konusu modelde talep hızı fiyatın yanı sıra eldeki envanter düzeyine ve talepteki mevsimsel dalgalanmalara bağlı olarak değişmektedir. Smith ve Achabal, bu modeli kullanarak en iyi envanter düzeylerini ve en iyi fiyatları belirlemektedir. Gallego ve Van Ryzin

(1994), tek bir ürün için kalıcı indirim problemini, talep hızı kontrol problemi olarak ele almakta ve talebi fiyata bağlı olarak azalan homojen Poisson dağılımı, fiyatı ise zamanın sürekli bir fonksiyonu olarak modelleyerek en iyi fiyatları belirlemekte; fiyatların kesikli olduğu durum için ise sezgisel bir algoritma geliştirmektedir. Feng ve Gallego (1995) ise, Gallego ve van Ryzin (1994)'de ele alınan kalıcı indirim problemine farklı bir açıdan yaklaşarak, uzunluğu önceden bilinen bir satış sezonu içerisinde fiyatı p_1'den p_2' ye düşürmek/yükseltmek için (fiyat değiştirmeye sadece bir kez izin verilmektedir) en uygun zamanın bulunması problemini ele almaktadır. Feng ve Xiao (2000) ise Feng ve Gallego (1995)'da ele alınan problemi ikiden fazla fiyata uyarlamaktadır. Yukarıda değinilen çalışmaların tümü tek bir mağazada satılan tek bir ürünü ele almaktadır. Bitran ve diğ. (1998) ise tek bir ürünün kalıcı indirim fiyatlarının birden fazla mağaza arasında koordinasyon problemini dinamik programlama formülasyonu yardımı ile incelemektedir. Ancak, dinamik programlama formülasyonu uygulamada karşılaşılan büyük boyutlu problemlerin çözümünü zorlaştırdığından sezgisel algoritmalar geliştirmişler ve bu algoritmaları gerçek perakende sektörü verileri ile test etmişlerdir.

Talebin rassal olduğu durumda başlangıç envanter düzeyini bir karar değişkeni olarak ele alan Mantrala ve Rao (2001), stokastik dinamik programlamaya dayalı bir kalıcı indirim karar destek sistemi geliştirmiştirler. Söz konusu karar destek sistemi satış noktalarından elde edilen verileri kullanarak en iyi başlangıç envanter düzeylerini ve fiyatlarını belirlemektedir. Mantrala ve Rao (2001) ile Smith ve Achabal (1998) tarafından geliştirilen karar destek sistemleri, piyasada bulunan diğer karar destek sistemlerinde olduğu gibi ürünler arasındaki talep esnekliğini ve ikame etkisini göz önünde bulundurmamakta ve sadece bir tek ürünün en iyi başlangıç envanter düzeyini hesaplamaktadır. Bitran ve diğ. (1998) ile Heching ve diğ. (2002)' nin ortaya koyduğu çözümler ise profesyonel yazılımlara dönüşmemiş olsa da gerçek verilerle ve endüstriden katılımla denenmiş oldukları için ilginç çalışmalardır. Heching ve diğ. (2002) deterministik bir model kullanarak bir giyim firmasının bazı ürünlerinde kendi geleneksel yöntemleri yerine analitik modellemeyi tercih etmesi halinde elde edebileceği kazancı göstermektedir. Bitran ve diğ. (1998) tek mağazalı diğer çalışmaların aksine dinamik programlama yoluyla Şili'deki bir giyim zincirinin değişik mağazaları için kalıcı indirimleri koordine eden bir model üzerinde çalışmıştır.

Gelir yönetimi literatüründe birden fazla ürünün ele alındığı çalışmalar, üretimde ya da ürün tesliminde aynı kaynağı kullanan ürünleri incelemektedir. Örneğin, Gallego ve van Ryzin (1997) aynı ortak kapasiteyi kullanan ürünlerin en iyi kapasite tahsislerini kısmen karakterize etmektedirler. Gallego ve van Ryzin (1997), eldeki envanter düzeyi düştükçe en iyi ürün fiyatların düştüğünü ve eldeki envanter düzeyi sabit kalmak koşulu ile en iyi fiyatların zamana bağlı olarak azaldığını göstermektedirler. Tek bir ürün için benzer yapısal sonuçlar Gallego ve van Ryzin (1994), ve Zhao ve Zheng (2000) tarafından da elde edilmiştir. Gupta ve diğ. (2006) ise tüketicilerin rezervasyon fiyatlarının zamana bağlı olarak azaldığı durumlarda talebin deterministik ve rassal olduğu koşullar için tek ürün kalıcı indirimlerinin eniyilemesi için kullanımı oldukça kolay algoritmalar geliştirmişlerdir. Gupta ve diğ. (2006), ayrıca sayısal örnekler yardımıyla, genel olarak geliştirdikleri deterministik talep algoritmalarının rassal talep algoritmalarından daha iyi olduğunu göstermişlerdir.

Maglaras ve Meissner (2006) ise aynı ortak kapasiteyi kullanan birden fazla ürün için hem kapasite hem de fiyat kontrol problemini sıvı yaklaşımı (fluid approximation) kullanarak incelemişler, her iki problemin nasıl aynı problem formülasyonu ile ifade edilebileceğini ve bu formülasyonu kullanarak, Gallego ve van Ryzin'in (1997) birden fazla ürün kapasite kontrol probleminin ve Lee ve Hersh'in (1993) fiyat kontrol probleminin uygun içbükey gelir fonksiyonları için nasıl tek ürün fiyat kontrol problemi haline getirilebileceğini göstermişlerdir.

Bizim ele aldığımız problem bu çalışmalarda incelenen problemlerden iki yönden farklıdır: (1) Her iki çalışmada da ürünlerin ortak kapasite kullandığı varsayılmış ve *t.* dönemde *x* birim kapasite kaldığında *i.* ürün fiyatının ne olması gerektiği problemi incelenmiştir. Bizim ele aldığımız kalıcı indirim probleminde ise elde *x* adet ürün kaldığında *i.* ürün fiyatının azaltılıp azaltılmayacağı ve azaltılacak ise en uygun indirim oranının ne olması gerektiğidir. Gallego ve van Ryzin (1997) ve Maglaras ve Meissner (2006)'de incelenen problemlerde ürünler aynı kapasiteyi kullandığından *i.* ürün kapasitesinin azaldığı ve diğer kapasitelerin daha fazla olduğu durumda en iyi politika *i.* ürün fiyatını arttırmak veya sabit tutmak olacaktır. Bizim incelediğimiz problemde, elde *x* adet kalan *i.* ürünün en iyi fiyatı arttırılamayacağından en iyi fiyat politikalarının yapısı tamamen farklı olacaktır. (2) Gallego ve van Ryzin (1997) ve Maglaras ve Meissner (2006), ürünler arasındaki karşılıklı esnekliğin ve ikame

etkisinin fiyatları nasıl etkilediğini incelememektedir. Bizim tezimizin amaçlarından biri de bu soruya yanıt vererek, hazır giyim perakendecilerinin indirim kararlarında kullanabilecekleri çıkarımlarda bulunmaktır.

Bu literatür özetinden de görüldüğü gibi ikame etkisinin çoklu ürünlerin dinamik fiyatlandırması üzerindeki etkileri gelir yönetimi literatüründe araştırılmamıştır. Bitran ve Caldentey'in (2003) de belirttiği gibi ikame etkisinin tüketici davranışları üzerindeki etkisinin anlaşılmasına yönelik benzer çabaları gelir yönetimi literatürünü sadece teğet geçmektedir. Bu tezin temel hedeflerinden biri perakende sektöründe kalıcı indirim kararlarında ikamenin ne tür etkileri olduğunu araştırmaktır.

Tezimizde tüketicilerin istedikleri ürünlerin fiyatları diğerlerinden fazla olduğunda ikamelerinin alınabileceği varsayımına dayanan bir talep modeli geliştirdik. Bu ürünler renk, kalite, özellikle fiyat ve satın alma zamanları açısından farklılık gösterebilirler. Bu yüzden modelimiz fiyattan kaynaklanan ikameye dayanarak geliştirilmiştir. Literatüre bakıldığında genellikle stoksuz kalmaya ve ürün çeşitliliğine bağlı ikame modelleri çalışılmıştır.

Perakendecilikte ikame olasılıkları üç grupta sınıflandırılmaktadır: i) Tüketici günlük ihtiyaçları için sürekli bir mağazaya alışverişe gelmektedir, ve bir gün alacağı şeyin stokta kalmamasından dolayı diğer bir ürünü almaktadır. Bu stoksuz kalmaya bağlı ikame olarak adlandırılmaktadır. ii) Tüketicinin reklamlardan ya da diğer mağazalardaki önceki alışverişlerinden dolayı istediği bir ürün vardır, fakat o gün gittiği mağazada o ürün yoktur. Bu da ürün çeşitliliğine bağlı ikame olarak adlandırılmaktadır. iii) Tüketici rafta bir ürün görür ve çok beğenir. Satın almama seçeneğinden daha iyi bir seçenekse bu ürünü satın alır. Bu durumda tüketicinin tercih edebileceği farklı ürünlerde olabilirdi ama perakendeci bu ürünleri ya bulundurmuyordur ya da stokta yoktur. İlk iki durum gıda gibi tekrar almayı gerektiren durumlara uyar, sonuncusu da tekstil ürünleri için uygundur. Biz bu üç durumdan farklı olarak dördüncü bir durum üzerinde çalıştık. Tüketici istediği belli bir ürünü almak için bir mağazaya gelir, ama indirime girmiş, alacağı ürüne ikame olabilecek diğer bir ürünü, daha ucuz olduğu için, tercih edebilir. Bu da fiyattan kaynaklanan ikame olarak adlandırılır. Literatürden edindiğimiz bilgilere göre böyle bir ikame etkisi daha önce hiç çalışılmamıştır. Sadece Aydin ve Portesus (2005) bir newsvendor modeli (tek periyot ve stokastik talep) üzerinde ele alınan ürün grubu için en iyi envanter düzeyleri ve fiyatlarına karar vermişlerdir. Tüketici ikame bir

ürünü tercih ettiğinde ilk tercih ettiği üründe satış kaybı olacaktır ve ikame ürünün talebi artmış olacaktır (Smith ve Agrawal, 2000).

Bir perakende mağazasında bulunan ürün sayısı çok fazla olabilir. İkame için bu ürünleri alt gruplara bölmek verimli yollardan biridir. Bölme bu alt gruplar içindeki farklılıkların çok küçük, farklı alt gruplar arasında ise daha büyük olmasına dikkat edilerek yapılmaktadır.

Çalışmamızda ikamenin alt gruplar içinde yer aldığı farz edilmiştir. İkamenin bilinen iki modeli vardır. Birincisi ürün yelpazesindeki her ürüne ilişkin bir fayda değerinin oluşturulduğu fayda tabanlı ikame modelidir. Multinomial Logit (MNL) Modeli ekonomi ve pazarlama literatüründe ve özellikle ürün çeşidi planlama modellerinde sıklıkla kullanılan fayda tabanlı bir ikame modelidir (McFadden, 1974; Guadagni ve Little, 1983; Ben Akiva ve Lerman, 1985; Gupta, 1988; Ryzin ve Mahajan, 1999). Maddah ve Bish (2007) tüketici tercih modeli için MNL kullanmışlar ve envanter yapısı için de bir gazeteci çocuk model tipi düşünmüşlerdir. Bazı özel önemli durumlar için de en iyi ürün çeşidi yapısını geliştirmişlerdir.

İkame modelinin ikincisi olan daha çok envanter modellerinde kullanılan dış ikame modelinde (exogenous model) ise tüketiciler ürün kümesinden bir ürünü seçerler, seçtikleri ürün herhangi bir nedenden dolayı mevcut değilse verilen bir ikame olasılığı ile başka bir ürünü tercih edebilirler (Netessine, Rudi, 2003). Anupindi ve diğ. (1998) ikame olasılıklarını tahmin etmek için bir model önermişler ve bunu bir uygulama üzerinde göstermişlerdir. Rajaram ve Tang (2001) ikame etkisinin sipariş miktarı ve kar üzerindeki etkilerini analiz etmişlerdir. Analizleri gerçekleştirirken bir ürünün artakalan envanter miktarını stok dışı ürünlerin bir ikamesi olarak ele almışlardır. Sonuç olarak ikamenin kar üzerindeki etkisinin ikame olmadığı durumdakine göre daha yüksek olduğunu göstermişlerdir.

Yücel ve diğ. (2008)' nin çalışmasında ise yine ikame altında ürün yelpazesi ve envanter planlama analizleri yapılmıştır. İkamenin ihmal edildiği durum, tedarikçi seçim kararının dahil olmadığı durum ve alan kısıtının ihmal edildiği durum olmak üzere üç performans analizi gerçekleştirmişlerdir. İkame olasılıklarının bilindiği farz edilmiştir. Biliyoruz ki ikame olasılıklarının hesaplanması zor bir problemdir. Tezimizin ana bölümlerinden birini de bu olasılıkların belirlenmesi oluşturmaktadır.

1.1.2 Pazarlama faktörlerinin ayrıştırılması

Pazarlama değişkenlerinin tüketici satın alma davranışlarını ve ürün satışlarını nasıl etkilediği literatürde çok fazla ele alınan konuların başında gelmektedir. Promosyonlardaki artışla bu ilgi daha da büyümektedir. İmalatçılar satış artışlarını bir promosyon sürecinde görseler de, önemli bir soru tartışılmaya devam etmektedir: Satışlardaki bu artış tüketicilerin ürün tercihlerini değiştirmesiyle mi yoksa ürünleri stoklama isteği ile mi gerçekleşmektedir? Satın almanın artması ve marka değişimi bir promosyonun ana talep ve ikincil talebiyle ilgilidir. Gupta (1988) bu etkileri tek bir modelde incelemiş ve bir ürünün toplam esnekliğini bu etkilere ayrıştırmıştır. Promosyon etkisinin çoğunluğunun (84%) ikincil taleplerden kaynaklandığını (ürün seçimi değişimi), az bir kısmınında (16%) ana taleplerden kaynaklandığını (stoklama ve satın alma olasılığının artması) bulmuştur. Bell ve diğ. (1999) de yaklaşık olarak aynı sonuçları savunmuşlar, farklı olarak bu sonuçlarda ürün faktörleri ve tüketici özelliklerinin de etkisi olduğunu belirlemişlerdir. Stoklama satış promosyonlarının temel sonuçlarından biridir (Neslin, 2002). Çünkü promosyon tüketicileri daha sonra alacakları bir şeyi ya önceden alarak ya da daha fazla alarak stoklamaya teşvik etmektedir (Neslin ve diğ., 1985). Stoklama davranışının varlığına yönelik nesnel kanıtlara satın alma olasılığı ve miktarına yönelik panel veri analizleri ile ulaşılmıştır (Gupta, 1988; Bucklin ve Gupta, 1992; Bucklin ve diğ., 1998; Chintagunta ve Haldar, 1998). Neslin ve diğ. (2007) stoklamanın kar ve maliyetlerini detaylı olarak çalışmıştır. Fakat stoklama kahve, çay gibi bazı ürünlerle sınırlı kalmaktadır. Çabuk bozulacak gıda ürünlerinde ya da tekstil ürünlerinde bundan bahsetmek zordur. Bir tüketici indirime girmiş bir tekstil ürününü sırf indirime girmiş diye daha fazla almaz, ihtiyacı ne kadarsa o kadarını alır. Aynı kategorideki bir ürünün fiyatı düşmüşse tüketici bu ürünü tercih edebilir. Buna ikame etkisi denilmektedir. Bizimde inceleyeceğimiz etkilerden birini ikame etkisi oluşturmaktadır.

Pazarlama değişkenlerinin tüketici satın alma davranışlarını etkilemesi konusu yeni bir konu değildir. Birçok çalışma göstermiştir ki fiyat ve satış promosyonlarının tüketicilerin ürün seçme, satın alma zamanı ve ne kadar alacakları gibi kararlar üzerinde etkileri vardır. Guadagni ve Little (1983) pazarlama değişkenlerinin marka seçimi üzerindeki etkisini, Raj, Stelin ve Mitchell (1977) satın alma zaman aralıkları üzerindeki etkisini, Little ve Guadagni (1986) marka seçimi ve satın alma zaman

aralıklarını birlikte, ve Krishnamurthi ve Raj (1988) marka seçimi ve satın alma miktarını birlikte incelemişlerdir.

Satışlar tüketicilerin ne zaman, hangi ürün ve ne kadar satın alacakları gibi kararlarının bir sonucu olmasına rağmen bu üç tüketici kararının bir pazarlama elemanı üzerindeki etkilerini ayrıştırmak için üçünü birlikte inceleyen çok az çalışma bulunmaktadır. Gupta (1988) promosyon sürecinde bu üç kararın satışlar üzerindeki etkilerini şöyle ayrıştırmıştır: marka değişiminden (brand switching) dolayı satışlardaki artış, satın alma aralıklarının (interpurchase time) artmasından dolayı ve stoklamadan (stockpiling) dolayı satışlardaki artış.

Bu çalışmalardan farklı olarak biz sezon sonu ürünlerde satışlardaki değişimin etkilerini 2 gruba ayrıştırdık: ikame etkisi ve zamanın etkisi. Zaman sezon ürünlerinin satışında negatif etkiye sahiptir ve daha önce çalışılmayan bir parametredir. Şöyle ki, literatürde değerlendirilen zaman etkileri genellikle havanın sıcak, soğuk olması, tatil günleri, kutlama günleri gibi modele 0-1 ikili (binary) değer olarak alınmış ve değerlendirilmiştir. Biz ise zamanı sadece o periyodun etkisi yani birinci periyodun etkisi, ikinci periyodun etkisi gibi, ele alıp değerlendirdik. Sezon sonuna doğru talep genellikle azaldığından dolayı talep üzerindeki negatif etkisi dolayısıyla zamanın satışlar üzerindeki etkisi artacaktır. İkamenin ise talep üzerinde pozitif bir etkisi vardır. Yani bir ürünün fiyatı düştüğünde diğer ürünün talebinde bir artış olacaktır.

1.1.3 Ödüllü öğrenme (ÖÖ)

Pazarlama literatüründe pek fazla incelenmeyen diğer bir konu ise boyut problemidir. Literatürdeki modellerin çoğunda aşırı parametre kullanımından dolayı özellikle birçok ürünün olduğu bir ortamda, çapraz ikame davranışları incelendiğinde, çok fazla hafıza ve süre gerektirdiğinden dolayı problemlerin çözümü zorlaşmaktadır (Chan ve diğ., 2008). Boyut probleminden dolayı seçenek kümesinde birçok ürün olduğunda dinamik eniyileme problemlerini çözmek zordur. Bu nedenle bu problemin çözümü için ya ürün grupları oluşturma – kümeleme (aggregation) ya da diğer bazı basit metotlar kullanılmıştır. Fakat marka düzeyinde ürünleri bir araya getirme ayrı ayrı oldukları durumdaki bazı çapraz ikame davranışlarını ortadan kaldıracaktır ve bilgi eksikliğine neden olacaktır. Chan ve diğ. (2008) bu problem

için bir çözüm önermişlerdir. Kişi fayda fonksiyonlarını modellemek için 'hedonic yaklaşımı' kullanmışlar ve bazı basit yaklaşımlarla boyut problemini çözmeye çalışmışlardır. Tezimizde çoklu ürün grupları incelendiğinden dolayı durum boyutu büyümektedir. Çok büyük boyuttaki problemleri dinamik programlama ile çözmek imkansız olduğundan, Markov Karar Süreci modellerini çözmekte iyi sonuçlar veren Ödüllü Öğrenme algoritmalarını kullandık. Ayrıca eldeki envanter düzeyine dayanan kümeleme ve fonksiyon uydurma tekniklerini bu algoritmalarda kullanarak ele aldığımız problemi çözdük.

Kaelbling ve diğ. (1996) Makine öğrenmesi olarak da bilinen Ödüllü Öğrenme ile ilgili geniş bir bilgi taraması yapmış. Moriarthy ve diğ. (1999) evrimsel algoritma uygulamaları üzerinde odaklanmışlar, uygulamalarla ÖÖ algoritmasının zayıf ve güçlü yönlerini sunmuşlardır. Sutton ve Barto (1998) diğer yaklaşımlardan daha çok hedef amaçlı öğrenme odaklı olan ÖÖ algoritmasını geliştirmişlerdir. ÖÖ algoritması hayvan öğrenmesinden daha birçok farklı öğrenme probleminin olduğu durumlara uygulanabilmektedir (Whatkins, 1989).

2. TÜKETİCİ TERCİH MODELLERİ

Kesikli Seçim Modelleri (Discrete choice models) tüm alternatifler arasında karar vericilerin seçimlerini nasıl yaptıklarını açıklamaya çalışan modellerdir. Seçim kümesinin üç karakteristik özelliğe sahip olması gerekir. Birincisi karar vericiler açısından alternatiflerin birbirinden ayrık olması (yani karar verici sadece bir alternatifi seçmeli), ikincisi seçim kümesinin tüm olağan alternatifleri içermesi, son olarak da alternatif sayısının sonlu olması gerekir.

Kesikli seçim modelleri karar vericiler tarafından fayda-enbüyüklemesi davranışı yaklaşımı altında elde edilmiştir. Karar verici seçim kümesinden en fazla faydayı hangi alternatif sağlıyorsa onu seçer.

Kesikli seçim modellerinden bazıları Logit, GEV, Probit ve Karma Logit modelleridir. Diğer modellerde zaten bunların kombinasyonlarından oluşmaktadır (karma probit, vb.). Bu bölümde tezimizde kullandığımız Logit Modelinin özellikleri ve farklılıklarına yer verilmiştir (Train, 2003).

2.1 Logit Modeli

Logit Modeli sıklıkla ve kolaylıkla kullanılan bir kesikli seçim modelidir. Bu yaygınlığı seçim olasılıkları için kullanılan formülünün kapalı formda yazılabilmesi ve bundan dolayı daha kolay kullanılabilmesidir. Logit formülü farklı alternatiflerin bağımsızlık özelliği (iia) olarak bilinen, seçim olasılıklarının karakteristik özelliklerini bazı yaklaşımlarla tanımlayan Luce tarafından ilk olarak 1959 yılında ortaya konmuştur.

Logit modelinde fayda fonksiyonu $u_j = v_j + \varepsilon_j$ olarak tanımlanmaktadır. v_j fayda fonksiyonunun gözlenebilen kısmını (temsili fayda fonksiyonu), ε_j de gözlenemeyen kısmını göstermektedir. Karar verici *n*, *j* alternatifi yerine *i* alternatifini *j*'nin fayda değeri *i*'nin fayda değerinden büyük olduğu zaman seçecektir. Seçim olasılığı ise

$$
\begin{aligned}
P_{ni} &= \text{Prob}(\text{u}_{ni} > u_{nj}) \\
&= \text{Prob}(v_{ni} + \varepsilon_{ni} > v_{nj} + \varepsilon_{nj}) \\
&= \text{Pr ob}(\varepsilon_{nj} < \varepsilon_{ni} + v_{ni} - v_{nj})
\end{aligned}
\tag{2.1}
$$

şeklinde gösterilebilir.

Eğer ε_{ni} verilirse, bu ifade $\varepsilon_{ni} + v_{ni} - v_{nj}$ 'ye göre değerlendirilen her ε_{nj} için bir kümülatif dağılımdır ki o da $\exp(-\exp(-(\varepsilon_{ni} + v_{ni} - v_{nj})))$ değerine eşittir. ε'ler bağımsız olduğu için tüm $j \neq i$ için bu kümülatif dağılım tüm bireysel kümülatif dağılımların çarpımına eşittir:

$$
P_{ni} \mid \varepsilon_{ni} = \prod_{j \neq i} e^{-e^{(\varepsilon_{ni} + v_{ni} - v_{nj})}} \tag{2.2}
$$

Tabii ki ε_{ni} verilmediği için, seçim olasılılığı yukarıdaki ifadenin ε_{ni}'ye göre integrali olacaktır:

$$
P_{ni} = \int (\prod_{j \neq i} e^{-e^{(\varepsilon_{ni} + v_{ni} - v_{nj})}}) e^{-\varepsilon_{ni}} e^{-e^{-\varepsilon_{ni}}} d\varepsilon_{ni} \tag{2.3}
$$

Bu integral bazı ayarlamalardan sonra aşağıdaki gibi bir kapalı formda ifade edilir ki bu logit seçim olasılığıdır:

$$
P_{ni} = \frac{e^{v_{ni}}}{\sum_{j} e^{v_{nj}}} \tag{2.4}
$$

Fayda fonksiyonunun gözlenebilen kısmını ifade eden temsili fayda v_{nj} genellikle kullanılan parametrelerle lineer ifade edilir:

$$
v_{nj} = \beta x_{nj} \tag{2.5}
$$

Burada x_{nj} alternatif *j*' ye ilişkin gözlenebilen değişkenler vektörünü gösterir. Bu gösterimle birlikte logit olasılığı artık aşağıdaki gibi ifade edilebilir:

$$
P_{ni} = \frac{e^{\beta x_{ni}}}{\sum_{j} e^{\beta x_{nj}}} \tag{2.6}
$$

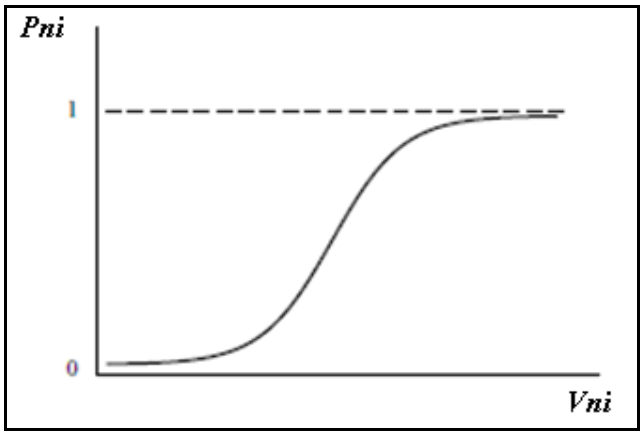
ε_{nj} değerinin tüm *j*'ler için birbirinden bağımsız olduğu (iia) varsayılır. Bu varsayımın kritik noktası alternatiflerde gözlenemeyen faktörlerin korelasyonlarının

olmaması ayrıca aynı varyansa sahip olmalarıdır. Bu yaklaşım sınırlayıcı olmasına rağmen seçim olasılıklarının oluşmasında güvenilirdir. Logit Modelinin yaygın olması bundan gelmektedir. Diğer modellerin gelişimi de logit içindeki bu bağımsızlık yaklaşımından kurtulmak üzerine kurulmuştur.

Seçim davranışlarını göstermek için logit modelinin gücünü ve sınırlarını 3 kavram açıklar. Logit Modellerinin uygulanabilirliği şu şekilde özetlenebilir:

1. Logit sistematik karar vericinin gözlenebilen özelliklerine (temsili fayda özellikleri) ilişkin beğeni farklılığı (taste variation) gösterir.
2. Logit Modeli alternatifler karşısında araştırmacının belirttiği temsili fayda verildiği takdirde orantılı ikame etkisini içerir.
3. Zamanla tekrarlayan seçim durumlarında gözlenemeyen faktörler bağımsız ise, logit bu tekrarlayan seçimin dinamiklerini yakalayabilir. Fakat zamanla bu faktörler arasında korelasyon olursa logit bu durumları yakalayamayabilir.

Bir alternatifin temsili fayda değeri diğer alternatiflerle karşılaştırılamayacak derecede düşükse, bu alternatifin fayda değerindeki küçük bir artış seçilmesine ilişkin olasılığı üzerinde küçük bir etkisi olacaktır: Diğer alternatifler zaten yeterince iyi olduklarından dolayı bu küçük gelişme onları çok etkilemeyecektir. Benzer bir şekilde, bir alternatif gözlenebilen özellikleri açısından diğerlerine göre çok iyiyse, temsili faydasındaki büyük bir iyileşme seçim olasılığını yine çok değiştirmeyecektir. Temsili faydadaki artış noktasının seçilme olasılığı üzerindeki etkisi bu olasılık 0,5 olduğunda en büyüktür. Bu durumda insanların seçimindeki küçük bir iyileşme olasılık üzerinde büyük değişimlere neden olacaktır. Logit olasılıklarının sigmoid şekli (Şekil 2.1) çoğu kesikli seçim modelleri tarafından paylaşılmaktadır ve birçok önemli uygulamaları vardır.



Şekil 2.1 : Logit dağılım grafiği (Train, 2003).

Logit Modeli alternatifler arasında temsili fayda verildiğinde orantılı ikameyi göstermektedir. Bir alternatifin özellikleri geliştirildiğinde (ör: fiyat düştüğünde) bu alternatifin seçilme olasılığı artar. Başka alternatifleri seçme eğiliminde olan bazı karar vericiler artık bu alternatifler yerine fiyatı düşen alternatifi tercih edeceklerdir.

Tüm alternatiflerin seçim olasılıkları toplamı 1 olduğundan, bir alternatifin seçilme olasılığındaki artış aynı şekilde diğerleri üzerinde azalışa neden olacaktır. Yani ikameden dolayı ürünlerin taleplerinde değişiklik olacaktır. Böylelikle ikame modelleri araştırmacı pazar payının nerden geldiği ile ilgilenmeksizin sadece bu payla alakalı olsa bile önem arz etmektedir. Demek ki bir araştırmacının temsili fayda değerleri verildiğinde, ikame bu şekilde oluyorsa, logit modeli uygun bir modeldir.

2.2 Farklı Alternatiflerin Bağımsızlığı Özelliği

İki *i* ve *k* alternatifi için, logit olasılıkları oranı aşağıdaki gibidir:

$$\frac{P_{ni}}{P_{nk}} = \frac{e^{v_{ni}} \Big/ \sum_j e^{v_{nj}}}{e^{v_{nk}} \Big/ \sum_j e^{v_{nj}}} = \frac{e^{v_{ni}}}{e^{v_{nk}}} = e^{v_{ni} - v_{nk}} \qquad \textbf{(2.7)}$$

Bu oran görüldüğü gibi *i* ve *k* alternatiflerinden başka alternatiflere bağlı değildir. Yani, *i* ve *k* alternatiflerinin göreceli seçimleri, ne hangi alternatifler olduğuna ne de o alternatiflerin hangi özelliklerinin olduğuna bağlı değildir. Diğer alternatiflerden bağımsız olduğu için de buna farklı alternatiflerin bağımsızlığı (iia) denilmektedir ve Logit Modeli bu özelliği içermektedir.

2.3 İkame Modelleri

i alternatifinin bir özelliğinin değiştiğini düşünelim. Bu değişimin diğer tüm alternatiflerin olasılıkları üzerinde ne gibi etkileri olduğunu bilmek isteriz. Bu sorulara cevap bulmak için seçim olasılıklarının türevleri hesaplanır (Train, 2003).

Bizim incelemek istediğimiz şey çoklu ürün grubunda yer alan bir ürünün fiyatını değiştirdiğimizde diğer ürünlerin seçilme olasılıkları üzerinde ya da kendi seçilme olasılığı üzerinde ne gibi etkilerinin olduğunu araştırmaktır.

i alternatifinin bir özelliğinin (Fiyat) değiştiğini düşünelim. Fiyat parametresi yerine **(2.8)** ve **(2.9)** denklemlerinde *f* kullanılmıştır. *i* alternatifini seçme olasılığındaki değişim (diğer alternatiflerin temsili faydası aynı kalmak şartıyla)

$$\begin{aligned}\frac{\partial P_{it}}{\partial f_{it}} &= \frac{\partial(e^{v_{it}} / \sum_j e^{v_{jt}})}{\partial f_{it}} \\ &= \frac{e^{v_{it}}}{\sum e^{v_{jt}}}\frac{\partial v_{it}}{\partial f_{it}} - \frac{e^{v_{it}}}{(\sum e^{v_{jt}})^2} e^{v_{it}}\frac{\partial v_{it}}{\partial f_{it}} \\ &= \frac{\partial v_{it}}{\partial f_{it}}(P_{it} - P_{it}^2) \\ &= \frac{\partial v_{it}}{\partial f_{it}} P_{it}(1-P_{it}) = \beta_{f,it} P_{it}(1-P_{it}) = O_{it}\end{aligned} \qquad \textbf{(2.8)}$$

olarak hesaplanır. β fiyat özelliğinin katsayısını göstermektedir. Bu değişim ürünün kendi fiyatından kaynaklanan esneklik (own elasticity) olarak adlandırılır ve O_{it} olarak gösterilmiştir. *i* ürününe olan talep D_{it} ise, bu değişimden sonraki talep $d_{it} = D_{it} + O_{it} D_{it}$ olacaktır.

Aynı şekilde *j* ürünün bir özelliği değiştiğinde (Fiyat) diğer ürünler üzerindeki etkisi de hesaplanabilir**.**

$$\frac{\partial P_{it}}{\partial f_{jt}} = \frac{\partial(e^{v_{it}} / \sum_k e^{v_{kt}})}{\partial f_{jt}}$$

$$= -\frac{e^{v_{it}}}{(\sum e^{v_{jt}})^2} e^{v_{jt}} \frac{\partial v_{it}}{\partial f_{jt}} \qquad \textbf{(2.9)}$$

$$= -\frac{\partial v_{it}}{\partial f_{jt}} P_{it} P_{jt} = -\beta_{f,jt} P_{it} P_{jt} = E_{it}$$

Buna da çapraz fiyat esnekliği (cross price elasticity) denir ve E_{it} olarak gösterilmiştir. Bu değişimden sonraki talep ise $d_{it} = D_{it} + E_{it} D_{it}$ olacaktır.

İkame etkisi bir ürünün fiyatı düşürüldüğünde diğer ürünün talebinde olan artış veya tam tersi olarak ifade edilmektedir. Yani *j* ürünün fiyatı düştüğünde, *i* ürünün talebi E_{it} oranında azalacak, *j* ürününe bu oranda bir geçiş olacaktır Dolayısıyla ikame etkisi çapraz esneklik tanımından yola çıkarak bulunmuştur.

İncelediğimiz diğer bir etki ise zamandan kaynaklan esnekliktir. Sezon sonuna doğru tüketicilerin satın alma eğilimleri azaldığından zamanın talep üzerinde negatif etkisi olacaktır. Zamanın seçim olasılıkları üzerindeki etkisi ise yine aynı şekilde hesaplanmıştır**.** Zaman parametresi yerine **(2.10)** denkleminde *z* kullanılmıştır.

$$\frac{\partial P_{it}}{\partial z_{it}} = \frac{\partial(e^{v_{it}} / \sum_j e^{v_{jt}})}{\partial z_{it}}$$

$$= \frac{e^{v_{it}}}{\sum e^{v_{jt}}} \frac{\partial v_{it}}{\partial z_{it}} - \frac{e^{v_{it}}}{(\sum e^{v_{jt}})^2} e^{v_{it}} \frac{\partial v_{it}}{\partial z_{it}} \qquad \textbf{(2.10)}$$

$$= \frac{\partial v_{it}}{\partial z_{it}} (P_{it} - P_{it}^2)$$

$$= \frac{\partial v_{it}}{\partial z_{it}} P_{it}(1 - P_{it}) = \beta_{z,it} P_{it}(1 - P_{it}) = T_{it}$$

Zaman esnekliği T_{it} ile gösterilmiştir. Bu değişimden sonraki talep ise $d_{it} = D_{it} + T_{it} D_{it}$ olacaktır.

2.4 Multinomial Logit Modelinde (MNL) Talep Tahmini

Tahmin metodu mevcut olan verinin tipine bağlıdır. Satış verilerinde mevcut olan bilgi panel verilerinden farklıdır. Panel verileri karar vericilerin satın alma davranışlarını belli bir zaman süresince mağazalarda özel kartlar gibi farklı yollarla takip edilmesiyle elde edilir ve farklı bir yaklaşım gerektirmektedir. Diğer bir yaklaşımda Kök ve Fisher (2007)' de ele alınan yaklaşımdır. Bir talep modelinin parametrelerini tahmin için mevcut olan veriler tipik olarak bir gündeki mağazayı ziyaret eden müşteri sayısını, her ürün satışını, ayrıca talebi etkileyen tatil, hava şartları, fiyat ve promosyon gibi pazarlama değişkenlerini içermektedir.
Tüketici satın alma davranış modeli üç karara bağlıdır:

1. Ürün alt kategorisinden ürün satın alınıp alınmadığı (satın alma olasılığı)
2. Satın alma olasılığı verildiğinde hangi ürünün alındığı
3. Hangi üründen kaç tane alındığı

Bu hiyerarşik model pazarlama literatüründe oldukça standart ve kullanılan bir modeldir. *j* ürünü için talep aşağıdaki gibidir:

$$D_j = K(PQ)_j = K\pi p_j q_j \qquad \textbf{(2.11)}$$

K verilen bir gündeki gelen müşteri sayısını, $(PQ)_j$ her müşteri için *j* ürününe ait ortalama talebini, π satın alma olasılığını (gelen bir müşterinin bir ürün alt kategorisinden herhangi bir şey alma olasılığı), p_j seçim olasılığını (satın alma olasılığı verildiğinde bir müşterinin *j* ürününü seçme olasılığı), ve q_j , satın alma olasılığı ve *j* ürününün seçildiği bilgisi verildiğinde bir müşterinin o üründen kaç tane aldığını göstermektedir.

p_j, **(2.4)** denklemindeki gibi ifade edildiğinde ürün seçimi MNL çerçevesinde modellenebilir. Bir müşterinin *j* ürününe ilişkin ortalama faydası u_j, ürünlerin fiyatlarının ve satın alınma zamanının bir fonksiyonu olarak düşünülmüştür. *t* zaman indeksini göstermektedir. O halde p_{jt}

$$p_{jt} = \exp(\alpha_0 + \alpha_{1j} Fiyat_{jt} + \sum_{i \neq j} \alpha_{2i} Fiyat_{it} + \alpha_{3j} Zaman_{jt} + \varepsilon_{jt}) \qquad \textbf{(2.12)}$$

olarak ifade edilen bir üssel fonksiyon ve $i, j \in S$ (S ürün kümesini göstermektedir). Her iki tarafın logaritması alınarak denklem lineer hale getirilebilir:

$$\ln p_{jt} = \alpha_0 + \alpha_{1j} Fiyat_{jt} + \sum_{i \neq j} \alpha_{2i} Fiyat_{it} + \alpha_{3j} Zaman_{jt} + \varepsilon_{jt} \quad \textbf{(2.13)}$$

Ürün kategorisindeki $|S|$ tane ürün için bunu yapıp ortalamasını aldığımızda şöyle bir fonksiyon elde ederiz:

$$\frac{\ln p_{1t} + \ln p_{2t} + ... + \ln p_{|S|t}}{|S|} = \frac{\ln(\prod_{j=1}^{|S|} p_{jt})}{|S|} = \ln (\prod_{j=1}^{|S|} p_{jt})^{1/|S|} = \ln \overline{p}_t \quad \textbf{(2.14)}$$

Fiyat ve zamandan farklı değişkenleri de bu yaklaşıma katmak tabii ki mümkündür. $\overline{p}_t = (\prod_{j \in S} p_{jt})^{1/|S|}$ t anında tüm j ürünleri için geometrik ortalamayı göstermektedir.

Satış verilerinden p_{jt}' yi t anında j ürününü alan müşteri sayısının herhangi bir ürünü alan toplam müşteri sayısına oranla hesaplayabiliriz. Her ürünün seçilme olasılığının ortalamadan farkını fayda olarak kabul edersek fayda fonksiyonunu şu şekilde gösterebiliriz:

$$\ln p_{jt} - \ln \overline{p}_t = \ln\left(\frac{p_{jt}}{\overline{p}_t}\right) = v_{jt} = \beta_{0j} + \beta_{1j} Fiyat_{jt} + \sum_{i \neq j} \beta_{2i} Fiyat_{it} + \beta_{3j} Zaman_j \quad \textbf{(2.15)}$$

β_j katsayıları lineer regresyon ile log-merkezi transformasyonuna uyarlanarak tahmin edilebilir. Bu denklemden her ürün için v_{jt} fayda değerlerini bulabileceğimize göre artık her ürün ve karar verici için logit olasılıkları hesaplanabilir. Bu olasılık değerlerini kullanarak ikame olasılıklarını **(2.9)** ve taleplerini bulabiliriz.

Satış verilerinden yine aynı mantıkla satın alma olasılığı π_t hesaplanabilmektedir. Mağazaya gelip alışveriş yapan müşteri sayısının o gün mağazaya gelen tüm müşteri sayısına oranla bu değer bulunabilir. Katsayıları tahmin edebilmek için lojistik (logistic) regresyon denklemi kullanılmıştır:

$$\ln\left(\frac{\pi_t}{1-\pi_t}\right) = \vartheta = \alpha_0 + \alpha_1 T_t + \alpha_2 HDI_t + \sum_k \gamma_k G_t + \alpha_{4t}\bar{A}_t + \sum_l \beta_l E_t \qquad \textbf{(2.16)}$$

t hava sıcaklığını, *HDI* (Kişi rahatsızlık indeksi, güneş ışığı nem gibi), *G* haftanın günlerini 0-1 değeri olarak, *E* Yılbaşı gibi tatil günlerini 0-1 olarak ifade etmektedir. Yine farklı değişkenler uygun olarak denkleme katılabilir.

q_{jt} satış verilerinden satılan *j* ürün sayısının *j* ürününü alan müşteri sayısına oranla hesaplanabilir. Lineer regresyon yardımıyla katsayılar tahmin edilir:

$$q_{jt} = \alpha_{0j} + \alpha_{1j} A_{jt} + \alpha_{2j} HDI_t + \sum_l \beta_{jl} E_t \text{ , tüm } j \in S \text{ için} \qquad \textbf{(2.17)}$$

K_t *t* anında işlem yapan tüm müşteri sayısını ifade etmektedir. Log-lineer regresyon kullanarak katsayılar tahmin edilir:

$$\ln(K_t) = \alpha_0 + \alpha_1 T_t + \alpha_2 HDI_t + \sum_k \gamma_k G_t + \sum_l \beta_{lt} E_t \qquad \textbf{(2.18)}$$

Bu dört aşamalı model talep tahmininde kullanılmaktadır (Kök ve Fischer, 2006; Kök ve Fischer, 2007).

3. ÖDÜLLÜ ÖĞRENME

3.1 Markov Karar Süreci

Verdiğimiz kararların o anda ve uzun vadede birçok sonuçları vardır. Kararlar her şeyden bağımsız olarak ele alınmamalıdır. Bugün verilen karar yarın verilecek bir kararı, yarın verilecek karar bir sonraki gün verilecek kararı etkiler. Bugün ve gelecekte verilecek kararlar arasında bir ilişki kurmazsak iyi sonuç elde edemeyebiliriz.

Sıralı karar modeli olarak bilinen Markov Karar Süreci (MDP) kararların belirsiz etkilere sahip olduğu, durumlar arasında stokastik geçişlere neden olduğu ve sistemde içinde bulunduğumuz durumun belli bir olasılıkla bilindiği stokastik bir kontrol sürecidir. Bir MDP modelinde 5 parametre vardır: Karar dönemleri, durumlar, kararlar, geçiş olasılıkları, ve gelirler. Her $t \in T$ karar döneminde sistem bir *s* durumunu işgal eder. Karar verici $s \in S$ durumunu inceler ve bu *s* durumunda mevcut alınabilecek kararlardan bir $a \in A_s$ kararını seçebilir. *t* anında *s* durumunda $a \in A_s$ kararının seçilmesi sonucunda, karar verici bir $r_t(s,a)$ geliri elde eder ve bir sonraki s' durumuna geçer. Bu geçiş olasılığı $p_t(s' \mid s,a)$ ile ifade edilir. *t* anında *s* ve *a* bilindiği takdirde geçiş olasılığı daha önceki hiçbir durum ve karardan etkilenmez. Yani bir MDP'nin durum geçişleri *Markov özelliğini* taşımaktadır. *X* markov özelliği taşıyan bir stokastik süreci ifade etsin. Bu durumda sistemin *t* anında x_t durumunda bulunma olasılığı sadece $t-1$ anındaki durumdan etkilenir**.**

$$P(X_t = x_t \mid X_{t-1} = x_{t-1}, ..., X_0 = x_0) = P(X_t = x_t \mid X_{t-1} = x_{t-1}).$$

Bu parametreler kümesi $\{T, S, A_s, p_t(s' \mid s,a), r_t(s,a)\}$ bir Markov Karar Sürecini ifade eder. MDP'lerde asıl amaç karar verici için en iyi politika bulmaktır. Politika π tüm karar dönemlerinde kullanılan bir karar kurallar kümesidir ve *t* anında alınan en iyi a_t^* kararlarından oluşur, $\pi = (a_1^*, a_2^*, ..., a_T^*)$.

MDP problemleri Dinamik Programlama (politika yineleme ve değer yineleme algoritmaları) ile çözülebilir (Puterman, 1994). Fakat *"boyutun laneti"* olarak bilinen durum, karar ve dışarıdan gelen bilgi uzayı çok büyüdüğünde Dinamik programlama bir çözüm sağlamayacaktır. Çünkü Dinamik programlama tüm durumlar için geçiş olasılıkları gerektirmektedir. Bu durumda geçiş olasılıklarının hesaplanmasını gerektirmeyen Ödüllü Öğrenme (ÖÖ) algoritmaları ile probleme çözüm getirilmektedir.

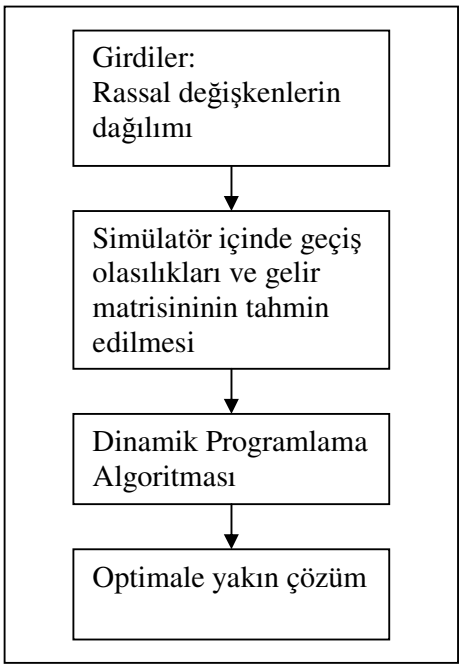
Ödüllü Öğrenme ortamda herhangi bir öğretici olmasını gerektirmeyen, sistemin dış dünya ile olan deneysel etkileşimini kullanan deneysel bir öğrenme metodudur.
ÖÖ yapay zeka sinir ağları, istatistiksel model tanıma, makine öğrenmesi gibi çok yakın zamanda çalışılan bir öğrenme şekli olan yönetilen öğrenme (supervised learning)' den farklıdır. Yönetilen öğrenme, bilgili, dışarıdan bir yöneten tarafından sağlanan bir öğrenmedir. Önemli bir öğrenmedir, fakat tek başına etkileşimli öğrenme için yetersizdir. Bu öğrenme tipinde tüm bilgiler sisteme verilir ve bir bilgi istendiğinde bu bilgilere göre cevap alınır. Başka bir deyişle yönetilen öğrenme cevaplarıyla birlikte birçok soru gerektirir. Bir soru sorulduğunda da tanımlanan tüm sorulara bakarak sorulan soruyu öğrenir ve cevabı vermeye çalışır. Ancak bazı durumlarda bu öğrenme yoluyla doğru cevaplar bulunamayabilir. Mesela sinir ağları elimizde belli bir bilgi birikimi yoksa buna cevap bulamaz dolayısıyla yönetilen öğrenme yöntemi ihtiyacımızı karşılamaz (Harmon ve diğ., 2000). ÖÖ'de ise öğrenen (learning agent) tüm bilgileri kullanarak tecrübe edinir ve tüm edindiği tecrübeleri kullanarak zamanla performansını geliştirmeye çalışır.

ÖÖ'de ana araç simülasyondur. Geçiş olasılıklarının hesaplanması gerekliliğini ortadan kaldırmak için simülasyon kullanır. Simülatör sistemin davranışını yöneten rassal değişkenlerin dağılımına ihtiyaç duyar. Geçiş süreleri ve gelirleri simülatör içinde otomatik olarak hesaplanır.

Simülatörler içinde geçiş olasılıklarını tahmin etmek için bir i durumunun sonsuz sayıda incelenmesi $(V(i,a))$ gerekir. i durumundan j durumuna kaç kez gidildiği $(W(i,a,j))$ bilindiği takdirde geçiş olasılıkları $p(i,a,j)$ yaklaşık olarak hesaplanabilir:

$$p(i,a,j) \simeq \frac{W(i,a,j)}{V(i,a)} \qquad \textbf{(3.1)}$$

MDP modelinde geçiş matrisi bilindikten sonra artık DP kullanılarak çözüme ulaşılabilir. Fakat durum uzayı büyük olduğunda her durumun geçiş olasılığını hesaplamak ve bunu hafızada tutmak zor ve masraflı olduğundan durum-kümeleme yaklaşımı kullanılarak bu problem ortadan kaldırılabilir. Kümeleme ile durum sayısı azalacak ve daha yönetilir hale gelecektir. İleriki bölümlerde kümeleme tekniğine daha detaylı değinilecektir. Şekil 3.1'de bir simülatör içinde MDP'nin çözüm yöntemi adımları gösterilmiştir. Şekil 3.2' de ise ÖÖ ve DP yöntemleri arasındaki farklılıklara değinilmiştir.

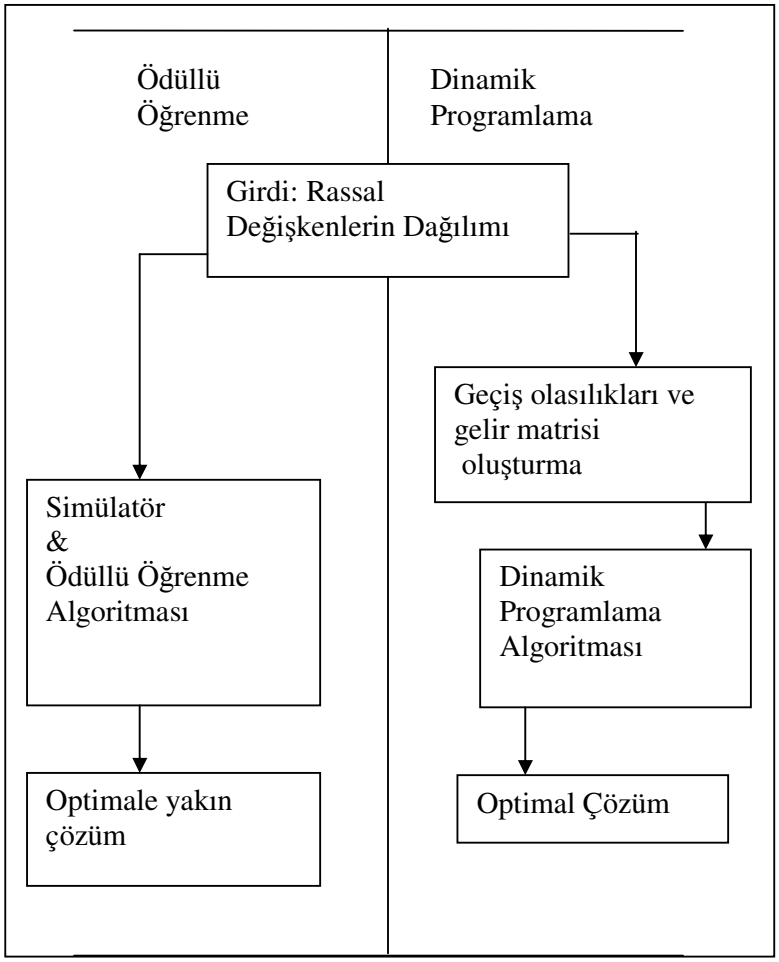


Şekil 3.1 : Simülatör içinde MDP'nin çözüm yöntemi adımları (Sutton ve Barto, 1998).

3.2 Neden Ödüllü Öğrenme ?

Dinamik Programlama (DP), MDP ve yarı-MDP' lerin optimal çözümleri için garanti vermektedir. Fakat geçiş olasılıklarının, geçiş sürelerinin ve geçiş gelirlerinin elde edilmesi gerçekten zordur ve karmaşık bir matematik gerektirmektedir. DP bahsedilen bu niceliklerin değerlerine ihtiyaç duymaktadır. ÖÖ ise buna ihtiyaç duymaz, optimale yakın çözümler üretir.

Geçiş olasılıkları, geçiş süreleri ve gelirleri birlikte sistemin teorik modelini oluştururlar. MDP modelinin kullanılmasıyla bir kontrol-optimizasyon problemi optimal çözümler sağlar. Fakat tüm kontrol-optimizasyon problemlerine MDP uygulanamaz. Çünkü karmaşık sistemlerde MDP formülasyonu için gerekli olan teorik modeli yapılandırmak çok zordur. Bu yüzden gerçek hayatta genellikle sezgisel yaklaşımlar kullanılır. İşte ÖÖ teorik modele gerek duymadan MDP çözen yöntem olarak sunulmaktadır. ÖÖ geçiş olasılıklarına ihtiyaç duymamaktadır.



Şekil 3.2 : Ödüllü Öğrenme ve Dinamik Programlama arasındaki farklılıklar (Sutton ve Barto, 1998).

1000 durum ve her duruma ait 2 karar olan bir sistemde bile $1000^2 = 10^6$ tane durum oluşacaktır ve DP bunu çözemeyecektir. Sistem çok boyutlu olduğunda durum sayısı dahada artacaktır. Bu problem 'boyutun laneti' olarak bilinmektedir.

ÖÖ bu zorlukların üstesinden 2 şekilde gelmektedir:

1. Modelden bağımsız olan ÖÖ geçiş olasılıklarına ihtiyaç duymaz. Bu yüzden bu problemin üstesinden gelebilir.
2. ÖÖ DP'nin değer fonksiyonunda kullanılan elemanlara ihtiyaç duymaz. $Q-$faktörleri olarak bilinen bir yapıyla değer fonksiyonunu saklar. MDP'de milyon sayıda durum olduğunda, ÖÖ bu $Q-$faktörlerinin hepsini depolamaz. Yapay sinir ağları, regresyon, interpolasyon, kümeleme gibi fonksiyon yaklaşımı tekniklerini kullanarak gerektiğinde istenilen durumlara kolayca erişebilir.

ÖÖ yaklaşık sonuçlar veren bir yöntem olmasına rağmen, MDP modellerini kullandığından sezgisel yaklaşımlardan daha güçlü bir modeldir ve optimale yakın sonuçlar sağlamaktadır. Sezgisel algoritmalar genellikle sistem hakkında birçok varsayım yaparak çözüme giderler ve optimal çözümden uzaktırlar. Sonuç olarak şöyle diyebiliriz: Geçiş olasılıklarının zor hesaplandığı problemlerde, sezgisellere göre daha güçlü ve optimale yakın çözümler saplayabilen ÖÖ kullanmak iyi bir çözüm olacaktır.

3.3 ÖÖ'nün Elemanları

Öğrenen (learning agent) ve öğrenme ortamının (environment) ötesinde bir ÖÖ sisteminde 4 ana alt eleman tanımlanabilir: politika, gelir fonksiyonu, değer fonksiyonu, ve opsiyonel olarak öğrenme ortamının modeli. ÖÖ, durum, karar ve gelir açısından öğrenen ve ortamı arasındaki ilişkiyi tanımlayan bir yapı kullanır. Bu yapı basit bir yapay zeka probleminin gerekli özelliklerini gösterir.

Politika belli bir anda öğrenenin nasıl davranacağını belirler. Kabaca ifade edecek olursak, politika ortamda incelenen durumları eşleştirdiğimiz kararlardır. Bazı durumlarda basit bir fonksiyon veya tablo formu olabileceği gibi bazen de bir araştırma süreci gibi kapsamlı bir hesaplama içerebilir. Sadece politika ile nasıl davranılacağına karar verilebileceği için ÖÖ'nün temelini oluşturur. Politikalar stokastik de olabilir.

Gelir fonksyionu bir ÖÖ probleminde amacı tanımlar. Yani incelenen durumun getirisini ifade eder, böylelikle o durumun ne derece önemli olduğu anlaşılır. ÖÖ'de öğrenenin asıl amacı uzun dönemde toplam geliri enbüyüklemektir. Politikayı oluştururken temel teşkil eder. Gelir fonksiyonları da stokastik olabilir.

Gelir fonksiyonu o anda neyin iyi olduğunu gösterirken, ***değer fonksiyonu*** ise uzun koşumda neyin iyi olduğunu gösterir. Kabaca ifade edecek olursa, bir durumun değeri bulunduğumuz andan dönem sonuna kadarki toplam geliri ifade eder. Mesela bir durum sürekli düşük bir gelir sağlarken yüksek bir değere sahip olabilir. Çünkü bu durumu takip eden diğer durumlar yüksek gelir sağlayabilirler. Ya da tam tersi doğru olabilir. Gelir olmadan değer oluşmaz, ve değer tahmin ederken tek amaç daha fazla gelir sağlamaktır. Karar seçimleri değerlere bakılarak yapılır. Karar verirken en yüksek geliri değil en yüksek değeri oluşturan kararlara bakılır, çünkü bu kararlar uzun koşumda büyük gelir sağlayan kararlardır. Değerlere karar vermek gelirlere karar vermekten daha zordur. Çünkü gelirler bulunulan ortamda direkt olarak oluşur fakat değerler öğrenicinin tüm süreç boyunca yaptığı bir dizi incelemelerden tahmin edilir. Aslında tüm ÖÖ algoritmalarının en önemli bileşeni bu değerleri en iyi, verimli şekilde tahmin etme yöntemidir.

Genetik algoritmalar, benzetim tavlama ve diğer fonksiyon eniyileme yöntemleri gibi arama yöntemleri ÖÖ problemlerini çözmek için kullanılabilir. Fakat bunlar değer fonksiyonlarına başvurmadan politika uzayında doğrudan arama yaparlar. Bu evrimsel yöntemler politika uzayı yeterince küçük olduğunda ve dolayısıyla iyi politikaları bulmak kolay olduğunda etkin olabilirler.

ÖÖ ortamla ilişki kurarak öğrenir, evrimsel yöntemlerde böyle bir şey yoktur. Evrimsel yöntemler ÖÖ'nün kullanışlı olan bazı yapılarını ihmal ederler: ÖÖ algoritmasında hangi durumlarda hani kararlar alındığı önemliyken ve bunlar hafızada tutulurken, evrimsel yöntemlerde bu durumlar ve kararlar tutulmaz.

Bazı ÖÖ sistemler için dördüncü ve son eleman ortamın ***model***idir. Model ortamın davranışını gösterir. Mesela bir durum ve karar verildiğinde model bir sonraki durumu ve geliri tahmin edebilir. Modeller planlama için kullanılırlar. ÖÖ sistemleri o anda deneme yanılma yöntemiyle öğrenirler, ortamın modelini oluştururlar ve bu modeli daha sonra planlama için kullanırlar.

3.4 Öğrenen – Ortam Etkileşimi

Ödüllü öğrenmede öğrenen ve aktif öğrenme sürecini sağlayan ortam olmak üzere iki nesne vardır. Her bir zaman diliminde öğrenici o anki durumunu (state) değerlendirerek bir karar (action) seçer ve ortamdan aldığı geri beslemeyi (reward)

sisteme uygular. Öğrenenin amacı herhangi bir durumda en iyi davranışı seçebilecek duruma gelerek daha önceden belirlenmiş olan bir performans ölçütünü eniyilemektir. En iyi karar dizisini belirlemek için her durumda yeterli sayıda güncelleme yapmak gerekir. İdeal olanı sonsuz sayıda güncelleme olmasına rağmen bu mümkün olmadığından daha önceden yeterli olacağı varsayılmış sayıda güncelleme yapma yoluna gidilmektedir.

Tüm ödüllü öğrenme metotları dış ortamla deneme yanılma yoluyla sıralı karar işlemleri dizisini çözme amacını içermektedir. Bir sıralı karar işleminde, öğrenici *t* anında $a_t \in A(s_t)$ kararını seçerek s_t durumunu inceler ve sistem bu inceleme sonucunda öğreniciye sayısal kar veya maliyet gibi bir geri bildirim ($r(s_t)$) sağlar. Seçilen her karar sistemi başka bir s_{t+1} durumuna götürür.

Öğrenicinin amacı her duruma bir karar atayan, $\pi : S \to A$ karar politikası oluşturmaktır. Dolayısıyla ÖÖ karı maksimize edecek bu politikanın nasıl oluşulacağına dair bir öğrenme şeklidir.

Bu optimal π^* politikasının nasıl bulunduğuna yönelik iki ana yaklaşım kullanılmaktadır. Birincisi karar politikası uzayını araştıran politika yineleme, diğeri de değer fonksiyonları uzayını araştıran değer yineleme yöntemidir. Her iki yöntemde dış ortamdaki tüm bilgiler verildiği takdirde sonlu MDP' lerde optimal politika ve değer fonksiyonlarını hesaplamak için kullanılabilir (Sutton, Barto, 1998; Moriarthy ve diğ., 1999).

3.5 Politika Yineleme Yöntemi

Politika yineleme yöntemi politika değerlendirme ve politika iyileştirme olmak üzere 2 kısımdan oluşur. Öncelikle bir π politikası belirlenir ve bu politikayı kullanarak hesaplanan V^{π} değer fonksiyonlarıyla daha iyi bir π' politikası sağlanır. Aynı şekilde π' politikasını kullanarak hesaplanan $V^{\pi'}$ değerleriyle daha iyi bir π'' politikası sağlanır ve bu durum politikada bir iyileşme sağlanamayana dek devam eder. π politikası iyileştirilmeden önce gerçekleştirilen iterasyonlarla V^{π} değer fonksiyonlarının bir yakınsama göstermesi (policy evaluation) sağlanır ve politika güncellenir.

3.5.1 Politika değerlendirme

Hayali bir π politikası için V^{π} durum-değer fonksiyonlarının hesaplanması gerekmektedir. Dinamik Programlama (DP) literatüründe buna politika değerlendirme denir. Bunu bir tahmin problemi olarak ele alabiliriz. r_t, t anındaki geliri, γ indirim faktörünü göstersin. V^{π} durum-değer fonksiyonları tüm $s \in S$ durumları için aşağıdaki gibidir.

$$\begin{aligned} V^{\pi}(s) &= E_{\pi}\{r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \ldots \mid s_t = s\} \\ &= E_{\pi}\{r_{t+1} + \gamma V^{\pi}(s_{t+1}) \mid s_t = s\} \end{aligned} \quad \textbf{(3.2)}$$

$$= \sum_a \pi(s,a) \sum_{s'} P^a_{ss'}[R^a_{ss'} + \gamma V^{\pi}(s')] \quad \textbf{(3.3)}$$

$\pi(s,a)$, π politikası altında s durumunda a kararının alınma olasılığını göstermekte ve π indisli gösterilen değişkenlerde o fonksiyonların π politikası altındaki değerlerini ifade etmektedir. $P^a_{ss'}$ s durumunda a kararı alındığında s' durumuna geçiş olasılığını ifade etmekte ve $P^a_{ss'} = \Pr\{s_{t+1} = s' \mid s_t = s, a_t = a\}$ şeklinde gösterilmektedir. $R^a_{ss'}$ o anki beklenilen gelir olarak ifade edilmekte ve $R^a_{ss'} = E\{r_{t+1} \mid a_t = a, s_t = s, s_{t+1} = s'\}$ şeklinde gösterilmektedir. Bu V^{π} değerlerin hesabı için sıralı çözüm metotları daha uygundur. V_0, V_1, V_2 gibi bir yaklaşık değer fonksiyonu dizisi düşünelim. V_0 ilk yaklaşık değeri hayali olarak seçilir ve her ardışık yaklaşık değer Bellman Denklemine **(3.4)** göre her $s \in S$ için güncellenir:

$$V_{k+1}(s) = \sum_a \pi(s,a) \sum_{s'} P^a_{ss'}[R^a_{ss'} + \gamma V_k(s')] \quad \textbf{(3.4)}$$

Dolayısıyla V_k değerini kullanarak V_{k+1} değerleri ardışık olarak hesaplanır. Her iterasyondan sonra $\max_{s \in S} |V_{k+1}(s) - V_k(s)|$ değerlerine bakılır, ne zaman ki bu değer yeterince küçükse algoritma sonlanır.

3.5.2 Politika iyileştirme

Bir politika için değer fonksiyonlarını hesaplamamızın nedeni daha iyi politikalar bulmaktır. Deterministik hayali bir π politikası için V^{π} değer fonksiyonlarını bildiğimizi farz edelim. Herhangi bir s durumu için mevcut politikanın değiştirilip değiştirilemeyeceğini o durum için geçerli kararlar kümesindeki kararları deneyerek bulabiliriz. Bu $Q^{\pi}(s,a)$ karar-değer fonksiyonları şu şekilde hesaplanır:

$$\begin{aligned} Q^{\pi}(s,a) &= E_{\pi}\{r_{t+1} + \gamma V^{\pi}(s_{t+1}) \mid s_t = s, a_t = a\} \\ &= \sum_{s'} P^{a}_{ss'}[R^{a}_{ss'} + \gamma V^{\pi}(s')] \end{aligned} \tag{3.5}$$

(3.5) denklemi kullanılarak tüm $s \in S$ durumları ve tüm olağan $a \in A(s)$ kararları için yeni bir politika oluşturulur **(3.6)**. Buna miyopik politika denir.

$$\begin{aligned} \pi'(s) &= \arg\max_{a} Q^{\pi}(s,a) \\ &= \arg\max_{a} \sum_{s'} P^{a}_{ss'}[R^{a}_{ss'} + \gamma V^{\pi}(s')] \end{aligned} \tag{3.6}$$

Artık $V^{\pi'}(s)$ şu şekilde ifade edilebilir:

$$V^{\pi'}(s) = \max_{a} \sum_{s'} P^{a}_{ss'}[R^{a}_{ss'} + \gamma V^{\pi'}(s')] \tag{3.7}$$

Eğer **(3.8)** sağlanıyorsa bulunan miyopik politika orijinal politikadan daha iyidir ya da onun kadar iyidir denilebilir. Orjinal politikanın değer fonksiyonlarına göre miyopik ya da yaklaşık miyopik ile yeni bir politika geliştirme sürecine *politika iyileştirme* denir.

$$Q^{\pi}(s, \pi'(s)) \geq V^{\pi}(s) \tag{3.8}$$

Böylelikle tüm $s \in S$ durumları için **(3.9)** sağlanmış olur ve bu da politikanın iyileştiğini gösterir.

$$V^{\pi'}(s) \geq V^{\pi}(s) \quad , \forall s \in S \tag{3.9}$$

Politika yineleme yönteminin dezavantajlarından biri politika değerlendirme kısmında V^{π} değerleri için gerçekten bir yakınsamanın beklenmesidir. Yani yakınsamanın gerçekleşmesi için çok sayıda iterasyon yapılması gerekebilir. Pratikte bu yukarıda da bahsedildiği gibi iyileşme kaydedilmeyene dek yapılarak önlenmeye çalışılır.

3.6 Değer Yineleme Yöntemi

Politika yineleme yöntemi politikada bir değişiklik olmayana dek devam etmektedir. Bu yüzden her politika iyileştiğinde politika değerlendirme yapılması gerekmektedir. Değer yineleme yönteminde ise öncelikle V değer fonksiyonları bulunur **(3.10)** ve her durum için bu değer değişmeyene dek iterasyon gerçekleştirilir.

$$V(s) = \max_{a} \sum_{s'} P_{ss'}^{a} [R_{ss'}^{a} + \gamma V(s')] \qquad \textbf{(3.10)}$$

Son olarak politika oluşturulur **(3.11).** Politika yinelemede olduğu gibi politika iyileştirilmesine gidilmez.

$$\pi(s) = \arg\max_{a} \sum_{s'} P_{ss'}^{a} [R_{ss'}^{a} + \gamma V(s')] \qquad \textbf{(3.11)}$$

Görüldüğü gibi DP'de tüm güncellemeler sonraki oluşan durum değerlerine dayanarak yapılmaktadır. Bu genel fikre önyükleme (bootstrapping) denir. Çoğu ÖÖ algoritmaları da önyüklemeyi gerçekleştirmektedir, fakat DP'de olduğu gibi tüm modele (geçiş matrisi gibi) ihtiyaç yoktur.

ÖÖ' de asıl amaç deneylerle bu değer fonksiyonlarını öğrenen algoritmalar tasarlamaktır. Bunun için kullanılan en yaygın yaklaşımlar Monte Carlo ve Zamansal Fark (Temporal Difference) (TD) yöntemleridir.

3.7 Monte Carlo Yöntemleri

Monte Carlo yöntemleri sadece deneyim gerektirirler: dış ortamla gerçekleştirilen etkileşimlerden sağlanan durum (states), karar (actions) ve gelirlerden (rewards) oluşan örneklem dizileri. Benzetim yardımıyla deneyim kazanma güçlü bir öğrenmedir. Çünkü dinamik programlamada olduğu gibi durum geçişlerinde tüm

olağan geçişlerin geçiş olasılıklarının bilinmesine gerek yoktur, sadece örneklemin geçişini bilmek yeterlidir. Monte Carlo yöntemleri ÖÖ problemini çözmekte kullanılan ve aynı durum incelendiğinde bu durumun (örneklemin) örneklem gelirlerinin (sample returns) ortalamasını almaya dayanan bir yöntemdir. Değer fonksiyonlarını hesaplama dışında Dinamik Programlamadaki (DP) tüm fikirler Monte Carlo için geçerlidir. Değer fonksiyonu tahminleri diğer değer fonksiyonlarına dayanarak güncellenmez (bootstrapping) (Sutton ve Barto, 1988).

3.7.1 Monte Carlo politika değerlendirme

Bir durumun değeri o durumdan başlayarak son periyoda kadarki gelirlerin beklenen değeri olarak ifade edilmektedir. Bunu deneyimlerden tahmin etmenin en açık yolu o duruma tekrar gelindiğinde tüm gelirlerin ortalamasını almaktır. O duruma ne kadar çok gelinirse ortalama beklenen değere o kadar çok yaklaşır. Monte Carlo (MC) yöntemlerinin temel fikrini de bu oluşturur.

Özellikle bir *s* durumunun π politikası altındaki değeri olan $V^{\pi}(s)$ durum değer fonksiyonunu tahmin etmek istediğimizi farz edelim. Öncelikle π politikasını takip eden bir epizod (olaylar zinciri) seçilir. Bu epizodda *s* durumunda bulunmaya ziyaret (visit) denir. MC yöntemine göre tüm epizodlar kümesinde *s* durumuna tüm ziyaretlerde (every-visit MC) oluşan gelirlerinin ortalamasıyla $V^{\pi}(s)$ tahmin edilir. Verilen bir epizodda *s* durumun ilk kez ziyaret edilmesine *s* durumuna ilk ziyaret (first-visit MC) denir. İlk-ziyaret MC yöntemi sadece *s* durumuna ilk ziyaretler sonucu oluşan gelirlerin ortalamasını alır. Bu her iki yöntem bazı küçük teorik özellikler dışında çok benzerdir. Her iki yöntemde de *s* durumuna ziyaret sayısı arttıkça $V^{\pi}(s)$ bir değere yakınsar. İlk-ziyaret MC yöntemi için bunu görmek çok kolaydır. Bu durumda her gelir $V^{\pi}(s)$'den ayrı ve bağımsız dağılmıştır. Büyük sayılar kanununa göre bu tahminlerin ortalamasının sırası bunların beklenen değerine yakınsar. Her ortalama kendi başına bir yansız (unbiased) tahmindir, ve bu hatanın standart sapması, *n* ortalaması alınan gelirlerin sayısını göstermek üzere, $1/\sqrt{n}$' dir. Tüm-ziyaretler MC yöntemininki ise daha az anlaşılır olmakla beraber, bunun tahminleri $V^{\pi}(s)$' e asimptotik olarak yakınsar. Şekil 3.3' te ilk-ziyaret MC yönteminin adımları verilmiştir.

İlk değerleri ata:

$\pi \leftarrow$ politika

$V \leftarrow$ hayali bir durum değer fonksiyonu

$Gelirler(s) \leftarrow$ tüm $s \in S$ için boş bir liste

Sonsuz sayıda aşağıdaki adımları tekrarla:

a) π politikasını kullanarak bir epizod (olaylar zinciri) türet

b) Epizodda oluşan her s durumu için:

$R \leftarrow s$ durumuna ilk ziyarette oluşan gelir

$Gelirler(s) \leftarrow R$

$V(s) \leftarrow$ ortalama($Gelirler(s)$)

Şekil 3.3 : V^{π} tahmini için ilk-ziyaret MC yöntemi (Sutton ve Barto, 1998).

3.7.2 Monte Carlo ile karar-değerleri tahmini

Eğer bir model yoksa durum değerleri yerine karar değerleri tahmin etmek için en kullanışlı yollardan biridir. Bir model olduğunda, politikaya karar vermek için durum değerleri tek başına yeterlidir; DP' de olduğu gibi sadece bir adım sonrasına bakılır ve en iyi gelir ve gelecek durum kombinasyonunu sağlayan karar seçilir. Fakat bir model yoksa durum değerleri tek başına yeterli değildir. Bir politika önermek için tüm karar değerlerinin açıkça tahmin edilmesi gerekir. Böylece MC yöntemleri için asıl amaçlardan biri $Q*$ değerlerini (durum-karar değerleri) tahmin etmektir. Bunun için diğer bir politika değerlendirme problemi ele alınmalıdır.

Karar değerleri için politika değerlendirme problemi s durumunda a kararının verilmesi ve π politikasının takip edilmesiyle oluşan beklenen karar-değerleri $Q^{\pi}(s,a)$'in tahmin edilmesidir. Burada uygulanacak Monte Carlo yöntemleri durum değerlerinin tahmininde kullanılan yöntemlerle aslında aynıdır. Tüm-ziyaretler MC yöntemi bir kararın alınmasıyla incelenen durum ziyaret edildiğinde oluşan gelirlerin ortalamasıyla bulunan durum-karar çiftinin değerini tahmin eder. İlk-ziyaret MC yöntemi de karar seçilip incelenen durum ziyaret edildiğinde her epizoddaki ilk ziyaretlerin gelirlerinin ortalamasını alır. Her iki yöntemde karar-değer çiftlerinin ziyaret sayısı sonsuza giderken beklenilen doğru değere kuadratik olarak yakınsar.

Fakat bazı karar-değer çiftleri hiç ziyaret edilmemiş olabilir. π determistik bir politikaysa, bu politikaya bağlı olarak her durum için sadece bir kararın geliri

gözlenebilir. Ayrıca gelirler ortalama alınmadan oluşmuşsa MC yöntemi deneyimlerle iyileşmemiş olan kararları tahmin edecektir. Bu ciddi bir problemdir. Alternatifleri kıyaslayabilmek için her durumun tüm karar değerlerinin tahmin edilmesi gerekir.

Politika değerlendirirken karar değerleriyle çalışabilmek dolayısıyla tüm karar değerlerini görebilmek için karar keşiflerine (exploration) devam etmemiz gerekir. Bunu gerçekleştirmenin bir yolu epizodun ilk adımında bir durum-karar çiftiyle başlamaktır. Böylece başlangıçta bütün durum-karar çiftleri sıfırdan farklı bir seçilme olasılığına sahip olur. Sonuç olarak sonsuz sayıda epizod incelendiğinde durum-karar çiftlerinin sonsuz sayıda ziyaret edilmesi garanti edilmiş olur. Bu varsayım keşif yöntemi (exploration starts) olarak adlandırılır.

3.7.3 Monte Carlo kontrolü

Monte Carlo'da yaklaşık optimal politikalar dinamik programlamada olduğu gibi genelleştirilmiş politika yineleme (GPI) fikrine dayalı olarak oluşturulur. GPI' da hem yaklaşık bir politika hem de yaklaşık değer fonksiyonu bulunur. Değer fonksiyonu o anki politika için değer fonksiyonunu daha doğru bulabilmek için sürekli değişir ve politika da aynı şekilde o anki değer fonksiyonuna göre sürekli iyileşir.

Bu değerlendirme ve iyileştirme operasyonları birbirlerine karşı sürekli bir hedef oluşturarak devam ederler ve sonunda politika ve değer fonksiyonları optimale yaklaşmış olurlar.

İlk olarak Monte Carlo'yu klasik politika yineleme olarak ele alalım. Bu yöntemde hayali bir π_0 politikası ile başlanır, politika değerlendirme (*D*) ve politika iyileştirme (*I*) adımlarını gerçekleştirdikten sonra optimal π^* politikası ve optimal Q^* karar-değer fonksiyonu ile tamamlanır:

$$\pi_0 \xrightarrow{D} Q^{\pi_0} \xrightarrow{I} \pi_1 \xrightarrow{D} Q^{\pi_1} \xrightarrow{I} \ldots \xrightarrow{I} \pi^* \xrightarrow{D} Q^*$$

Politika değerlendirme aynı, önceki bölümde anlatıldığı gibi yapılmaktadır. Fakat şimdi keşif yapılarak sonsuz sayıda epizod incelenir. Bu yaklaşımlar altında Monte

Carlo yöntemi hayali bir π_k politikası altında tüm Q^{π_k} karar-değer fonksiyonlarını hesaplayacaktır.

Politika iyileştirme o anki değer fonksiyonuna göre miyopik politika yapılarak gerçekleştirilir. Bu durumda bir karar değer fonksiyonumuz vardır ve bu yüzden bir miyopik politika yapılandırmamıza gerek yoktur. Herhangi bir karar değer fonksiyonu Q için, ona ilişkin miyopik politika tüm $s \in S$ için deterministik olarak Q değerini maksimize eden kararı seçer:

$$\pi(s) = \arg\max_a Q(s,a) \qquad \textbf{(3.12)}$$

Politika iyileştirme daha sonra her π_{k+1} politikasının $Q^{\pi k}$'a göre yapılandırılmasıyla yapılabilir. Böylelikle politika iyileştirme teoremi tüm $s \in S$ için π_k ve π_{k+1}' e uygulanır:

$$\begin{aligned} Q^{\pi k}(s, \pi_{k+1}(s)) &= Q^{\pi k}(s, \arg\max_a Q^{\pi k}(s,a)) \\ &= \max_a Q^{\pi k}(s,a) \\ &\geq Q^{\pi k}(s, \pi_k(s)) \\ &= V^{\pi}(s) \end{aligned} \qquad \textbf{(3.13)}$$

Bu teorem her π_{k+1} politikasının π_k politikasından daha iyi olduğunu, π_k politikasına eşit olmadığı durumda da her iki politikanın optimal olduğunu söyler. Bu, tüm bu sürecin optimal bir politika ve durum değer fonksiyonuna yakınsamasını sağlar. Böylece Monte Carlo yöntemi hiçbir dışarı kaynaklı bilgiye sahip olmadan sadece örneklem epizodlarının verilmesiyle optimal politikaların bulunmasında kullanılabilir.

Görüldüğü gibi Monte Carlo yönteminde yakınsama için iki varsayım yapılmıştır. Birincisi keşif yöntemi ikincisi de politika değerlendirmenin sonsuz sayıda yapılarak sonsuz sayıda epizodun incelenmiş olmasıdır. İkinci varsayım kolayca ortadan kaldırılabilir ve değer yineleme yönteminde olduğu gibi her politika değerlendirmede tek epizod incelenebilir, fakat sonsuz sayıda yapıldığında ortaya çıkan sonuçlara yakın olması beklenemez. Buna göre her epizoddan sonra incelenen gelirler politika

değerlendirme için kullanılır ve daha sonra politika epizodda incelenen tüm durumlar için iyileştirilir. Monte Carlo Keşif Yöntemi olarak adlandırılan diğer algoritmaya Şekil 3.4'te yer verilmiştir.

Monte Carlo yönteminin yakınsamasında yapılan birinci varsayımın ortadan kaldırılması için tüm kararların sonsuz sayıda seçilmesi sağlanmalıdır. Bunu sağlamak için ise politikaya bağlı (on-policy) ve politikadan bağımsız (off-policy) yöntemler olmak üzere iki yaklaşım önerilebilir.

Tüm $s \in S$ ve $a \in A(s)$ için ilk değerleri ata

Hayali $Q(s,a)$ değerleri ata

Hayali bir $\pi(s)$ politikası belirle

Boş bir $Gelirler(s,a)$ listesi oluştur

Sonsuz sayıda aşağıdaki adımları tekrarla:

a. Keşif ve π politikası kullanarak bir epizod türet

b. Epizodda yer alan her s,a çifti için

$R \leftarrow$ İlk incelenen s,a çiftinin geliri

$Gelirler(s,a) \leftarrow R$

$Q(s,a) \leftarrow ortalama(Gelirler(s,a))$

Epizodda yer alan her *s* için

$$\pi(s) \leftarrow \arg\max_a Q(s,a)$$

Şekil 3.4 : Monte Carlo keşif yöntemi (Suttton ve Barto, 1998).

3.7.4 Politikaya bağlı (on-policy) Monte Carlo kontrolü

Politikaya bağlı yöntemler karar vermek için kullanılan politika değerlendirme ve iyileştirme için girişimde bulunurlar. Bu yöntemlerde genellikle politika yumuşaktır (soft), yani tüm $s \in S$ ve $a \in A(s)$ için politika $\pi(s,a) > 0$' dır. Bu yöntemler üzerine birçok şey uygulanabilir. Bu varyasyonlardan bir tanesi $\varepsilon-$miyopik politikalardır. $\varepsilon-$miyopik politikalara göre çoğu zaman değer fonksiyonunu maksimize eden karar seçilir, ε olasılıkla da rassal bir karar seçilir. $\varepsilon-$miyopik politikalar $\varepsilon-$yumuşak politikalar için bir örnektirler ve bu politikalar tüm durum ve

kararlar için $\varepsilon > 0$ olmak üzere $\pi(s,a) \geq \frac{\varepsilon}{|A(s)|}$ olarak ifade edilirler. Herhangi bir π $\varepsilon-$yumuşak politika için, Q^{π}' a göre oluşturulan $\pi' \varepsilon-$miyopik politikalar π politikasından ya daha iyidir ya da aynıdır.

$$
\begin{aligned}
Q^{\pi}(s,\pi'(s)) &= \sum_{a} \pi'(s,a) Q^{\pi}(s,a) \\
&= \frac{\varepsilon}{|A(s)|} \sum_{a} Q^{\pi}(s,a) + (1-\varepsilon) \max_{a} Q^{\pi}(s,a) \\
&\geq \frac{\varepsilon}{|A(s)|} \sum_{a} Q^{\pi}(s,a) + (1-\varepsilon) \sum_{a} \frac{\pi(s,a) - \frac{\varepsilon}{|A(s)|}}{1-\varepsilon} Q^{\pi}(s,a) \qquad \textbf{(3.14)} \\
&= \frac{\varepsilon}{|A(s)|} \sum_{a} Q^{\pi}(s,a) - \frac{\varepsilon}{|A(s)|} \sum_{a} Q^{\pi}(s,a) + \sum \pi(s,a) Q^{\pi}(s,a) \\
&= V^{\pi}(s)
\end{aligned}
$$

Böylelikle politika iyileştirme teoremine göre, tüm durumlar için $\pi' \geq \pi$, $(V^{\pi'}(s) \geq V^{\pi}(s))$, olduğu görülür.

3.7.5 Politikadan bağımsız (off-policy) Monte Carlo kontrolü

Politikaya bağlı yöntemlerin politikadan bağımsız yöntemlerden farkı politikanın değerini politikayı kontrol için kullanırken tahmin etmesidir. Politikadan bağımsız yöntemlerde bu iki fonksiyon ayrılmıştır. Davranışı oluşturmak için kullanılan *davranış politikası*, değerlendirilip iyileştirilen politika olan *tahmin edilen politikadan* bağımsızdır. Bu ayırımın avantajı davranış politikaları tüm olağan kararlarla devam ederken tahmin edilen miyopik politika gibi deterministik olabilir.

Bu yöntemde tahmin edilen politika öğrenilip iyileştirilirken davranış politikası takip edilir. Bu teknik tahmin edilen politikayla seçilebilecek tüm kararları seçerken davranış politikasında bu kararların seçilme olasılığının sıfırdan farklı olmasını gerektirir. Tüm olasılıkları keşfetmek için de bu davranış politikasının yumuşak olması gerekir.

Monte Carlo yöntemleri görüldüğü gibi DP'den iki yönüyle farklıdır. Birincisi örneklemler üzerinde işlem görmesi, böylelikle DP'de gerekli olan modele ihtiyaç yoktur. İkincisi de değer tahminlerinin diğer değer tahminlerine göre

güncellenmemesidir (bootstrapping). Gelecek bölümde Monte Carlo gibi örneklemlerle sistemi öğrenen fakat DP' de olduğu gibi değer fonksiyonlarını güncelleyen Zamansal Fark yöntemlere yer verilmiştir.

3.8 Zamansal Fark Yöntemleri

Zamansal Fark (TD) yöntemleri hem Monte Carlo gibi benzetim yoluyla (örneklemler üzerinde) elde edilen deneyimlerden öğrenmeyi sağlar, tüm modeli bilmemize gerek yoktur, hem de DP gibi değer tahminlerini yaparken önceki değerleri kullanarak güncelleme yapar (bootstrap). Yani o durum daha önce değerlendirildiyse bir daha o durumun değerini bulmaya gerek yoktur, ona göre güncelleme yapılır. Tüm modele, yani gidilecek olağan durumlara, onlardan sağlanacak getirilere ve geçiş olasılıklarına gerek duyulmaması DP' ye göre sağlanılmış en önemli kazançlardan biridir. Deneyimlerini en verimli şekilde kullanan bir algoritma yapısına sahip olurken çok hızlı bir yakınsama gösterir ve çok daha iyi tahminler üretirler (Sutton, 1988). TD kontrol metotları Monte Carlo yöntemlerinde olduğu gibi iki ana sınıfta incelenir: Politikaya bağlı (on-policy) ve politikadan bağımsız (off-policy) olmak üzere.

3.8.1 TD ile tahmin

Hem TD hem de Monte Carlo yöntemleri tahmin problemlerini çözmek için deneyimlerini kullanırlar. Deneyimlerle oluşan π politikası verildiğinde her iki yöntemde V^{π} değerlerini kullanarak V değer fonksiyonlarını güncellerler. Herhangi bir t anında son durum olmayan s_t durumu ziyaret edildiğinde her iki yöntemde daha önceki bu duruma ziyaretlerine dayanarak $V(s_t)$ değerlerini günceller. Monte Carlo (MC) yöntemleri böyle bir ziyaret oluncaya dek bekler ve daha sonra $V(s_t)$ değerlerini günceller. Monte Carlo yönteminde güncelleme

$$V(s_t) \leftarrow V(s_t) + \alpha[R_t - V(s_t)] \qquad \textbf{(3.15)}$$

şeklinde olur ki R_t t anından sonraki gerçek kazancı, α da sabit adım-büyüklüğü parametresini göstermektedir. Bu yönteme sabit-α MC denir. MC yöntemleri

$V(s_t)$ 'nin güncellemesi için epizod sonunu beklerken TD yöntemleri sadece bir sonraki periyodu bekler.

$t+1$ anında hemen bir hedef değer oluşur ve gözlemlenen r_{t+1} ve $V(s_{t+1})$ kullanılarak güncelleme yapılır. Robbins-Monro algoritması olarak bilinen bu güncelleme algoritması örneklemlerden elde edilen bir rassal değişkenin beklenen değerini tahmin etmeye yarayan eski bir algoritmadır (Robbins ve Monro, 1951):

$$V(s_t) \leftarrow V(s_t) + \alpha[r_{t+1} + \gamma V(s_{t+1}) - V(s_t)] \quad \textbf{(3.16)}$$

Dolayısıyla hedef değer MC'de R_t, TD' de $r_{t+1} + \gamma V(s_{t+1})$ 'dir. Çünkü TD güncelleme kısmını DP' de olduğu gibi var olan tahminlere dayanarak yapar (bootstrapping). MC'de hedef değer tahmini bir değerdir, çünkü beklenen değer bilinmez, gerçek beklenen değer yerine örneklem geliri kullanılır **(3.17).** DP'deki hedef değer beklenen değerden dolayı değil de $V^{\pi}(s_{t+1})$ değerinin bilinmemesinden dolayı tahmini değerdir **(3.18)**, bunun yerine o anki tahmini değer olan $V_t(s_{t+1})$ kullanılır.

$$V^{\pi}(s) = E_{\pi}\{R_t \mid s_t = s\} \quad \textbf{(3.17)}$$

$$= E_{\pi}\left\{\sum_{k=0}^{\infty} \gamma^k \, r_{t+k+1} \mid s_t = s\right\}$$

$$= E_{\pi}\left\{r_{t+1} + \sum_{k=0}^{\infty} \gamma^k \, r_{t+k+2} \mid s_t = s\right\}$$

$$= E_{\pi}\left\{r_{t+1} + \gamma V^{\pi}(s_{t+1}) \mid s_t = s\right\} \quad \textbf{(3.18)}$$

TD hedef değeri ise her iki nedenden dolayı (yani **(3.18)** deki gibi hem beklenen değer hem de V^{π} yerine o anki V_t değerini kullandığından dolayı) tahmini değerdir. Böylece TD yöntemleri MC ile DP' yi birleştirmiştir. Şekil 3.5' te TD yönteminin adımları yer almaktadır.

Hayali bir $V(s)$ ve π politikası belirle

Her epizod için aşağıdakileri tekrarla

İlk s durumunu belirle

Epizodun her adımı için aşağıdakileri tekrarla

$a \leftarrow$ s için π'dan gelen karar

a kararını al, r gelirini ve s' gelecek durumunu belirle

$V(s) \leftarrow V(s) + \alpha[r + \gamma V(s') - V(s)]$

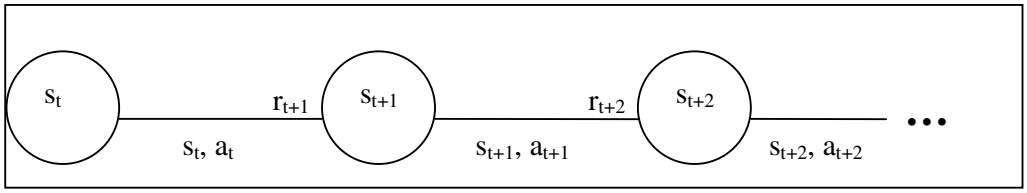
$s \leftarrow s'$

s don durum oluncaya kadar

Şekil 3.5 : V^{π} tahmini için tablo halinde TD (Sutton ve Barto, 1998).

3.8.2 SARSA: Politikaya bağlı (on-policy) TD kontrolü

Genelde kontrol problemi için genelleştirilmiş politika yineleme (GPI) kullanılır ama bu bölümde TD yöntemleri kullanılarak tahmin veya değerlendirme yapılmıştır. Monte Carlo yöntemlerinin kullanılmasıyla keşif ve önceki bilgileri kullanma (exploitation) kavramları ile politikaya bağlı (on-policy) ve politikadan bağımsız (off-policy) olmak üzere iki sınıf ortaya çıkmıştır. SARSA politikaya bağlı bir TD yöntemidir. SARSA algoritmasında ilk adım durum-değer (state-value) fonksiyonu yerine karar-değer (action-value) fonksiyonunun öğrenilmesidir. Özellikle politikaya bağlı yöntemler için o anki π politikası altında tüm s durum ve a kararları için $Q^{\pi}(s,a)$ karar-değer fonksiyonlarının tahmin edilmesi gerekmektedir. V^{π} değer fonksiyonu tahmin edilirken her durumun değeri sadece o durumdan diğer duruma geçişlerle ifade edilerek öğrenilirdi. Fakat şimdi geçişler durum-karar çiftinden diğer durum-karar çiftine şeklinde olmakta ve durum-karar çiftlerinin $Q^{\pi}(s,a)$ değerleri öğrenilmektedir.



Şekil 3.6 : Durum-karar çiftlerinin sırası (Sutton ve Barto, 1998).

t anında s_t durumunda a_t kararı alındığında sisteme r_t kadar bir getiri sağlanmakta ve $t+1$ anında s_{t+1} durumuna geçilmektedir. Yeni alınan kararla bir sonraki duruma geçilir ve bu son periyoda kadar bu şekilde devam eder. Böylelikle bir iterasyon gerçekleşmiş olur. İterasyon sayısı ne kadar fazla olursa o kadar çok durum incelenmiş olur ve $Q(s_t, a_t)$ değerlerine daha çabuk ulaşılır. $Q(s_t, a_t)$ karar-değer fonksiyonu Robbins-Monro algoritmasına göre güncellenmektedir **(3.19)**.

$$\begin{aligned} Q(s_t, a_t) &\leftarrow Q(s_t, a_t) + \alpha\left[r_t + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\right] \\ &\leftarrow (1-\alpha) Q(s_t, a_t) + \alpha\left[r_t + \gamma Q(s_{t+1}, a_{t+1})\right] \end{aligned} \qquad \textbf{(3.19)}$$

α adım büyüklüğü parametresini, γ da indirim oranını göstermektedir. Tüm politikaya bağlı yöntemlerde olduğu gibi Q^π değerleri sürekli π politikası altında tahmin edilir ve aynı zamanda π politikası Q^π' ya göre güncellenir. İterasyon bittikten sonra incelenen yani ziyaret edilen durumlara bakılır ve o durumun değerini maksimize eden karar π politikasına atanır:

$$\pi(s) = \arg\max_a \left[Q(s,a)\right] \qquad \textbf{(3.20)}$$

SARSA kontrol algoritmasının genel hali Şekil 3.7' de gösterilmektedir. SARSA algoritmasının yakınsama özellikleri Q'ya bağlı politikanın doğasına bağlıdır. Mesela $\varepsilon-$miyopik veya $\varepsilon-$yumuşak politikaları kullanılabilir. Tüm durum-karar çiftler sonsuz sayıda ziyaret edildiğinde, SARSA 100% olasılıkla optimal politikaya yakınsar ve limitte politika miyopik politikaya ($\varepsilon-$miyopik politikalarda $\varepsilon = 1/t$ ayarlanmasıyla düzenlenebilir) yakınsar. Algoritma şu şekilde ilerlemektedir: Keyfi bir π politikası belirledikten sonra $Q(s,a)$ fonksiyonunun başlangıç değerleri atanır. Daha sonra m iterasyon için algoritma çalıştırılır. Her s durumunda ε-miyopik' e göre fiyat kararı seçilir. ε-miyopik' e göre çoğu zaman değer fonksiyonunu maksimize eden fiyat kararı seçilir, diğer zamanlarda da olurlu fiyat karar kümesinden rassal bir karar seçilir. Her iterasyondan sonra politika güncellenir. π politikası her iterasyondan sonra güncellenebileceği gibi politika yinelemede olduğu gibi politika değerlendirme kısmı birçok iterasyondan oluşabilir ve politika daha sonra güncellenebilir. Bizim algoritmamızda politika her iterasyondan sonra güncellenmiştir (Şekil 3.7).

Adım 0: Keyfi π politikası belirle

Adım 1: $Q(s,a)$ fonksiyonunun başlangıç değerlerini ata

Adım 2: n iterasyon sayısını ve T periyot sayısını belirle

Adım 3: Her $i = 1, 2, ..., m$ için aşağıdaki adımları yap

Adım 3a: s ilk durumunu belirle

Adım 3b: Bir a kararı seç (ε-miyopik)

Adım 3c: Her $t = 1, 2, ..., T$ için aşağıdaki adımları gerçekleştir

Adım 3c_1: a kararını al, r getirisini ve bir sonraki s' durumunu incele

Adım 3c_2: Her s' durumu için π politikasından gelen a' kararlarını seç

Adım 3c_3: $Q(s,a) \leftarrow Q(s,a) + \alpha\left[r + \gamma Q(s',a') - Q(s,a)\right]$

Adım 3c_4: $s \leftarrow s'; a \leftarrow a'$

Adım 3c_5: $\pi(s) = \arg\max_a \left[Q(s,a)\right]$

Adım 3d: $i \le m$ ise Adım 3' e geri dön. Değilse Adım 4' e git.

Adım 4: π politikasını döndür.

Şekil 3.7 : SARSA: Politikaya bağlı TD kontrol algoritması (Sutton ve Barto, 1998)

Algoritma sonlandığında artık her durum için bir karar mevcuttur. Algoritma birçok iterasyondan oluştuğundan ve talep rassal olduğundan her t anında sistem farklı envanter düzeyinde yada aynı envanter düzeyinde bulunabilir. Aynı s_t envanter düzeyinde bulunuyorsa önceki iterasyonlarda t-1 anında hangi fiyat kararların verildiği önem kazanmaktadır. Çünkü t anında ona göre a_t fiyat kararı verilmiştir.

3.8.3 Q-öğrenme : politikadan bağımsız TD kontrolü

ÖÖ' nün en önemli buluşlardan biri $Q-$öğrenme (Q-learning) algoritması olarak bilinen (Watkins, 1989) politikadan bağımsız TD kontrol algoritmasıdır. En basit haliyle $Q-$öğrenme algoritmasında $Q-$faktörleri (durum-karar çiftleri) aşağıdaki gibi hesaplanır.

$$Q(s_t, a_t) = r_{t+1} + \gamma Q^*(s_{t+1}, a) \qquad \textbf{(3.21)}$$

$t+1$ anında s_{t+1} durumundaki en iyi durum-karar değeri $Q^*(s_{t+1}, a)$, bu durumun değerini enbüyükleyen karara göre elde edilir:

$$Q^*(s_{t+1}, a) = \max_a Q(s_{t+1}, a) \quad \textbf{(3.22)}$$

İterasyonlar ilerledikçe bu durum-karar fonksiyonları yine Robbins-Monro algoritmasına göre güncellenir:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t) \right] \quad \textbf{(3.23)}$$

Bu durumda öğrenilen karar değer fonksiyonu Q politikadan bağımsız takip edilen optimal karar-değer fonksiyonu Q^*'a yakınsar. Bu da algoritma analizini oldukça basitleştirir ve daha erken yakınsamayı sağlar. Politikanın hala hangi durum-karar çiftlerinin ziyaret edildiği ve güncellendiğine ilişkin etkisi devam etmektedir, fakat doğru bir yakınsama için tüm çiftlerin güncellenmesi devam etmelidir. Stokastik adım büyüklüğü parametrelerinin kullanılmasıyla Q_t'nin Q^*'a 100% olasılıkla yakınsadığı gösterilmiştir. $Q-$öğrenme algoritması Şekil 3.8'de verilmiştir.

Hayali $Q(s,a)$ değerleri ata

Her epizod için aşağıdaki adımları tekrarla

  İlk durum s'yi belirle

  Epizodun her değeri için aşağıdaki adımları tekrarla

    Q'dan türetilen ($\varepsilon-$miyopik gibi) politikayı kullanarak s için bir a kararı seç

    a kararını al, r ve s''yi gözlemle

    $Q(s,a) \leftarrow Q(s,a) + \alpha \left[r + \gamma \max_{a'} Q(s', a') - Q(s,a) \right]$

    $s \leftarrow s'$

  s son durum oluncaya dek

Şekil 3.8 : $Q-$öğrenme: Politikadan bağımsız TD kontrol algoritması (Sutton ve Barto, 1998).

3.8.4 *Q-P* öğrenme algoritması

Q-P öğrenme algoritması politika yinelemeye dayanan bir ÖÖ algoritmasıdır. Politika değerlendirme aşamasında *Q*-öğrenme algoritmasını kullanır. Her politika değerlendirme bir epizod olarak tanımlanmaktadır. Politika geliştirme aşamasında ise oluşan *Q*-faktör değerleri *P*-faktörlerine kopyalanır ve *Q*-faktörleri yok edilir. Sonra yeni iterasyonla aynı aşamalar devam eder. Ayrıntılı olarak algoritmanın adımları şu şekildedir:

Adım 1. *P*-faktörlerinin, $P(s,a)$ $s \in S$ ve $a \in A(s)$, ilk hayali değerlerini ata. İterasyon sayısı $E = 1$ olarak ata ve maksimum epizod sayısı $E_{\max}$'i ve her epizodda gerçekleştirilecek maksimum iterasyon sayısı $k_{\max}$' i belirle. Tüm durum-karar çiftlerine ilişkin ziyaret sayılarını, $V(s,a)$, 0 olarak ata.

Adım 2. *Politika değerlendirme*: Simülasyona başla. Tüm *Q*-faktörleri değerlerini 0 olarak ata. İçinde bulunduğumuz sistem durumu i olsun ve iterasyon sayısı $k = 1$ olarak ata.

Adım 2a. $a \in A(i)$ kararını $1/A(i)$ olasılıkla simüle et.

Adım 2b. Simülatörde gelinecek diğer durum j olsun. Ziyaret sayısı $V(i,a)$'ı 1 artır. $r(j \mid i,a)$, i durumundan j durumuna geçerken elde edilen geliri göstersin. Adım büyüklüğünü $\alpha = 1/V(i,a)$ olarak ata ve $Q(i,a)$ değerini güncelle:

$$Q(i,a) \leftarrow (1-\alpha)Q(i,a) + \alpha[r(j \mid i,a) + \lambda Q(j, \arg\max_{b \in A(j)} P(j,b))] \quad \textbf{(3.24)}$$

Adım 2c. k değerini 1 artır. $k < k_{\max}$ ise $i \leftarrow j$ olsun ve Adım 2a'ya dön, değilse Adım 3'e git.

Adım 3. *Politika iyileştirme:* Tüm *s* durum ve *a* kararları için

$$P(s,a) \leftarrow Q(s,a) \quad \textbf{(3.25)}$$

E'yi 1 artır. $E = E_{\max}$ ise Adım 4'e git, değilse Adım 2'ye git.

Adım 4. Her durum için optimal kararlara karar ver.

$$a^*(s) \in \arg\max_{a \in A(s)} Q(s,a) \quad \textbf{(3.26)}$$

Q-faktörler ile ilgili doğru tahminler yapılmayabilir. Bunun nedeni $k_{\max}$ sayısının sonlu bir sayı olmasıdır. $k_{\max}$ sayısını *E* ile birlikte artırırsak daha doğru sonuçlar verebilir. Bu durum politika değerlendirme performansını algoritma optimale yaklaşırken artıracaktır. Son olarak *a* kararı eşit olasılıkla değil de keşif stratejisi kullanılarak seçilebilir (Gosavi, 2003).

3.8.5 Aktör – kritik yöntemleri

TD yöntemlerinden Aktör-kritik yöntemleri değer fonksiyonunun politikadan bağımsızlığını açıkça gösteren ayrı bir yapıya sahiptir. Politika yapısı 'aktör' olarak bilinir, çünkü kararları seçmek için kullanılır. Tahmin edilen değer fonksiyonu ise 'kritik' olarak bilinir, çünkü aktör tarafından alınan kararların kritiğini yapar, değerlerini oluşturur. Öğrenme her zaman politikaya bağlıdır: kritik, aktör tarafından izlenilen politika her ne olursa olsun onu öğrenmelidir.

Aktör-kritik yöntemleri Ödüllü Öğrenme yöntemlerinin doğal bir uzantısıdır. Kritik tipik olarak bir durum-değer fonksiyonudur. Her karar seçildikten sonra, kritik yeni durumun kritiğini yapar, yani durum iyiye mi kötüye mi gidiyor, onu belirler. Bu değerlendirme TD hatası olarak adlandırılır:

$$\delta_t = r_{t+1} + \gamma V(s_{t+1}) - V(s_t) \quad \textbf{(3.27)}$$

V kritik tarafından uygulanan o anki değer fonksiyonunu gösterir. Bu TD hatası s_t durumundaki a_t kararını değerlendirmek için kullanılabilir. Eğer TD hatası pozitif ise, bundan sonraki kararların seçiminde bu kararın seçilme eğiliminin güçlendirilmesi gerektiğini, değilse, bu eğilimin azaltılması gerektiğini söyler. Kararların Gibbs softmax yöntemine göre **(3.28)**' deki gibi alındığını farz edelim:

$$\pi_t(s,a) = \Pr\{a_t = a \mid s_t = s\} = \frac{e^{p(s,a)}}{\sum_b e^{p(s,b)}} \quad \textbf{(3.28)}$$

$p(s,a)$, s durumu için a kararının seçilme eğilimini gösteren t anındaki değeri ifade eder. Kararın seçilme eğilimini güçlendirme ve zayıflatma bu parametrenin artırılıp azaltılmasıyla uygulanabilir:

$$p(s_t,a_t) \leftarrow p(s_t,a_t) + \beta\delta_t \tag{3.29}$$

β pozitif adım büyüklüğü parametresini gösteren bir katsayıdır. Çok önceki TD yöntemlerini kullanan ÖÖ algoritmalarının çoğu aktör kritik yöntemlerdir. Daha sonra durum değer fonksiyonlarını öğrenerek bu değerlerden politikaların belirlenmesi ortaya çıkmıştır (SARSA ve *Q*-öğrenme algoritmaları gibi). Fakat kararların seçiminde çok küçük hesaplama gerektirdiğinden avantajlı bir yöntemdir.

3.9 Yaklaşık Değer Yineleme Yöntemi

ÖÖ yöntemlerinden biri olan Yaklaşık Değer Yineleme (YDY) yöntemi, durum uzayının büyük olduğu stokastik eniyileme problemlerini çözmede kullanılan bir yöntemdir (Powell, 2007). Yöntemin temeli dinamik programlamada olduğu gibi Bellman Denklemine dayanır:

$$V_t(i) = \max_{a_t}\left(C(i,a_t) + \gamma\sum_{j\in S} p(j \mid i,a_t)V_{t+1}(j) \right) \tag{3.30}$$

YDY' de temel prensip, değer fonksiyonu $V_t(i)$'nin yerine yaklaşık değer fonksiyonu $\bar{V}_t(i)$'nin kullanılmasıdır. YDY'nin dinamik programlamadan diğer bir farkı ise yaklaşık değer fonksiyonu $\bar{V}_t(i)$'nin sondan başa değil de baştan sona doğru hesaplanmasıdır. YDY'de herhangi bir s_0 durumu ile başlanır ve belirli bir $w\in\Omega$ örnek patikası izlenir. Bu işlem tekrarlanarak devam eder. $n-1$. yinelenmeden sonra *t* anında *i* durumunda bulunmanın yaklaşık değeri olan $\bar{V}_t^{n-1}(i)$' yi elde etmiş oluruz. *n.* örnek patikası w^n'e ait en iyi karar dizisi $\left(a_0^*(s_0), a_1^*(s_1), ..., a_T^*(s_T)\right)$'yi belirlemek için $\bar{V}_t^{n-1}(i)$ yaklaşık değerleri kullanılır. Örneğin, *n.* yinelenmeye *t* anında s_0 durumunda başladığımızı varsayalım. a_t^* en iyi kararını bulabilmek için

$$a_t^* = \arg\max_{a\in A_t}\left(C(s_0,a) + \gamma E\left\{ \bar{V}_{t+1}^{n-1}(j) \mid s_0 \right\} \right) \tag{3.31}$$

denklemini çözmemiz gerekir. Dinamik programlamadaki en büyük zorluk **(3.31)**'deki beklenen değerin hesaplanmasıdır. Bir an için her $a \in A_t$ kararı için $p(j \mid i, a_t)$ olasılıklarını içeren tek adım geçiş matrislerini bildiğimizi varsayalım. Dolayısı ile **(3.31)** denklemini **(3.32)** şeklinde yazabiliriz.

$$a_t^* = \arg\max_{a \in A_t} \left(C(s_0, a) + \gamma \sum_{j \in S} P_t(j \mid s_0, a) \bar{V}_{t+1}^{n-1}(j) \right) \qquad \textbf{(3.32)}$$

$\bar{V}_t^n(i)$ yaklaşık değer fonksiyonunu durum uzayındaki tüm durumlar için hesaplamak neredeyse imkansız olduğu için (aslında **(3.31)** veya **(3.32)**'deki beklenen değeri de hesaplamak kolay değildir. Ancak bir an için hesaplayabildiğimizi düşünelim.) *n.* yinelenme için rastgele bir örnek patikası w^n belirleyebilir ve değer fonksiyonu $\bar{V}_t^n(i)$'yi sadece bu değerler için güncelleyebiliriz. Örnek patikasının rastgele belirlenmesi genellikle *Monte Carlo Benzetimi* olarak adlandırılır. Örnek patikası kavramı, YDY'nin temelini oluşturur. *n.* yinelemeden sonra **(3.31)**'deki beklenen değer, *n.* yinelemede ziyaret edilen durumların güncellenmiş değer fonksiyonu, $\bar{V}_t^n(i)$'yi kullanarak; ziyaret edilmemiş durumlar için ise, bir önceki yinelemedeki $\bar{V}_t^{n-1}(i)$ değer fonksiyonunu kullanarak hesaplayabiliriz. *n.* yinelemede ziyaret edilmiş olan herhangi bir i durumu için $\bar{V}_t^n(i)$ **(3.30)** yardımı ile hesaplanır. Artık tüm durumları kapsayan bir döngü kurmak zorunda kalınmamaktadır. Her yinelemede $T+1$ adet durum ziyaret edilmektedir. Ancak, her yinelemede sadece w^n örnek patikasındaki durumların ziyaret edilmesi çözülmesi gereken başka problemler yaratır. Bunlar,

1) **(3.31)** no'lu denklemdeki beklenen değeri hesaplamak için hala tek adım geçiş matrisi $P(a)$'nin kullanılmasını gerektirir. Bu da neredeyse, tüm durum uzayını göz önünde bulundurduğumuz durum kadar zor bir durumdur. Çünkü eğer karar değişkenleri olan a_t, k boyutlu bir vektör ise her i durumu için k tane tek adım geçiş matrisine ihtiyaç olacaktır.

2) Yukarıda bahsettiğimiz gibi sadece ziyaret ettiğimiz durumların değer fonksiyonlarını güncellemekteyiz. *n.* yinelemede ziyaret etmediğimiz durumların değer fonksiyonlarını da tahmin etmememiz gerekir.

3) Bazı durumları hiçbir zaman ziyaret etmemek mümkündür. Milyonlarca durumun olduğu bir problemde 1000 iterasyon gerçekleştirdiğimizde ziyaret edilmemiş birçok durum olacaktır. Ziyaret etmediğimiz bir durumun çok iyi sonuçlar verdiğini hiçbir zaman fark edemeyebiliriz.

Şimdi, tek adım geçiş matrisi $P(a)$'yi kullanmaya gerek olmaksızın değer fonksiyonunu nasıl güncelleyebileceğimiz problemini ele alalım. Bunun için birçok yaklaşım bulunmakla birlikte, $\bar{V}_t^n(i)$'ni hesaplamanın en basit yolu, $t+1$ anında ortaya çıkacak olan rastgele bilginin (örneğin talep) ne olabileceğini tahmin etmek amacıyla bir rastgele örnek patikaları kümesi türetmektir. Türetilen bu örnek patikalar kümesini $\hat{\Omega}_{t+1}$ olarak adlandıralım. $p_{t+1}(\hat{w})$ ise bu kümenin elemanı olan $\hat{w}_{t+1}$ örnek patikasının olasılığı olsun. Bu durumda *t* anında *i* durumunda olmanın değer fonksiyonunu yaklaşık olarak aşağıdaki gibidir:

$$\hat{v}_t^n(i) = \max_{x_t \in X_t} \left\{ C(i,x_t) + \gamma \sum_{\hat{w} \in \hat{\Omega}_{t+1}^n} p_{t+1}(\hat{w}) \bar{V}_{t+1}^{n-1}(j, x_t, w_t) \right\} \qquad \textbf{(3.33)}$$

(3.33)'de $p_{t+1}(\hat{w})$ olasılığını hesaplamak için $\hat{\Omega}_{t+1}^n$ örnek patika kümesinin tüm gerçeklenmelerini türetmek gerekmektedir. Türetilen bu tüm gerçeklenmeler kümesine "tablo formu" denir. Eğer *N* tane örnek patikası türetilmiş ve her bir örnek patikası birbirinden farklı ise, (ki bir çok durumda böyledir) $p_{t+1}(\hat{w})$ olasılığı, $1/N$ olacaktır.

$n-1$. yinelenmede elde edilen yaklaşık değer fonksiyonu değeri $\bar{V}_{t+1}^{n-1}(i)$ ile *n.* yinelemede elde edilen $\hat{v}_t^n(i)$ değeri arasındaki fark olan $\varepsilon_t^n(i) = \hat{v}_t^n(i) - \bar{V}_t^{n-1}(i)$'ye *Bellman hatası* denir. Burada, $\hat{v}_t^n(i)$, *n.* yinelemede ortalaması $\bar{V}_t^{n-1}(i)$ olarak tahmin edilmiş olan bir dağılımdan çekilen gözlem değeri olarak düşünülebilir. Ancak, $\bar{V}_{t+1}^{n-1}(j)$ değerleri yaklaşık değerler olduğundan $\hat{v}_t^n(i)$ gözlem değeri yanlı (biased) bir gözlem değeridir. $\hat{v}_t^n(i)$'yi hesaplamak için $\hat{\Omega}_{t+1}$'den rastgele bir patika örneklediğimiz için $\bar{V}_{t+1}^{n-1}(j)$ değeri kesin olsa bile $\hat{v}_t^n(i)$' nin tahmininde örneklemeden kaynaklanan bir hata olacaktır. Bu nedenle, $\bar{V}_t^n(i)$'nin, yani *i* durumunda bulunmanın değer fonksiyonunun *n.* yinelemedeki tahmini değeri

$$\bar{V}_t^n(i) = (1-\alpha_{n-1})\bar{V}_t^{n-1}(i) + \alpha_{n-1}\hat{v}_t^n \qquad \textbf{(3.34)}$$

şeklinde güncellenir. Yani $\bar{V}_t^n(i)$, $n-1$. iterasyondaki $\bar{V}_t^{n-1}(i)$ ile *n.* iterasyondan elde edilen $\hat{v}_t^n(i)$ değerlerinin doğrusal kombinasyonudur. **(3.34)**' te α_{n-1}'e adım büyüklüğü denir ve genellikle [0,1] arasında değerler alır. Burada yapmaya çalıştığımız şey, örneklem hatası içeren $\hat{v}_t^n(i)$ değerlerini kullanarak, bu gözlem değerlerinin geldiği dağılımın ortalaması olan $\bar{V}_t^{n-1}(i)$'nin yaklaşık değerini bulmaktır. **(3.34)**'te ifade edilen düzeltmeye ihtiyacımız olmasının nedeni, beklenen değere yaptığımız yaklaşım (approximation) nedeni ile $\hat{v}_t^n(i)$ değerinde rassallık oluşmasıdır. Bu rassallığın nedeni yukarıda da belirtildiği gibi örnek bir patikanın rastgele seçilmesidir. Eğer **(3.33)** numaralı denklemde verilen beklenen değeri kesin olarak hesaplayabilseydik, denklem

$$\hat{v}_t^n(i) = \bar{V}_t^n(i) = \max_{x_t}(C(i,x_t) + \gamma\sum_{j\in S} p(j \mid i,x_t)V_{t+1}^{n-1}(j)) \qquad \textbf{(3.35)}$$

halini alacak ve $\hat{v}_t^n(i)$'de örnekleme hatası olmayacak ve örnekleme hatasını düzeltmek için $\bar{V}_t^{n-1}(i)$ değer fonksiyonu değerini kullanmak zorunda kalmayacaktık. Yani, **(3.34)** numaralı denklemde $\alpha_{n-1} = 1$ olacaktı. Şekil 3.9, **(3.30)** numaralı denklemde verilen *Bellman Denklemindeki* beklenen değerin, **(3.33)**'deki gibi yaklaştırıldığı durumda YDY algoritmasını göstermektedir.

Şekil 3.9' da verilen algoritmanın ön değer atama adımı olan 0'ıncı adımında, durum uzayındaki tüm durumlara bir başlangıç değeri atanır ve ilk yinelemeye başlanır. Adım 1'de rastgele bir örnek patikası w^n türetilir. İkinci adımda, *t*'ninci dönemde gelen rassal bilgiyi (örneğin talep) temsil eden gerçeklenmeler olan $\hat{\Omega}_{t+1}^n$ türetilir ve $w^n = (w_0^n, ..., w_t^n, ..., w_T^n)$'nin bir bileşeni olan, *t*'ninci dönemdeki rastegele bilginin gerçeklenmesi olan w_t^n'in olasılığı, $\hat{\Omega}_{t+1}^n$'ye bakarak hesaplanır ve $\hat{v}_t^n(S_t)$ değer fonksiyonunu enbüyükleyen en iyi karar a_t^* bulunur. $S_t = i$ durumundayken, w_t^n rastgele bilgisi gerçeklendiğinde ve a_t kararı alındığında, $t+1$'inci dönemde içerisinde bulunacağımız durum, $p_{t+1}(\hat{w})$ olasılığı ile $S_{t+1} = j$ olacaktır.

Adım 0. Ön Değer Atama:

Adım 0a. Tüm s_t durumları için ilk $\bar{V}_t^0(S_t)$ değerlerini ata

Adım 0b. Herhangi bir S_0^1 durumu seç

Adım 0c. $n = 1$

Adım 1. Bir w^n örnek patikası seç

Adım 2. Her $t = 0,1,2,...,T$ için aşağıdaki işlemleri yap:

Adım 2a. t ile $t+1$ arasında gelen bilgiyi gösteren rastgele $\hat{\Omega}_{t+1}^n \subset \Omega$ gerçeklenmeler türet.

Adım 2b. $\hat{v}_t^n(S_t) = \max_{a_t \in A_t^n} \left\{ C(S_t, a_t) + \gamma \sum_{\hat{w} \in \hat{\Omega}_{t+1}^n} p_{t+1}(\hat{w}) \bar{V}_{t+1}^{n-1}(S_{t+1}) \right\}$ değer fonksiyonunu çöz ve S_t durumu için en iyi karar a_t^*'ı bul.

Burada $S_{t+1} = S^M(S_t^n, a_t, W_{t+1}(\hat{w}))$ sistem denklemi ile ifade edilebilir.

Adım 2c. $\bar{V}_t^{n-1}(S_t^n)$ değerini güncelle:

$$\bar{V}_t^n(S_t) = \begin{cases} (1-\alpha_{n-1})\bar{V}_t^{n-1}(S_t^n) + \alpha_{n-1}\hat{v}_t^n & S_t = S_t^n \\ \bar{V}_t^{n-1}(S_t) & \text{Diğer durumlarda} \end{cases}$$

Adım 2d. $S_{t+1}^n = S^M(S_t^n, a_t^n, W_{t+1}(w^n))$ yi hesapla

Adım 3. $n = n+1$. $n < N$ ise, Adım 1'e git.

Şekil 3.9 : Tablo Formu gösterimini kullanan YDY algoritması (Powell, 2007).

Adım 2'de ise, **(3.34)** numaralı denklemde gösterdiğimiz gibi $\bar{V}_t^n(S_t)$ değeri için düzeltme gerçekleştirilir. *n*. yinelemede optimal karar değişkenini bulmaya çalıştığımız S_t durumu için bu düzeltme **(3.33)** no'lu denklemde gösterildiği gibi, *n*. yinelemede başlangıç durumu S_0^1'den erişilemeyen diğer durumlar için ise bir önceki iterasyondaki $\bar{V}_t^{n-1}(S_t)$ değeri alınır ve yineleme sayısı N'den küçük ise Adım 1'e geri dönülür.

ÖÖ'de boyut problemlerin üstesinden gelmek için geliştirilen en önemli strateji, karar verildikten sonra fakat yeni durum bilgisi gelmeden önce, sistemin durumunu

gösteren karar-sonrası durum değişkeni (*post-decision state variable*) kullanılmasıdır.

3.9.1 Karar sonrası durum değişkeni

YDY'nin pratik uygulamalarında en zor adımlardan biri maks operatörü içindeki beklenen değeri hesaplamaktır. Çoğu DP ders kitaplarında tek adım geçiş matrisi veri olarak verilerek bu problemin üstesinden gelinmeye çalışılır. Fakat çoğu uygulamada tek adım geçiş matrisinin hesaplanması imkansızdır. Bu yüzden karar sonrası durum değişkeni kullanılmıştır.

Geçiş denklemi, $S_{t+1} = S^M(S_t, a_t, W_{t+1})$ (diğer sık kullanılan bir deyim ise sistem modelidir) karar sonrası durum değişkeni kullanımı ile artık iki kısımda gösterilebilir:

$$S_t^a = S^{M,a}(S_t, a_t) \tag{3.36}$$

$$S_{t+1} = S^{M,W}(S_t^a, W_{t+1}) \tag{3.37}$$

S_t karar verilmeden hemen önceki sistemin durumunu, **(3.36)** numaralı denklemdeki S_t^a ise karar verildikten sonraki durumu gösterir. $S_t^a = S^{M,a}(S_t, a_t)$'ye karar-sonrası durum değişkeni denir. Çözmeye çalıştığımız çoklu ürün stokastik eniyileme modelinde a_t karar değişkeni *t.* dönem fiyatları $a_t = \left(a_t^1, a_t^2, ..., a_t^m\right)$'den, W_{t+1} de talep bilgisinden oluşmaktadır. *t* anında S_t stok miktarı a_t kararı alındığında henüz talep gerçeklenmediği için yine S_t olarak kalacak, talep bilgisi W_{t+1} geldiğinde **(3.37)** $S_t - W_{t+1}$ şeklini alacaktır. Yani bu durumda yeni değişkenlerimiz

$$\begin{aligned} S_t^a &= S_t \\ S_{t+1} &= S_t - W_{t+1} \end{aligned} \tag{3.38}$$

şeklinde yazılabilir. Daha önce tanımladığımız gibi, $V_t(S_t)$, S_t durumunda bulunmanın değeri (karar vermeden hemen önce) ve $V_t^a(S_t^a)$ ise, karar verdikten hemen sonra S_t^a durumunda bulunmanın değeri olsun. $V_{t-1}^a(S_{t-1}^a)$'i $V_t(S_t)$'nin bir fonksiyonu olarak yazarsak,

$$V_{t-1}^{a}(S_{t-1}^{a}) = E\left[V_t(S_t) \mid S_{t-1}^{a}\right] \quad \textbf{(3.39)}$$

olur. Benzer şekilde, $V_t(S_t)$ 'yi de $V_t^a(S_t^a)$ 'nin bir fonksiyonu şeklinde yazarsak,

$$V_t(S_t) = \max_{a_t \in A_t} \left(C_t(S_t, a_t) + \gamma V_t^a(S_t^a)\right) \quad \textbf{(3.40)}$$

olur. Şimdi de **(3.38)**'e benzer biçimde, *t.* dönemdeki karar-sonrası değer fonksiyonu $V_t^a(S_t^a)$'i bir sonraki dönem karar öncesi değer fonksiyonu $V_{t+1}(S_{t+1})$ ile ilişkilendirirsek,

$$V_t^a(S_t^a) = E\left[V_{t+1}(S_{t+1}) \mid S_t^a\right] \quad \textbf{(3.41)}$$

yazabiliriz. Eğer **(3.41)** denklemini **(3.40)**'da yerine yazarsak, bildiğimiz standart Bellman denklemini elde ederiz:

$$V_t(S_t) = \max_{a_t \in A_t} \left(C_t(S_t, a_t) + \gamma E\left[V_{t+1}(S_{t+1}) \mid S_t^a\right]\right) \quad \textbf{(3.42)}$$

Diğer taraftan, eğer **(3.40)**'ı **(3.39)**'da yerine yazarsak;

$$V_{t-1}^{x}(S_{t-1}^{x}) = E\left[\max_{x_t \in X_t} \left(C_t(S_t, x_t) + \gamma V_t^x(S_t^x)\right) \mid S_{t-1}^{x}\right] \quad \textbf{(3.43)}$$

eniyileme denklemini karar-sonrası durum değişkeni cinsinden ifade etmiş oluruz. Dikkat edilirse **(3.43)**'te beklenen değer operatörü, maks operatörünün dışındadır. Yani beklenen değer içerisinde bulunan bir eniyileme problemi çözmemiz gerekmektedir. Bu sayısal hesaplama açısından önemli bir avantaj sağlar: **(3.40)** deterministik bir eniyileme problemidir, bazen çözümü zor olmasına rağmen, çözüm için önerilmiş birçok yöntem bulunmaktadır. **(3.39)** ve **(3.41)**' deki beklenen değer kolay görünmekle birlikte büyük boyutlu problemlerde hesaplanması çoğunlukla imkansızdır. ÖÖ' de, bu beklenen değerlerin hesaplanmasında yaklaşık yöntemler kullanılır. Bellman denkleminin karar-sonrası durum değişkeni cinsinden yazılması sayısal hesaplamalarda büyük bir avantaj sağlar ve oldukça yeni bir yaklaşımdır (Powell, 2007). Karar sonrası durum değişkeninin kullanıldığı algoritmada da, daha önce olduğu gibi algoritma yinelemeli olarak çalıştırılır. Varsayalım ki, *n*

yinelemede ve *t* anında, $S_{t-1}^{a,n}$ karar-sonrası durumundayız. Şimdi $t-1$ ile *t* arasında gelen bilginin gerçeklenmesi olan $W_t(w^n)$'i kullanarak bir sonraki karar-öncesi durum $S_t^{a,n} = S^{M,W}(S_{t-1}^a, W_t(w^n))$ şeklinde yazılabilir. Karar sonrası durum değişkeninin kullanıldığı algoritmanın adımları Şekil 3.10' da verilmiştir.

Adım 0. Ön Değer Atama

Adım 0a. Her *t* için ilk $\bar{V}_t^0$ değerlerini ata

Adım 0b. $n = 1$

Adım 0c. İlk S_0^1 değerini ata

Adım 1. Bir w^n örnek yolu seç

Adım 2. Her $t = 0,1,2,...,T$ için aşağıdaki işlemleri yap:

Adım 2a. $\hat{v}_t^n = \max_{a_t \in A_t^n} \left\{ C_t(S_t^n, a_t) + \gamma \bar{V}_t^{n-1}(S^{M,a}(S_t^n, a_t) \right\}$ değer fonksiyonunu çöz ve a_t kararlarını ver

Adım 2b.. Eğer t>0 ise, $\bar{V}_{t-1}^{n-1}(S_{t-1}^{a,n})$ değerini güncelle:

$$\bar{V}_{t-1}^n(S_{t-1}^{a,n}) = (1-\alpha_{n-1})\bar{V}_{t-1}^{n-1}(S_{t-1}^{a,n}) + \alpha_{n-1}\hat{v}_t^n$$

Adım 2c. Karar-sonrası durum değişkenini bul

$$S_t^{a,n} = S^{M,a}(S_t^n, a_t)$$

Ve bir sonraki karar-öncesi durum değişkenini hesapla

$$S_{t+1}^n = S^{M,W}(S_t^{a,n}, W_{t+1}(w^n))$$

Adım 3. $n = n+1. n < N$ ise, Adım 1'e git.

Adım 4. $(\bar{V}_t^n)_{t=0}^T$ değer fonksiyonlarını hesapla.

Şekil 3.10 : Karar-sonrası durum değişkeninin kullanıldığı ileri dinamik programlama (Powell, 2007).

Algoritmanın ön değer atama ve birinci adımları Şekil 3.9' da verilen karar öncesi durum değişkeninin kullanıldığı YDY algoritması ile aynıdır. Adım 2a, *t* anındaki karar öncesi durum S_t^n'deki değeri eniyileyen kararı karar-sonrası durum değişkeni ile ilişkilendirmekte, Adım 2b ise $\bar{V}_{t-1}^{n-1}(S_{t-1}^{a,n})$, $n-1$. yinelenmedeki karar-sonrası durumun değer fonksiyonunu güncellemektedir: a_{t-1}^n kararı, bizi S_{t-1}^n karar öncesi

durumundan, $S_{t-1}^{n,a}$ karar-sonrası durumuna götürür ve rassal $W_t(w^n)$ bilgisi ise, bizi $S_{t-1}^{n,a}$' den t anındaki karar-öncesi durum değişkeni S_t^n'ye götürür. Dolayısıyla, $\hat{v}_t^n$, S_t^n karar sonrası durumunda bulunmanın bir örneklemi olmasının yanı sıra, $\hat{v}_t^n$, $S_{t-1}^{n,a}$ karar sonrası durumunun bir örneklemi olarak da görülebilir. Bu nedenle karar-sonrası değer fonksiyonu 2b'de gösterildiği gibi

$$\overline{V}_{t-1}^n(S_{t-1}^{a,n}) = (1-\alpha_{n-1})\overline{V}_{t-1}^{n-1}(S_{t-1}^{a,n}) + \alpha_{n-1}\hat{v}_t^n \qquad \textbf{(3.44)}$$

güncellenebilir. Adım 2c'de ise S_t^n'den a_t'ye bağlı olarak gidilecek karar-sonrası durum $S_t^{n,a}$ ve bir sonraki dönemdeki karar öncesi durum S_{t+1}^n belirlenir, bir sonraki yinelemeye gidilir ve son yinelemeden sonra değer fonksiyonunun değeri hesaplanır.

3.10 Fonksiyon Yaklaşımı Teknikleri

Klasik dinamik programlama değer fonksiyonunu kesin bir gösterimle ifade eder. Bütün $s \in S$ durumları için s durumunda bulunmanın değerini veren v_s parametre değerlerini tahmin etmemiz gerekir. İleri doğru dinamik programlama geriye doğru dinamik programlamanın gerektirdiği tüm durumları inceleme kısmını ortadan kaldırabilir fakat durum uzayındaki klasik boyutun laneti problemini çözemez. İleri dinamik programlama gerçekten ziyaret ettiğimiz durumlara odaklanır, fakat bu da ziyaret edebileceğimiz durumların değerlerinin bilinmesini gerektirir.

ÖÖ' de bütün büyük ölçekli problemler değer fonksiyonlarının daha az sayıda parametreyle nasıl tahmin edebileceğinin kararı üzerine odaklanırlar. Geriye doğru dinamik programlamada her durumda bir parametre vardır ve çok sayıdaki bu durumları incelemekten kaçınırız. İleri dinamik programlamada ise her şey Monte Carlo örneklemine dayanır ve asıl sorun istatistiksel hatadır. Dolayısıyla daha az sayıda parametreyle ifade edilen fonksiyonları tahmin etmek oldukça basittir. Uygulamada ÖÖ her zaman problemin yapısını anlamayı gerektirir. Bunu da kolaylaştıran bazı genel stratejiler vardır.

ÖÖ büyük durum uzayında karşılaşılan sorunların üstesinden gelmek için bir fikir sunmaktadır. Uygulamada geriye doğru dinamik programlamada yapılan gidilebilecek tüm durumları hesaplama problemi sadece bazı durumların değerlerini tahmin etmeye yönelik bir istatistiksel problem haline getirilmiştir. Sadece ziyaret

ettiğimiz durumun değerini bilmemiz yeterli değildir, gidilebilecek durumların değerlerini tahmin etmemiz gerekir. Gidilebilecek durumların sayısı tabii ki dinamik programlamadaki tüm durumların sayısı kadar olmasa da büyük bir sayıdır.

Değer fonksiyonlarının yaklaşık değerlerini hesaplayabilmek için birçok istatistiksel teknikler vardır. Bunların en popüler olanları kümeleme (aggregation), interpolasyon (interpolation) gibi yöntemlerle ya da regresyon modellerini açıklayan bazı temel fonksiyonlarla bu değerleri hesaplamaktır.

3.10.1 Kümeleme ile fonksiyon yaklaşımı

Kümeleme tekniği değer fonksiyonlarının yaklaşık değerlerini hesaplamada belki de çoğunlukla kullanılan tekniklerden birincisidir. Kümeleme farklı durumları bir araya getirmektir. Bir araya getirmekle birlikte sistemde daha az sayıda durum olacaktır. Eğer tüm durumları ayrı ayrı inceleyip hafızada tutacaksak bu zaten tablo formu yaklaşımı olacaktır. Durumlar bir araya geldiğinde genellikle Markov özelliğini kaybederler ve Dinamik Programlama çözümü daha fazla optimal olmayı garantileyemeyebilir ve sonuç olarak yöntem sezgisele dönüşebilir. Fakat deneysel sonuçlar göstermiştir ki kümelenmiş bir durum uzayı üzerinde DP yaklaşımı, kendiside bir sezgisel gibi davranmasına rağmen diğer sezgisel yöntemlerden daha iyi olabilmektedir. Bazı durumlarda Markov özelliğini kaybetmeden durumları bir araya getirmek mümkün olabilir, fakat optimal sonuç garanti değildir. Buna rağmen kümeleme boyut probleminden kurtulmak için en güvenilir yaklaşımlardan biridir (Gosavi, 2003).

Durumları bir araya getirmekte en genel kural, benzer özellikte olanları bir araya getirmektir. “Benzer” ifadesinin ne açıdan olduğu tabii ki önemlidir. Belli bir aralıktaki değerleri ya da önemsiz sayabileceğimiz bazı durumları bir araya getirebiliriz. Bizim incelediğimiz sistemde benzer özellik kavramını ürünlerin envanter düzeyleri oluşturmaktadır. Buna göre 50’lik 100’lük kümeler oluşturduk. Yani 50’lik kümeleme yapıyorsak 0-50 adet envantere sahip olanları bir kümeye, 50-100 adete sahip olanları bir kümeye, 100-150 adete sahip olanları diğer bir kümeye aldık. 100’lük kümeleme yapıyorsak da 0-100 adete sahip olanlar bir kümeye, 100-200 adete sahip olanlar ise başka bir kümeye alınmıştır.

Farz edelim ki bir kümeleme fonksiyonu ailesine sahibiz ve $\xi =$ kümeleme düzeyine ilişkin indis kümesini; $s^{(g)} = G^g(s)$, s durumunun g. kümeleme düzeyini ifade etsinler. G^g daha küçük durum uzayı $S^{(g)}$' yi oluşturan S durum uzayında hareket eden bir kümeleme fonksiyonudur:

$G^g : S \to S^{(g)},\ g \in \xi$.

Yapay zeka (AI) camiasında kümeleme bir durum soyutlama şekli olarak kabul edilir. $n.$ iterasyonda t anında s_t^n durumunu ziyaret edersek, güncelleme denkleminin tahmini

$$\bar{V}_t^{(g,n)}(s) = \begin{cases} (1-\alpha_{n-1})\bar{V}_t^{(g,n-1)}(s) + \alpha_{n-1}\hat{v}_t^n, & \text{eğer } G(s_t^n) = s; \\ \bar{V}_t^{(g,n)}(s^{(g)}), & \text{diğer durumlarda} \end{cases} \quad \textbf{(3.45)}$$

şeklinde olur. $\bar{V}_t^{(g,n)}(s)$, $s^{(g)} = G^g(s)$ durumunda olmanın tahmini değeridir. Bu yapı her durumun tanımlanmış bir kümeye ait oluncaya dek devam eden genel bir kümeleme fikrini kullanır (Powell, 2010).

Kümeleme büyük durum uzayı problemlerini çözmekte sıklıkla kullanılan bir yöntemdir (Rogers ve diğ., 1991). Kısıtlayıcı yönü ise doğru kümeleme düzeyine karar verirken iyi bir cevabının bulunmamasıdır. Cevabı algoritmayı kaç iterasyon çalıştırdığımıza bağlıdır. Bertsekas ve diğ. (1989) ve Luus (2000) ise tek kümeleme düzeyine göre daha esnek olan bir strateji geliştirmişler ve farklı kümeleme düzeyleri oluşturmuşlardır. Buna göre değer fonksiyonu tahminleri farklı düzeylerdeki küme değerlerinin ağırlıklı ortalaması şeklinde yapılmıştır:

$$\bar{v}_s^n = \sum_{g \in \xi} w_s^{(g)} \bar{v}_s^{(g)} \quad \textbf{(3.46)}$$

$w_s^{(g)}$, $g.$ küme düzeyinde s durumunda bulunmanın değer tahminini yaparken uygulanan ağırlığı ifade eder. Bu model bir durumu hangi sıklıkla ziyaret ettiğimize dayanarak ağırlıkları ayarlar. Bu demek değildir ki birçok ağırlık hesaplanacaktır. Durum uzayı büyük olduğunda ve kümeleme kullanılmadığında zaten çoğu durum

incelenmeyecektir, ama çok sayıda kümeleme düzeyi kullanıldığında mutlaka bu kümeler incelenecektir.

$w_s^{(g)}$ ağırlıklarını tahmin etmek için birçok strateji vardır. Bir tanesi etkin olduğu ispatlanmış $\bar{v}_s^{(g)}$'nin varyansıyla orantılı ağırlıkları kullanmaktır (Powell, 2010). Bizde buradan yola çıkarak bir strateji geliştirdik. Durum uzayı çok büyük olduğunda 50'lik kümeleme bile yapsak bu kümelere iterasyonlar sırasında çok gelinmeyecektir. 100'lük ya da daha farklı kümelemeler yapılırsa da kümelerin varyansı artacaktır. Bu nedenle farklı kümeleme düzeylerini, varyanslarıyla doğru orantılı, ağırlıklı olarak bir araya getirerek kümelerin ve fonksiyonların değerlerini bulduk. "Karışık Kümeleme (mixed aggregation)" olarak adlandırılan bu tekniğin detayları şu şekildedir:

(g) kümeleme indeksi düzeyini göstersin. 50'lik kümeleme (g_1) ile, 100'lük kümeleme (g_2) ile ifade edilirse, $S_t^{(g)}$ durumunun karışık kümeleme değeri **(3.46)**' ya göre şöyle ifade edilmiştir:

$$Q(S_t^{(g)}) = w^{(g_1)} Q(S_t^{(g_1)}) + w^{(g_2)} Q(S_t^{(g_2)})$$

$w^{(g_i)}$, i. kümeleme düzeyinin $S_t^{(g_i)}$ durumuna ait, kümelerin varyanslarıyla $(MSE^{(g_i)})$ doğru orantılı, ağırlık katsayılarını ifade etmektedir **(3.47)**.

$$w^{(g_1)} = \frac{MSE^{(g_1)}}{MSE^{(g_1)} + MSE^{(g_2)}} \text{ ve } w^{(g_2)} = \frac{MSE^{(g_2)}}{MSE^{(g_1)} + MSE^{(g_2)}} \qquad \textbf{(3.47)}$$

Her durumun yeni karışık kümeleme değerleri bulunduktan sonra varyansları tekrar güncellenir. *MSE* kullandığımız algoritmaları değerlendirirken bir performans ölçütü olarak kullanılmaktadır. İterasyon sayısı fazla olduğunda *MSE* hesaplama zorlaşabilir, bu nedenle her iterasyonda adım adım bu değer güncellenebilir. S_t durumunun bir başlangıç değeri varsa, $Q^{n-1}(S_t)$, ve bu durumun tahmin edilen yeni değeri $\hat{v}_n(S_t)$ ise *MSE* **(3.48)**'deki gibi güncellenebilmektedir.

$$MSE^n(S_t) = (1-\alpha_{n-1}) MSE^{n-1}(S_t) + \alpha_{n-1}(Q^{n-1}(S_t) - \hat{v}_n(S_t))^2 \qquad \textbf{(3.48)}$$

α_{n-1}, adım büyüklüğü parametresini ifade etmektedir. Biz bu parametrenin tahmini için McClain formülünü kullandık **(3.49)**:

$$\alpha_n = \frac{\alpha_{n-1}}{1+\alpha_{n-1}-\bar{\alpha}} \qquad \textbf{(3.49)}$$

$\bar{\alpha}$ belirtilen bir katsayıyı ifade etmektedir. Makine öğrenmesi camiasında da küme düzeyleri ve durum belirlemeye yönelik birçok teknik önerilmektedir. Bu tekniklere Hastie ve diğ. (2001)'nin çalışmalarından erişilebilir.

3.10.2 Fonksiyon uydurma ile fonksiyon yaklaşımı

Büyük durum uzayları ile uğraşmanın diğer popüler bir yolu da regresyon veya sinir ağları gibi fonksiyon uydurma (function fitting) yöntemlerini kullanmaktır.

Şimdiye kadar olan bölümlerde değer fonksiyonlarının tablo formu gösterimleri üzerinde odaklanıldı, yani eğer belirli bir s durumunda isek, s durumunda bulunmanın tahmini değeri olan yaklaşık $\bar{v}(s)$ değerleri hesaplanıldı. Kümeleme tekniği de bir anlamda tablo formu şeklindedir, sadece daha basit bir tablo formu oluşturmuş oluruz. Bu yöntemin avantajı durum değişkeni için özel yapılar oluşturulmasına ihtiyaç duyulmamasıdır, dezavantajı ise bu avantajı pek sağlayamamasıdır, çünkü klasik doğrusal regresyon denkleminde yer alan θ parametrelerinin tahmini zordur. Bir doğrusal regresyon denkleminde $(x_i)_{i\in I}$ deneyler kümesini kullanan *y* değişkeninin tahmini **(3.50)**' deki gibidir.

$$y = \theta_0 + \sum_{i=1}^{I} \theta_i x_i + \varepsilon \qquad \textbf{(3.50)}$$

ÖÖ dilinde x_i bağımsız değişkenleri, potansiyel büyük durum uzayını daha küçük özelliklere (features) dönüştüren, temel fonksiyonlar (basis functions) ile oluşturulur. Böylece x_i bağımsız değişkeni yerine $f \in F$ özelliğini göstermek üzere $\phi_f(S)$ temel fonksiyonu kullanılacaktır. $\phi_f(S)$ bir malın fiyatı, stoklardaki ürün miktarı gibi bir değişkeni göstermektedir. Bu özelliklerin sayısı 10, 20 tane olacağı gibi binlerce de olabilir. Genel haliyle değer fonksiyonu **(3.51)**' deki gibi ifade edilir.

$$\bar{V}(S \mid \theta) = \sum_{f \in F} \theta_f \, \phi_f(S) \qquad \textbf{(3.51)}$$

Bir lineer regresyon denkleminde θ parametreleri şu şekilde tahmin edilir. y^n tahmin edilecek bağlı değişkenimizin $n.$ gözlemini göstersin. $n.$ gözlemdeki bağımsız değişkenler de $(x_1^n, x_2^n, ..., x_I^n)$ şeklinde gösterilsin. Amacımız **(3.52)** denklemini çözen θ parametrelerini tahmin etmektir.

$$\min_{\theta} \sum_{m=1}^{n} \left(y^m - (\theta_0 + \sum_{i=1}^{I} \theta_i x_i^m) \right)^2 \qquad \textbf{(3.52)}$$

$\bar{\theta}^n$ bu problemin optimal çözümünü göstersin. $x_0 = 1$ dersek, $x^n = \begin{pmatrix} x_0^n \\ x_1^n \\ ... \\ x_I^n \end{pmatrix}$ şeklinde $I+1$ boyutlu bir kolon vektörü olsun. θ parametreleri gösteren bir kolon vektörü olarak ifade edilirse, model

$$y = \theta^T x + \varepsilon$$

olur. $(\varepsilon^1, \varepsilon^2, ..., \varepsilon^n)$ bağımsız ve aynı şekilde dağılmıştır. $\bar{y}^n = (\bar{\theta})^T x^n$, y^{n+1}'in tahmini değerini göstersin. O zaman tahmin hatası $\hat{\varepsilon}^n = y^n - (\bar{\theta})^T x^n$ olur. Amaç bu hatayı minimize eden θ parametrelerini bulmaktır **(3.53)**.

$$\min_{\theta} \sum_{m=1}^{n} (y^m - \theta^T x^m)^2 \qquad \textbf{(3.53)}$$

X^n, $n \times (I+1)$ boyutlu bir bağımsız değişkenler matrisini, Y^n de $n \times 1$ boyutlu bağlı değişkenler matrisini göstersin.

$$X^n = \begin{bmatrix} x_0^1 & x_1^1 \ldots x_I^1 \\ x_0^2 & x_1^2 \ldots x_I^2 \\ \ldots & \ldots\ldots\ldots \\ x_0^n & x_1^n \ldots x_I^n \end{bmatrix} \qquad Y^n = \begin{bmatrix} y^1 \\ y^2 \\ \ldots \\ y^n \end{bmatrix}$$

Gerekli işlemler yapıldığında optimal $\bar{\theta}$ parametre vektörü **(3.54)** bulunur.

$$\bar{\theta} = \left[(X^n)^T X^n \right]^{-1} (X^n)^T Y^n \qquad \textbf{(3.54)}$$

İstatistik camiasında **(3.53)**' deki gibi bir statik eniyileme problemini çözmek için **(3.54)** çoğunlukla kullanılan ortak bir yaklaşımdır. ÖÖ algoritmasını uygularken her iterasyonda matrislere yeni gözlemler ekleneceğinden problem tekrarlanan bir yapıya sahiptir ve bu şekilde ele alınmalıdır.

3.10.2.1 Regresyon modelleri için tekrarlanan yöntemler

Değer fonksiyonlarını tahmin ederken regresyon modellerinin tahmini aynı istatistiksel konulara ve araçlara sahiptir. Dinamik programlamadaki tek fark verilerin algoritma içinde üretilmesidir. Bu da her iterasyonda regresyon modelinin güncellenmesini gerektirir. Geleneksel yöntemler dinamik programlama kapsamında genellikle çok yavaş çalışırlar. Bunun için tekrarlanan yöntemleri kullanmak daha kullanışlıdır. ÖÖ' de kullanılan en basit tekrarlanan (recursive) yöntemlerden biri stokastik eğim algoritmasıdır (stochastic gradient algorithm).

Bir s durumunda bulunmanın tahmini değerinin güncellenme denklemini

$$\bar{v}_s^n = \bar{v}_s^{n-1} - \alpha_{n-1}\left[\bar{v}_s^{n-1} - \hat{v}_s^n \right] \qquad \textbf{(3.55)}$$

şeklinde vermiştik. Bu güncelleme algoritma içinde aşağıdaki problemi çözmek için gerekli olan bir adımdır:

$$\min_V E\frac{1}{2}(V - \hat{v})^2 . \qquad \textbf{(3.56)}$$

$\hat{v}$, $V(s)$'in örneklem tahminini göstermektedir. Değer fonksiyonunun parametrelerini gösterdiğimizde, $\bar{V}_s(\theta)$'yi gösterebileceğimiz bir fonksiyon yaratmış oluruz. Böylelikle θ değerlerini bulabiliriz:

$$\min_{\theta} E\frac{1}{2}(\bar{V}_s(\theta)-\hat{v})^2. \qquad \textbf{(3.57)}$$

Standart stokastik eğim algoritmasını uygulayarak, güncelleme denklemini **(3.58)** elde ederiz.

$$\bar{\theta}^n = \bar{\theta}^{n-1} - \alpha_{n-1}(\bar{V}_s(\theta^{n-1})-\hat{v}(w^n))\nabla_\theta \bar{V}_s(\theta^n) \qquad \textbf{(3.58)}$$

$\bar{V}_s(\theta^n) = \sum_{f\in F}\theta_f\phi_f(s)$ olduğundan, θ'ya göre eğim aşağıdaki gibi ifade edilir:

$$\nabla_\theta \bar{V}_s(\theta^n) = \begin{bmatrix} \frac{\partial \bar{V}_s(\theta^n)}{\partial \theta_1} \\ \frac{\partial \bar{V}_s(\theta^n)}{\partial \theta_2} \\ \cdots\cdots\cdots \\ \frac{\partial \bar{V}_s(\theta^n)}{\partial \theta_F} \end{bmatrix} = \begin{bmatrix} \phi_1(s^n) \\ \phi_2(s^n) \\ \cdots\cdots \\ \phi_F(s^n) \end{bmatrix} = \Phi(s^n) \qquad \textbf{(3.59)}$$

Böylece **(3.58)**, **(3.60)** olarak güncellenebilir.

$$\begin{aligned} \bar{\theta}^n &= \bar{\theta}^{n-1} - \alpha_{n-1}(\bar{V}_s(\theta^{n-1})-\hat{v}(w^n))\Phi(s^n) \\ &= \bar{\theta}^{n-1} - \alpha_{n-1}(\bar{V}_s(\theta^{n-1})-\hat{v}(w^n))\begin{bmatrix} \phi_1(s^n) \\ \phi_2(s^n) \\ \cdots\cdots \\ \phi_F(s^n) \end{bmatrix} \end{aligned} \qquad \textbf{(3.60)}$$

Stokastik eğim algoritması parametre vektörü için başlangıç bir $\bar{\theta}^0$ gerektirir, bu da genellikle $\bar{\theta}^0 = 0$ olarak alınır.

3.10.2.2 Fonksiyon uydurmanın zorlukları

Fonksiyon uydurma ile ilgili bazı zorluklar vardır:

1. Yukarıda da değinildiği gibi her iterasyonda yeni bilgiler ekleneceğinden parametrelerin katsayıları değişecektir. Bunun için tekrarlamalı yöntemler kullanılmalıdır. Fakat bu yöntemin çalışması için tüm verilerin aynı fonksiyonlarla bağlantısı olmalıdır. Ancak, ÖÖ'de değer fonksiyonu ($Q(s,a)$ fonksiyonu) güncellenirken aynı fonksiyonlar kullanılmayabilir.
2. Tablo formu yaklaşımında Q faktörü değiştiğinde, bu değişim sadece bu Q faktörüne ilişkin yapılır. Fakat fonksiyon yaklaşımı ile yapılan bu değişim diğer faktörleri de etkileyecektir. Ayrıca bu değişimin iyi yönde olmasını garantileyemeyiz.
3. Değer fonksiyonunun şeklini hiçbir zaman önceden bilemeyiz. Doğrusal değer fonksiyonları nöronlarla tahmin edilebilir ama doğrusal olmayan durumlarda nöronlar iyi çalışmaz ve bu durumda doğrusal olmayan sinir ağları (backpropagation) önerilir. Fakat doğrusal olmayan sinir ağları da fonksiyonun modeline ihtiyaç duymadığından yerel optimal noktalarda takılabilir.
4. Fonksiyon yaklaşımı kullanıldığında matematiksel olarak algoritmanın yakınsamasını göstermek zordur. Bu yüzden pratikte kullanılmasa da ÖÖ davranışını anlamak için bazı derinlemesine analizler yapılmalıdır (Bertsekas ve Tsitsiklis, 1996).

3.10.3 İnterpolasyon ile fonksiyon yaklaşımı

İnterpolasyon tekniği sayısal matematik ve hızla gelişen veri madenciliği alanında sıklıkla kullanılan yöntemlerden biridir. Fonksiyon tahmin etmede güvenilir bir teknik olarak bilinir. ÖÖ'de fonksiyon yaklaşımı için etkin biçimde kullanılabilir.

Temel fikir temsili Q-faktörleri ya da V fonksiyon değerlerini tutup, diğer tüm Q faktörleri ya da V fonksiyon değerlerinin bu temsili değerlere göre interpolasyon ile karar vermektir. İnterpolasyon için kullanılan yöntemlerden biri en yakın k-komşu (k-nearest neighbors) yöntemidir.

Güncelleme yapılacak fonksiyon değeri sorgu (query) noktası olarak adlandırılır. Bu sorgu noktasının etrafında birçok temsili olarak atandığımız fonksiyon değerleri mevcuttur. Sorgu noktasına en yakın uzaklıkta bulunan k-komşu değerin ya ortalaması alınarak ya da regresyon ile bu sorgu noktasının değeri bulunur. Uzaklıklar hesaplanırken öklit ($d_{öklid}$) ya da manhattan uzaklık (d_{mnhttn}) kullanılabilir:

$$d_{öklit} = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

$$d_{mnhttn} = |x_1 - x_2| + |y_1 - y_2|$$

3.10.4 SHAPE algoritması

Sürekli fonksiyonların tahmini en basit olarak yaklaşık bir başlangıç değerle başlar ve gelişerek devam eder. Başlangıç değeri iyi bir çözümle başlarsa doğal olarak daha iyi sonuçlar elde edilecektir. Stokastik eğim algoritmasını kullanarak eniyileme problemini çözdüğümüzde şöyle bir sorun karşımıza çıkabilir: Karar değişkenleri ve eğim birimleri birbiriyle uyuşmayabilir, bu yüzden adım büyüklüğü parametresinin seçimi büyük önem taşımaktadır.

Bir dinamik programlama problemini yaklaşık olarak çözmemiz gerektiğinde ne olduğunu düşünelim. S_t, t anındaki envanter düzeyini, a_t, t anındaki fiyat kararını gösteren iki değer olsun. Karar sonrası durum değişkeninin (S_{t-1}^a) kullanıldığı farz edilirse, örnek patika w^n seçildiğinde karar öncesi değişkenimiz $S_t^n = S^{M,W}(S_{t-1}^a, W_t(w^n))$ olur ve bulunduğumuz durumun değerini enbüyükleyen fiyat kararı bulunmaya çalışılır **(3.61)**:

$$\tilde{V}_t^n(S_t^n) = \max_{a_t \in A_t}(C(S_t^n, a_t) + \bar{V}_t^{n-1}(S^{M,a}(S_t^n, a_t))). \qquad \textbf{(3.61)}$$

SHAPE algoritmasına göre $\bar{V}$ yaklaşık fonksiyon değeri **(3.62)**' daki gibi güncellenmektedir.

$$\bar{V}_{t-1}^n(S_{t-1}^a) = \bar{V}_{t-1}^{n-1}(S_{t-1}^a) + \alpha_{n-1}(\hat{v}_t^n - \nabla_S \bar{V}_{t-1}^{n-1}(S_{t-1}^a))S_{t-1}^a \qquad \textbf{(3.62)}$$

$\hat{v}_t^n$, $\tilde{V}_t^n(S_t^n)$'nin türevini ifade etmektedir yani S_t' deki bir birim değişimin fonksiyon üzerindeki etkisini göstermektedir. Bu türevi elde etmenin birçok farklı yolu olmakla birlikte, basitçe şu şekilde hesaplanabilir:

$$\hat{v}_t^n = \tilde{V}_t^n(S_t^n + 1) - \tilde{V}_t^n(S_t^n). \quad \textbf{(3.63)}$$

SHAPE algoritmasının adımları Şekil 3.11'de gösterilmiştir. Bu algoritmaya göre kararlar **(3.64)**' deki gibi alınmaktadır.

$$a_t^n = \arg\max_{a_t \in A_t}(C(S_t^n, a_t) + \bar{V}_t^{n-1}(S^{M,a}(S_t^n, a_t))). \quad \textbf{(3.64)}$$

Oysaki stokastik eğim algoritmasında kararlar aşağıdaki gibi alınmakta ve bu da birimler arasındaki uyuşmazlığa neden olmaktadır.

$$a^n = a^{n-1} - \alpha_{n-1}\nabla_a \tilde{V}(a^{n-1}, W(w^n)). \quad \textbf{(3.65)}$$

SHAPE algoritması ile stokastik eğim yöntemindeki birim uyuşmazlığı ortadan kalkmıştır. Yaklaşık değer fonksiyonları artık iteratif olarak değişen bir fonksiyona uydurulduğu için daha kısa sürede güncellenmekte ve bu da büyük bir kazanç sağlamaktadır. İleriki bölümlerde algoritma sonuçlarına yer verilmiştir.

Adım 0. Başlangıç değerlerini ata:

Adım 0a. Başlangıç $\overline{V}_t^0$, $t \in T$ değerlerini ata

Adım 0b. $n = 1$

Adım 0c. Başlangıç S_0^1 değerini ata.

Adım 1. Örnek patika w^n'i seç

Adım 2. Her $t = 0,1,...,T$ için aşağıdaki adımları gerçekleştir

Adım 2a. Aşağıdaki denklemi çöz

$$\tilde{V}_t^n(S_t^n) = \max_{a_t \in A_t}(C(S_t^n, a_t) + \overline{V}_t^{n-1}(S^{M,a}(S_t^n, a_t))).$$

ve a_t^n yukarıdaki modeli çözen a_t'nin değerini göstersin.

Adım 2b. Aşağıdaki türevi hesapla

$$\hat{v}_t^n = \tilde{V}_t^n(S_t^n + 1) - \tilde{V}_t^n(S_t^n)$$

Adım 2c. Eğer $t > 0$ ise, yaklaşık değer fonksiyonunu güncelle

$$\overline{V}_{t-1}^n(S_{t-1}^x) = \overline{V}_{t-1}^{n-1}(S_{t-1}^x) + \alpha_{n-1}(\hat{v}_t^n - \nabla_S \overline{V}_{t-1}^{n-1}(S_{t-1}^x))S_{t-1}^x$$

Adım 2d. Durumları güncelle:

$$S_t^{a,n} = S^{M,a}(S_t^n, a_t^n)$$
$$S_{t+1}^n = S^{M,W}(S_t^{a,n}, W_{t+1}(w^n))$$

Adım 3. $n = n + 1$. Eğer $n \le N$ ise adım 1' e git.

Adım 4. Değer fonksiyonu $(\overline{V}_t^N)_{t=1}^T$'i döndür.

Şekil 3.11 : SHAPE Algoritması (Powell, 2007).

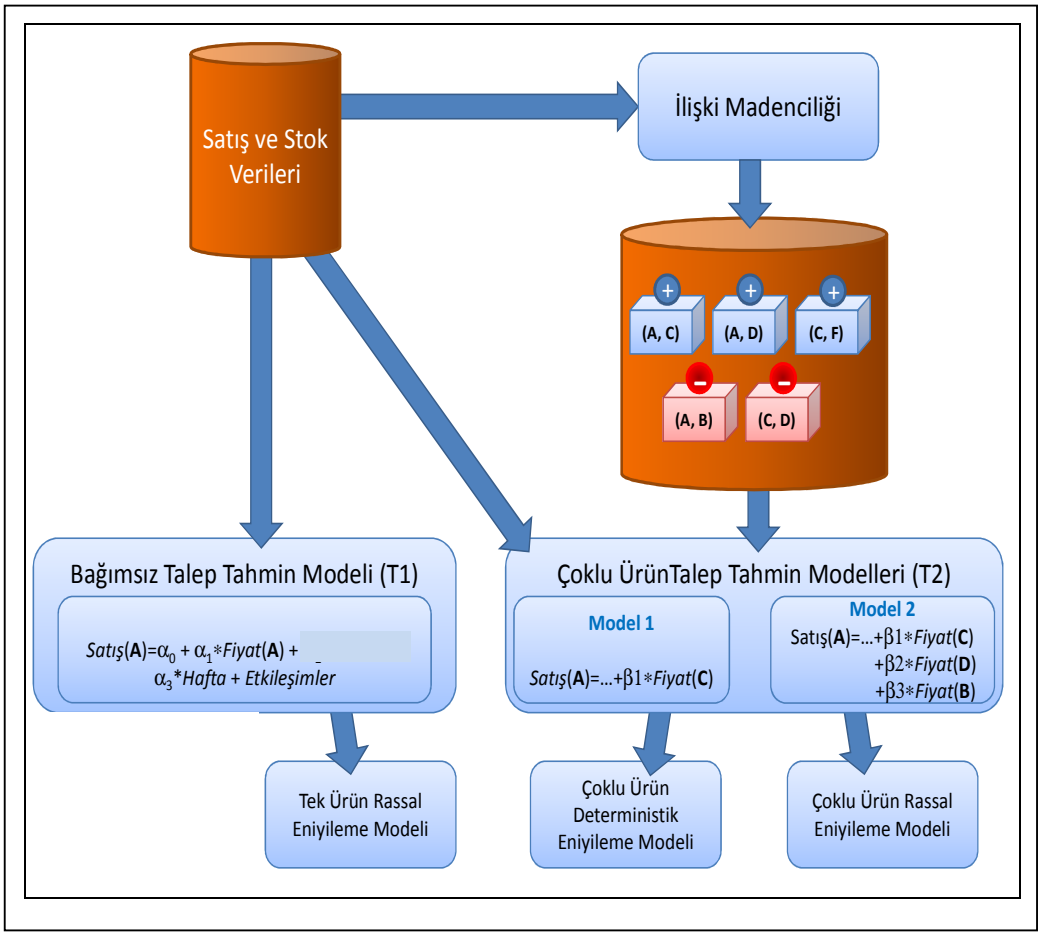
4. STOKASTİK TALEP ALTINDA TEK ÜRÜN KALICI İNDİRİM ENİYİLEMESİ

250'den fazla mağazaya sahip L.C.Waikiki firmasından elde edilen veriler Veri Madenciliği (VM) algoritmaları ile analiz edilerek ürünler arası ilişkiler belirlenmiş, tekil ve çoklu ürün grupları oluşturulmuştur. Ürünler arasında negatif ilişkiler bu ürünlerin birlikte alındıklarını, pozitif ilişkiler ise ürünler arasındaki ikame ilişkilerini ifade eder. Talep tahmini çalışmalarının ilk aşamasında her bir ürün diğer ürünlerin fiyatlarından bağımsız düşünülmüş; ürünün satış miktarı ile kendi fiyatı ve satış haftası ile ilişkisi araştırılmıştır. İkinci aşamada ise gelecek bölümde yer alan VM'den elde edilen pozitif ve negatif ilişkili aday kümeleri göz önünde bulundurularak, kümelerdeki her bir ürünün satış miktarı ve ilişkili olabilecek ürün fiyatları arasındaki anlamlı ilişkiler regresyon yardımıyla araştırılmıştır.

Şekil 4.1'de gösterildiği gibi aralarında anlamlı ilişkiler bulunan ürünler için Çoklu Deterministik ve Rassal Eniyileme modelleri, diğer ürünler için ise Tekil Rassal Eniyileme modelleri kullanılmıştır.

Her ürünün diğer bazı ürünlerle VM sonucunda ilişkisinin bulunması veya bulunan bu ilişkilerin istatistiksel olarak anlamlı olması beklenemez. Bu durumda tekil ürünler için bağımsız talep tahmini yapılmış, stokastik (rassal) eniyileme algoritmaları çalıştırılarak en iyi fiyatlandırma kararları verilmiştir.

Örnek teşkil etmesi açısından aralarında ilişki olmayan iki ürün (A ve B ürünleri) seçilmiş, stokastik dinamik programlama kullanılarak optimal politikalarına karar verilmiştir.



Şekil 4.1 : Modeller arasındaki ilişkiler.

4.1 Bağımsız Talep Tahmin Modeli

2007 yılına ait verilerde 716 tane ürün bulunmaktadır. 716 ürün opsiyonu içinden 15 hafta altında satışı gerçekleşmiş ürün opsiyonları analizlerimize dahil edilmemiştir. Bağımsız talep tahmin modelinde 560 ürün kullanılmıştır.

$d_t(a_t)$ talep fonksiyonu fiyat (a_t) ve zamanın (t) bir fonksiyonudur ve normal dağıldığı farz edilmiştir.

$$d_t(a_t) = \beta_0 + \beta_1 a_t + \beta_2 t + \varepsilon_t$$

$\varepsilon \sim N(0, \sigma^2)$ normal dağılmıştır. Sezon sonuna doğru ürünlerin talepleri azalacağından zamanın talep üzerinde negatif bir etkisi vardır. Son dönemde elde kalan ürünlerin hurda değerine sahip olmadığı, sadece elde tutma maliyeti olduğu ve stok düzeylerinin her zaman hafta başında incelendiği farz edilmiştir.

4.2 Stokastik Eniyileme Modeli

Bu MDP problemi değer yineleme algoritması kullanılarak çözülmüştür. En temel haliyle değer fonksiyonu **(4.1)** deki gibi ifade edilir.

$$V_t(S_t) = \max_{a_t \in A(S_t)} \left\{ r_t(S_t, a_t) + \gamma \sum_{s' \in S_{t+1}} p(s' \mid S_t, a_t) V_{t+1}(s') \right\} \quad \textbf{(4.1)}$$

Amaç geliri enbüyükleyecek optimal politikayı bulmaktır.

$$\pi^* = \arg\max_{\pi} E\left[\sum_{t=1}^{T} V^{\pi}(S_t, a_t^{\pi}) \right] \quad \textbf{(4.2)}$$

Sistem durumu S_t ürünün t anındaki envanter düzeyi s_t ve bir önceki dönemde verilen a_{t-1} kararı ile ifade edilir.

$$S_t = (s_t, a_{t-1}) \quad \textbf{(4.3)}$$

Çünkü *indirim kısıtı* **(4.4)** altında t anındaki a_t fiyat kararı t-1 anındaki S_{t-1} durumunda verilen a_{t-1} fiyat kararından büyük olamaz. Bu nedenle S_t durumunda verilebilecek olası fiyat kararlar kümesi $A(S_t)$ her zaman bu kısıta göre oluşturulmalıdır.

$$a_t \le a_{t-1}, \ \forall t \in T \quad \textbf{(4.4)}$$

$$A(S_t) = \{a_t : a_t \le a_{t-1}\} \quad \textbf{(4.5)}$$

t anında S_t durumunda $a_t \in A(S_t)$ kararı alındığında sisteme r_t kadar bir getiri sağlanmakta ve $t+1$ anında belli bir olasılıkla s' durumuna geçilmektedir. Bu geçiş olasılığı $p(s' \mid S_t, a_t)$ ile ifade edilmektedir. $r_t(S_t, a_t)$ gelir fonksiyonu elde edilen satışlardan elde tutma maliyetinin çıkarılmasıyla elde edilir **(4.6)**. h birim elde tutma maliyetini göstermektedir ve zamandan bağımsız ele alınmıştır.

$$r_t(S_t, a_t) = \min(s_t, d_t(a_t)) \times a_t - \max(0, s_t - d_t(a_t)) \times h \quad \textbf{(4.6)}$$

Sonlu periyotlar için Değer Yineleme Algoritmasının adımları Şekil 4.2'de verilmiştir. Şekilde de verildiği gibi son dönem değer fonksiyonunun aldığı değerin

belli olması gerekmektedir. Bizim modelimizde son dönem ürünlerin hurda değerine sahip olmadığı sadece elde tutma maliyetinin olduğu düşünülmüştür.

Adım 0: İlk değerleri ata:.

Son periyot değer fonksiyonu değerlerini ata, $V_T^0(s) = -s \times h, \forall s \in S$

Adım 1: Her $t = T-1, T-2, ..., 1$ için aşağıdaki adımları uygula

Adım 1a: Her $s \in S$ için V değer fonksiyonlarını hesapla

$$V_t\ (s) = \max_{a \in A(s)} \left\{ r_t(s,a) + \gamma \sum_{s' \in S_{t+1}} p(s' \mid s,a) V_{t+1}(s') \right\}$$

Adım 1b: $t > 1$ ise Adım 1'e git, değilse Adım 2'ye git.

Adım 2: Her $s \in S$ için optimal kararlara karar ver

$$a_t^*(s) = \arg\max_{a \in A(s)} \left\{ r_t(s,a) + \gamma \sum_{s' \in S_{t+1}} p(s' \mid s,a) V_{t+1}(s') \right\}$$

Şekil 4.2 : Sonlu periyotlar için Değer Yineleme algoritması (Puterman, 1994).

4.2.1 Kalıcı indirim politikaları ve analizler

Ürünlere ait satış verileri analiz edilmiş ve seçilen A ve B ürünlerinin talep dağılımları SAS yazılımı ile adım adım (stepwise) regresyon yapılarak aşağıdaki gibi elde edilmiştir.

$$\mu_{At} = 250 - t - 2a_t + \varepsilon_{At}$$

$$\mu_{Bt} = 275 - 0.8t - 3.3a_t + \varepsilon_{Bt}$$

μ ürünlere ait beklenen ortalamayı göstermektedir. ε_t normal dağılmıştır. Talep dağılımları fiyat ve zamana bağlı olarak değişmektedir. Standart sapma σ_t ortalamanın 15%'i olarak alınmıştır. A ürününün başlangıç fiyatı 100 TL, B ürününki ise 60 TL' dir. Her iki üründen başlangıçta 900 adet bulunmaktadır. Elde tutma maliyeti ise adet başına $h = 0,5$ TL/hafta olarak alınmıştır. Her periyot her durum için olası fiyat kararı kümesi $A(S_t)$, 10%, 30% veya 50% indirimli fiyatlarından ya da hiç indirim yapılmadığı fiyattan oluşmuştur. Buna göre A ürünün karar seti $A(S_t) = [100, 90, 70, 50]$, B ürünün ki ise $A(S_t) = [60, 54, 42, 30]$ olabilir.

Buna bağlı olarak taleplerin gelme olasılıkları hesaplanmış ve tüm olası fiyat kararları değerlendirilerek, karı maksimize eden fiyat kararına karar verilmiştir. Tüm periyot boyunca $a_t \geq a_{t+1}$ indirim kısıtı göz önünde bulundurularak kararlar alınmıştır.

Taleplerin gelme olasılıkları normal dağılım yaklaşımı (approximation) ile hesaplanmıştır (Patel ve diğ., 1982)

$$P(X \leq x) = \phi(x),$$
$$2\left[(1-\phi(x)\right] = [1+0,09979271x+0,04432014x^2+0,00969920x^3-0,00009862x^4$$
$$+0,00581551x^5]^{-8}+2\varepsilon(x),$$
$$\left|\varepsilon(x)\right| < 2\times10^{-5}, \qquad x \geq 0$$

Burada $x = \dfrac{talep-\mu}{\sigma}$ ' yı ifade etmektedir. μ talep ortalaması yukarıda verildiği gibi her fiyat kararı ve zamana göre değişmektedir. $\phi(x)$ ise *x*' e kadar olan birikimli olasılıkları göstermektedir. Bu nedenle talebin *x* olma olasılığı $\phi(x)-\phi(x-1)$ 'dır.

Örnek bir talep patikası ele alındığında, A ürününün optimal politikası ve ona ilişkin değerler Çizelge 4.1'de verilmiştir.

Çizelge 4.1'deki sonuçlara bakıldığında 13. haftada stok düzeyi 296 adet iken 10% indirim kararı alınarak 90 TL fiyat kararı verilmiş, 18. haftada 30% indirim kararı ile 70 TL ve 19. haftada 50% indirim kararı ile 50 TL fiyat kararı verilmiştir. Dinamik programlama sondan başa doğru çalıştığı için ve son anda bir karar oluşmadığı için bir fiyat kararı görülmemektedir. Hurda değeri olmadığı için son dönemde sadece elde bulundurma maliyeti ortaya çıkmıştır.

Çizelge 4.1 : A ürününün optimal politikası.

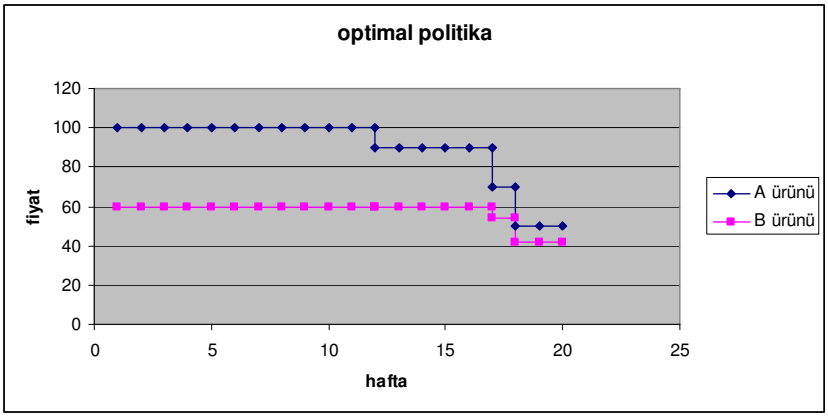
Hafta	Envanter düzeyi	Gelir (TL)	Fiyat Kararı
1	852	78.667	100
2	798	73.839	100
3	741	68.816	100
4	694	64.463	100
5	646	60.053	100
6	607	56.262	100
7	560	51.938	100
8	520	48.104	100
9	466	43.310	100
10	415	38.723	100
11	352	33.289	100
12	322	30.185	100
13	296	27.386	90
14	271	24.679	90
15	248	22.080	90
16	226	18.971	90
17	206	15.678	90
18	188	12.370	70
19	171	6.502	50
20	156	-78	-

B ürününe ilişkin sonuçlar ise Çizelge 4.2' de verilmiştir. Çizelge 4.2' ye göre $t=18$ anında 10% indirim kararı ile 54 TL, $t=19$ anında da 30% indirim kararı ile 42 TL fiyat kararı verilmiştir. A ürünü B ürününden daha erken indirime gitmiş (12. haftada 10% indirim), son haftalarda da B ürününe göre daha büyük indirimler uygulanmıştır. A ürününde 50%'ye varan indirimler yapıldığı halde, B ürününde en fazla 30% indirim kararı verilmiştir. Bunun nedeni A ürününün fiyatı daha yüksek olduğundan daha fazla indirime giderek talebi artırmak, dolayısıyla envanteri boşaltarak karı arttırmak olmuştur.

Çizelge 4.2 : B ürününün optimal politikası.

Hafta	Envanter düzeyi	Gelir	Karar
1	852	48.849	60
2	798	45.881	60
3	741	42.732	60
4	694	40.120	60
5	646	37.440	60
6	607	35.250	60
7	560	32.604	60
8	520	30.340	60
9	466	27.274	60
10	415	24.360	60
11	352	20.740	60
12	322	19.004	60
13	296	17.494	60
14	271	16.040	60
15	248	14.697	60
16	226	13.371	60
17	206	11.511	60
18	188	8.867	54
19	171	5.039	42
20	156	-78	-

Ürünlerin zamana bağlı fiyat politikaları Şekil 4.3'te verilmiştir. Algoritma sonucunda her t anında ve her s_t stok düzeyinde ne karar verileceği bellidir. Bir ürüne ait stok düzeyi en fazla 900 adet olabileceğinden ve her durum için 4 olası karar uygulandığından, değerlendirilen durum sayısı $900 \times 4 = 3600$ adettir. Demek ki her durum için 3600 tane de geçiş olasılığı hesaplanmıştır.



Şekil 4.3 : Ürünlerin optimal fiyat politikaları grafiği.

Bunu aralarında korelasyon olan k tane ürün için yapacak olursak;

$\underbrace{900\times...\times900}_{k}\times\underbrace{4\times...\times4}_{k}=(3,6\times1000)^{k}$ tane durum oluşacaktır.

Bu kadar durum demek bu kadar geçiş olasılığının hesaplanması demek olacağından dinamik programlama ile bu problemi çözmemiz mümkün olmayacaktır. Bu nedenle tüm durum uzayının ve geçiş olasılıklarının hesaplanmasına gerek duyulmayan Ödüllü Öğrenme kullanılmıştır.

5. STOKASTİK TALEP ALTINDA ÇOKLU ÜRÜN KALICI İNDİRİM ENİYİLEMESİ

Tezimizde perakendecilik sektöründe karşılaşılan, kalıcı indirim eniyileme problemi olarak adlandırılan birçok dönemli problem ele alınmıştır. Uygulaması hazır giyim perakende sektöründe yer alan L.C.Waikiki firmasında gerçekleştirilmiştir. Ürün çeşiti çok fazla olduğu için hangi ürünün hangi ürüne ikame, hangi ürüne tamamlayıcı olduğu gibi bilgiler bu etkileri analiz edeceğimiz için önem taşımaktadır.

L.C.Waikiki firmasından elde edilen satış verilerinde ürünlere ait başlangıç stok düzeyleri, hangi haftalarda ne kadar satış yapıldığı ve ne fiyata satıldıkları gibi bilgiler yer almaktadır. Bu veriler SAS yazılım paketi ile analiz edilmiş ve aralarında pozitif ve negatif ilişki bulunan ürünler bir araya getirilerek çoklu ürün grupları oluşturulmuştur. Çoklu grup bulmada şöyle bir yaklaşım kullanılmıştır: Temel (basic) ürünler (satış miktarı mevsimsellikten etkilenmeyen ürünler) inceleme dışı bırakılmıştır. Bu tür ürünlerde kalıcı indirim politikaları uygulanmamaktadır. Bu yüzden ürün kümesi filtrelenmiş ve kalıcı indirim politikası uygulanan ürünler firma yetkilileriyle yapılan görüşmelerden sonra belirlenmiştir. Bu yaklaşımda aşağıdaki adımlar izlenmiştir.

1. Bu ürünler satış rakamlarına göre azalan şekilde sıralanmış ve en çok satan üründen itibaren grup oluşturma işlemine başlanmıştır.
2. Ürün kümesinden seçilen bir ürün için bu kümede yer alan diğer ürünlerle en yüksek desteğe sahip olduğu iki pozitif ilişki seçilmiştir.
3. İkinci adımda seçilen bu ürün için geri kalan ürünler arasında negatif ilişkiye sahip olduğu iki ürün daha seçilir. Böylelikle ilgilenilen ürün için iki pozitif ve iki negatif ilişki bulunmuş ve en fazla beş üründen oluşan bir grup oluşturulmuştur.
4. İkinci ve üçüncü adım var olan tüm ilişkileri çıkarmak için tekrar edilir.

Bu adımlar sonucunda 600 ürün arasından kalıcı indirim uygulanabilecek 55 ürünü kapsayan bir alt küme bulunmuştur. Diğer adımların da birkaç kez tekrarı sonucunda çoklu ürün grupları elde edilmiştir.

Bir ürünün fiyatı arttığında diğer ürünün talebinde bir artış gözleniyorsa, bu ilişki negatif bir ilişkidir ve bu ürünlere ikame ürünler denir. Ancak bir ürünün fiyatı arttığında kendi talebinde bir azalma olurken diğer ürünün talebinde de bir azalma gözleniyorsa, bu ilişki pozitif bir ilişkidir ve bu ürünlere tamamlayıcı ürünler denir.

Tezimizde iki ve üç ürüne sahip çoklu ürün grupları incelenmiştir. Yaklaşık Değer Yineleme (YDY) ve SARSA algoritmasından elde edilen sonuçların karşılaştırılmasının yanı sıra her dönem talebin bilindiği farz edilerek oluşturulan doğrusal olmayan deterministik model GAMS ile çözülmüş ve sonuçları diğer algoritma sonuçları ile kıyaslanmıştır. Böylece bu algoritmaların deterministik modele ne kadar yaklaştığı, anlamlı sonuçlar verip vermediği gözlenmiştir. Bir sonraki bölümde deterministik model anlatıldıktan sonra, önerilen diğer algoritma analizlerine yer verilecektir.

5.1 Deterministik Eniyileme Modeli

Doğrusal olmayan deterministik model, GAMS kullanılarak programlanmış ve DICOPT non-lineer çözücüsü ile çözülmüştür. Modelde başlangıç envanter düzeyleri, başlangıç fiyatları ve talep dağılımları bilinmektedir. Ürünler dönem sonlarında elde bulundurma maliyeti ve sezon sonunda da hurda değerine sahiptirler. Amacımız karı enbüyükleyecek optimal fiyat politikasını bulmaktır.

Deterministik modelimizde kullanılan karar değişkenleri aşağıdaki gibidir:

$z =$ amaç fonksiyonu değeri

$a_{it} =$ ürün i'nin t haftasındaki fiyatı

$\beta_{im} =$ talep dağılımında i ürünün özelliklerine ait $m.$ katsayı, $m = 0,1,2,3$

$IS_i =$ ürün i'nin başlangıç stok düzeyi

$IP_i =$ ürün i'nin sezon başındaki fiyatı

$WIS_{it} =$ ürün i'nin t haftasındaki başlangıç stok düzeyi

$WFS_{it} =$ ürün i'nin t haftası sonundaki stok düzeyi

$FS_i =$ ürün i'nin sezon sonunda kalan stok miktarı

$TS_i =$ sezon boyunca satılan toplam i ürünü miktarı

$WD_{it} =$ ürün i için t haftasında gelen talep miktarı

$D_{it} =$ ürün i için t haftasında gelen pozitif talep miktarı

$S_{it} =$ ürün i için t haftasında yapılan satış miktarı

$h_{it} =$ ürün i için t haftasındaki birim elde tutma maliyeti

$sv_i =$ ürün i'nin hurda değeri

$disc(k) = k.$ indirim oranı

$r_{it} =$ t haftasında i ürünü için gelen talep pozitif ise 1, değilse 0 değerini alan ikili değişken

$f(i,t,k) = t$ haftasında ürün i için $k.$ indirim oranı uygulanırsa 1, uygulanmazsa 0 değerini alan ikili değişken

$M =$ çok büyük bir sayı

$n =$ toplam ürün sayısı

Deterministik Eniyileme Modeli

$$\max_{a_{it}} \left\{ \sum_{i=1}^{n} \sum_{t=1}^{T} a_{it} S_{it} + \sum_{i=1}^{n} sv_i FS_i - \sum_{i=1}^{n} \sum_{t=1}^{T} h_{it} WFS_{it} \right\} \quad \textbf{(5.1)}$$

s.t.

$$WD_{it} = \beta_{i0} + \beta_{i1} a_{it} + \sum_{i \neq j} \beta_{i2} a_{jt} + \beta_{i3} t \qquad \forall i = 1,..n \quad \textbf{(5.2)}$$

$$WFS_{it} = WIS_{it} - S_{it} \qquad \forall i, \forall t \quad \textbf{(5.3)}$$

$$WIS_{i1} = IS_i \qquad \forall i \quad \textbf{(5.4)}$$

$$D_{it} \leq M * r_{it} \qquad \forall i, \forall t \quad \textbf{(5.5)}$$

$$D_{it} - WD_{it} \leq M * (1 - r_{it}) \qquad \forall i, \forall t \quad \textbf{(5.6)}$$

$$S_{it} \leq D_{it} \qquad \forall i, \forall t \quad \textbf{(5.7)}$$

$$WIS_{it} \geq S_{it} \qquad \forall i, \forall t \quad \textbf{(5.8)}$$

$$TS_i = \sum_{t=1}^{T} S_{it} \qquad \forall i \quad \textbf{(5.9)}$$

$$FS_i = IS_i - TS_i \quad \forall i \quad \textbf{(5.10)}$$

$$WIS_{it+1} = WIS_{it} - S_{it} \qquad \forall i, \forall t = 1, ..., T \qquad \textbf{(5.11)}$$

$$a_{i,t} \le a_{i,t-1} + M * (1 - f(i,t,k)) \qquad \forall i,t \qquad \textbf{(5.12)}$$

$$a_{i,t+1} \le a_{it} * disc(k) + M * (1 - f(i,t+1,k)) \qquad \forall i,t,k \qquad \textbf{(5.13)}$$

$$\sum_{k} f(i,t,k) = 1 \qquad \forall i,t \qquad \textbf{(5.14)}$$

$$a_{iT} \ge sv_i \quad \forall i \qquad \textbf{(5.15)}$$

$$a_{i1} \le IP_i \quad \forall i \qquad \textbf{(5.16)}$$

Çoklu ürün deterministik eniyileme modelinde amaç **(5.1)**, bir sezondaki toplam geliri enbüyüklemektir. İlk kısıt **(5.2)**, talep denklemini ifade etmektedir. **(5.3)**. kısıt, hafta sonunda elimizde kalan stok miktarının o hafta başındaki stok miktarı ile o hafta boyunca satılan ürün miktarının farkına eşit olmasını sağlamaktadır. **(5.4)**. kısıt birinci haftadaki başlangıç stok miktarının sezon başındaki stok miktarına eşit olması gerektiğini belirtmektedir. **(5.5)** ve **(5.6)** numaralı kısıtlar, haftalık talebin sıfır veya pozitif olmasını sağlamaktadır. **(5.7)** ve **(5.8)** numaralı kısıtlar, haftalık satış miktarının hafta başındaki stok miktarını geçemeyeceğini, **(5.9)** numaralı kısıt toplam satış miktarını ifade etmektedir. **(5.10)** ve **(5.11)** numaralı kısıtlar envanter denge denklemleridir. **(5.12)** ve **(5.13)** indirim kısıtını göstermektedir. *t* anında verilen fiyat kararının bir önceki dönemde verilen karardan büyük olamayacağını sağlamaktadır ve **(5.14)** verilecek kararın indirimli fiyatlardan sadece birinin olmasını sağlamaktadır. Karar kümesi indirimli fiyatlardan oluşmaktadır. Yani *t* anında s_t durumunda, olası fiyat kararlar kümesi $A(s_t)$, $t-1$ anında verilen a_{t-1} kararına göre oluşur. *t* anında 4 karar verilebilir: ya hiç indirim uygulanmayabilir (0%), yada (10%, 30%, 50%) indirim kararlarından biri uygulanabilir. *t* anında verilebilecek karar, $t-1$ anında verilen karardan küçük olması gerektiğinden ($a_t \le a_{t-1}$) olası fiyat kararları kümesi $A(s_t)$:

$$A(s_t) = \{(1-0)a_{t-1}, (1-0,1)a_{t-1}, (1-0,3)a_{t-1}, (1-0,5)a_{t-1}\}$$

değerlerini alabilir. Bu yüzden her *t* anında verilebilecek kararlar değiştiğinden karar uzayı da büyümektedir. **(5.15)** son dönemde verilecek kararın hurda değerinden küçük olamayacağını ve **(5.16)** birinci dönemde verilecek kararın başlangıç fiyatından büyük olamayacağını göstermektedirler.

Deterministik model her iki ürün grubu için çalıştırılmış, sonuçları stokastik eniyileme modelinde elde edilen sonuçlarla kıyaslanmıştır.

5.2 Stokastik Eniyileme Modeli

Çoklu ürün gruplarında yer alan ürünler arasında korelasyon olacağından, bir ürünün optimal indirim politikası diğerini etkilemektedir, bu yüzden bunların optimal politikalarına birlikte karar verilmelidir. Amacımız hangi envanter düzeyinde hangi indirim kararlarının alınacağıdır. s_{jt} t anında j ürününe ait envanter düzeyini, a_{jt}, t anında j ürününe ait kararı göstersin. İndirim kısıtı altında t anında verilen a_{jt} kararı, bir önceki dönemde verilen karar $a_{j,t-1}$' den büyük olamayacağından, vereceğimiz karar her zaman bir önceki dönemde alınan karara bağlıdır. Bu yüzden sistem durumunun bir parametresini bir önceki dönemde alınan karar oluşturmaktadır. Bu durumda sistem durumu S_t, eğer çoklu ürün grubu k tane üründen oluşuyorsa, aşağıdaki gibi ifade edilmektedir:

$$S_t = (s_{1t}, s_{2t}, ..., s_{kt}, a_{1,t-1}, a_{2,t-1}, ..., a_{k,t-1}) \quad \textbf{(5.17)}$$

ÖÖ algoritması birçok iterasyondan oluştuğundan ve talep rassal olduğundan her iterasyonda, t anında, sistem aynı envanter düzeyine yada farklı envanter düzeylerine gelebilir. Eğer herhangi bir iterasyonda aynı envanter düzeyine gelindiyse, $t-1$ anında hangi kararın alındığı önemlidir, çünkü t anında bu karara göre karar verilecektir. Bu durumda S_t durumunun olası kararlar kümesi $A(S_t)$ aşağıdaki gibi ifade edilmektedir. a_{jt}, t anında j ürünün fiyatını göstermektedir.

$$A(S_t) = \{a_{1t}, a_{2t}, ..., a_{kt} : a_{1t} \le a_{1,t-1}, a_{2t} \le a_{2,t-1}, ..., a_{kt} \le a_{k,t-1}\}. \quad \textbf{(5.18)}$$

Eğer çoklu ürün grubunda 4 tane ürün varsa ve her biri 5000 adet envantere sahipse ve 5 tanede indirim kararı alınabiliyorsa, toplam durum sayısı $5000^4 \times 5^4 = 15,25 \times 10^{10}$ olacaktır. Ürün sayısı arttıkça bu sayı daha da artacaktır. Bu problemi klasik dinamik programlama ile çözmemiz mümkün olmayacağı için ÖÖ kullanılmıştır.

Çözüm yöntemi olarak iki algoritma kullanılmıştır. Birincisi ÖÖ algoritmalarından politikaya bağlı SARSA Algoritması, ikincisi Yaklaşık Değer Yineleme (YDY) Algoritmasıdır.

Optimal π politikası her S_t durumu için $\pi(S_t) = [a_{1t}^*, a_{2t}^*, ..., a_{kt}^*]$ şeklinde ifade edilmektedir ki optimal fiyat kararı $a_t^* = \arg\max_{a_t}(Q(S_t, a_t))$ olarak hesaplanır.

t anında S_t durumunda olduğumuzu düşünelim. Yeni bir a_t kararı alındığında, D_t talebi oluşmakta ve sistem durumu S_t' den S_{t+1} durumuna geçiş yapmaktadır. Bu geçiş fonksiyonu

$$S_{t+1} = (S_t, a_t, D_t(a_t)) \qquad \textbf{(5.19)}$$

şeklinde ifade edilir. Modelimizde karşılanamayan talep bir sonraki döneme aktarılmamakta ve kayıp olarak görülmektedir. Sonuç olarak toplam talep sadece o dönemin talebine, varsa ikame ve tamamlayıcı talepler toplamına eşittir. Talep geldiğinde j ürününün envanter düzeyi s_{jt}, bir sonraki dönemde $s_{j,t+1} = \max(0, s_{jt} - D_{jt})$ olacaktır. Bu stokastik eniyileme probleminin amacı karı enbüyükleyen optimal politikaya karar vermektir:

$$\max_{\pi} E\left[\sum_{t=1}^{T} Q^{\pi}(S_t, a_t^{\pi})\right] \qquad \textbf{(5.20)}$$

Optimizasyon problemlerinde, amaç fonksiyonunun nasıl davrandığı önemlidir. Aynı şekilde kesikli problemlerde de fonksiyonun nasıl davrandığını bilmemiz gerekir. Eğer enbüyükleme yapıyorsak en önemli özelliklerden biri fonksiyonun süpermodüler olup olmadığıdır. Eğer fonksiyonumuzun süpermodüler olduğunu gösterirsek, konkav olduğunu göstermiş oluruz.

Teorem 1: Fonksiyon süpermodüler ise, konkav bir fonksiyondur.

İspat 1: $s^+ \geq s^-$ olan iki durum ve $a^+ \geq a^-$ olan iki karar olduğunu farz edelim. Süpermodüler tanımına göre, $g(s,a)$ fonksiyonu süpermodüler ise,

$$g(s^+, a^+) + g(s^-, a^-) \geq g(s^+, a^-) + g(s^-, a^+). \qquad \textbf{(5.21)}$$

olarak ifade edilir. Benzer şekilde, bu kısıtı

$$g(s^+,a^+)-g(s^+,a^-)\geq g(s^-,a^+)-g(s^-,a^-)$$

şeklinde de yazabiliriz ki, bu ifade *s* değiştiğinde *a'* da ki değişimi göstermektedir.

$v_t(S_t)$ değer fonksiyonunun **(5.22)**, süpermodüler olduğunu göstermek için, $r_t(S_t,a_t)$ gelir fonksiyonu ve $\sum_{s'} P(s'\mid S_t,a_t)v_{t+1}(s')$ ifadesinin süpermodüler olduğunu göstermemiz gerekir.

$$\begin{aligned} v_t(S_t) &= \max_{a_t\in A(S_t)}\{r_t(S_t,a_t)+\gamma v_{t+1}(S_{t+1})\} \\ &= \max_{a_t\in A(S_t)}\{r_t(S_t,a_t)+\gamma\sum_{s'} P(s'\mid S_t,a_t)v_{t+1}(s')\} \end{aligned} \qquad \textbf{(5.22)}$$

Süpermodüler fonksiyonlarının toplamı yine süpermodüler olduğundan (Powell, 2007), $v_t(S_t)$ fonksiyonu da süpermodülerdir denir. Öncelikle, $r_t(S_t,a_t)$ gelir fonksiyonunun süpermodüler olduğunu gösterelim. *k* ürünümüz varsa, $r_t(S_t,a_t)$ gelir fonksiyonu aşağıdaki gibi ifade edilir:

$$\begin{aligned} r_t(S_t,a_t) = a_{1t}\min(S_{1t},D_{1t}) - h_{1t}\text{maks}(0,S_{1t}-D_{1t}) + ... + a_{kt}\min(S_{kt},D_{kt})... \\ - h_{kt}\text{maks}(0,S_{kt}-D_{kt}) \end{aligned}$$

Süpermodüler fonksiyonların toplamının yine süpermodüler olduğundan yola çıkarsak, bunu bir ürün için göstermemiz yeterli olacaktır.

İspat 1a: Birinci ürün için $S_{1t}^+ \geq S_{1t}^-$ ve $a_{1t}^+ \geq a_{1t}^-$ ise **(5.21)** denkleminden, aşağıdaki denklemi elde ederiz:

$$[a_{1t}^+\min(S_{1t}^+,D_{1t})-h_{1t}S_{1t}^+]+[a_{1t}^-\min(S_{1t}^-,D_{1t})-h_{1t}S_{1t}^-] \geq [a_{1t}^-\min(S_{1t}^+,D_{1t})-h_{1t}S_{1t}^+]+...$$
$$[a_{1t}^+\min(S_{1t}^-,D_{1t})-h_{1t}S_{1t}^-]$$

$$[a_{1t}^+\min(S_{1t}^+,D_{1t})-h_{1t}S_{1t}^+]-[a_{1t}^+\min(S_{1t}^-,D_{1t})-h_{1t}S_{1t}^-] \geq [a_{1t}^-\min(S_{1t}^+,D_{1t})-h_{1t}S_{1t}^+]-...$$
$$[a_{1t}^-\min(S_{1t}^-,D_{1t})-h_{1t}S_{1t}^-]$$

$$a_{1t}^+\{\underbrace{\min(S_{1t}^+,D_{1t})-\min(S_{1t}^-,D_{1t})}_{X}\}-\underbrace{h_{1t}(S_{1t}^+-S_{1t}^-)}_{Y} \geq a_{1t}^-\{\underbrace{\min(S_{1t}^+,D_{1t})-\min(S_{1t}^-,D_{1t})}_{X}\}...$$
$$-\underbrace{h_{1t}(S_{1t}^+-S_{1t}^-)}_{Y}$$

$$a_{1t}^+X-Y \geq a_{1t}^-X-Y$$

Görülüyor ki, $a_{1t}^{+} \geq a_{1t}^{-}$ olduğundan $a_{1t}^{+}X - Y \geq a_{1t}^{-}X - Y$' dır ve birinci ürün için fonksiyon süpermodülerdir. Bunu tüm ürünler için genelleştirirsek, süpermodüler fonksiyonların toplamı süpermodülerdir ve $r_t(S_t, a_t)$ gelir fonksiyonu süpermodülerdir.

İkinci olarak, $\sum_{s'} P(s' \mid S_t, a_t) v_{t+1}(s')$ fonksiyonunun süpermodüler olduğunu gösterelim. Bunun için iki durum gereklidir. Birincisi $v_t(S_t)$ değer fonksiyonunun monoton olduğu, ikincisi de geçiş fonksiyonunun monoton olduğudur. Bu önerme için öncelikle yardımcı önerme 1' e ihtiyacımız olacaktır.

Yardımcı Önerme 1: $p_j, p'_j, j \in J$ aşağıdaki eşitsizliği sağlayan her j için tanımlı bir olasılık olsun.

$$\sum_{j=j'}^{\infty} p_j \geq \sum_{j=j'}^{\infty} p'_j, \quad \forall j' \in J \qquad \textbf{(5.23)}$$

Ayrıca v_j azalan bir fonksiyon olsun. O zaman aşağıdaki eşitsizlik sağlanır.

$$\sum_{j=j'}^{\infty} p_j v_j \geq \sum_{j=j'}^{\infty} p'_j v_j,$$

İspat: $v_{-1} = 0$ olsun.

$$\begin{aligned}
\sum_{j=0}^{\infty} p_j v_j &= \sum_{j=0}^{\infty} p_j \sum_{i=0}^{j} (v_i - v_{i-1}) \\
&= \sum_{j=0}^{\infty} (v_j - v_{j-1}) \sum_{i=j}^{\infty} p_i \\
&= \sum_{j=1}^{\infty} (v_j - v_{j-1}) \sum_{j=i}^{\infty} p_i + v_0 \sum_{i=0}^{\infty} p_i \\
&\geq \sum_{j=1}^{\infty} (v_j - v_{j-1}) \sum_{i=j}^{\infty} p'_j + v_0 \sum_{i=0}^{\infty} p'_j \\
&= \sum_{j=0}^{\infty} p'_j v_j
\end{aligned}$$

Böylelikle yardımcı önerme 1 sağlanmış olur.

Önerme 1: Farz edelim ki,

a. $r_t(S_t, a_t)$ azalan bir fonksiyon

b. $r_T(S_T, a_T)$ azalan bir fonksiyon

c. $\sum_{S' \geq S_t} P(S' \mid S_t, a_t)$ azalan bir fonksiyon

olsun. O zaman $v_t(S_t)$ değer fonksiyonu da azalan bir fonksiyondur.

İspat: Bu ispatı tüme varımla yapabiliriz. Son periyotta $v_T(S_T) = r_T(S_T, a_T)$ olduğundan, $t = T$ varsayımı ile sonuca varabiliriz. Gelir fonksiyonunun süpermodüler olduğunu göstermiştik. Tüme varımdan yola çıkarak $t+1$ için v_{t+1} fonksiyonun süpermodüler olduğunu kabul edersek v_t için doğruluğunu göstermemiz gereklidir. $a_t^*(S_t)$ aşağıdaki denklemi çözen optimal karar olsun.

$$\begin{aligned} v_t(S_t) &= \max_{a_t \in A(S_t)} \left\{ r_t(S_t, a_t) + \gamma \sum_{s'} P(s' \mid S_t, a_t) v_{t+1}(s') \right\} \\ &= r_t(S_t, a_t^*(S_t)) + \gamma \sum_{s'} P(s' \mid S_t, a_t^*) v_{t+1}(s') \end{aligned}$$

$\hat{s} \geq s$ ise, (c) durumu $\sum_{s' \geq s} P(s' \mid s, a) \leq \sum_{s' \geq s} P(s' \mid \hat{s}, a)$ olduğunu belirtmektedir.

Yardımcı önerme 1 ile yukarıdaki durumun doğruluğunu göstermiştik. $v_{t+1}(s')$ fonksiyonu da azalan bir fonksiyon ise (tüme varım hipotezinin varsayımı),

$$\sum_{s' \in S} P(s' \mid s, a) v_{t+1}(s') \leq \sum_{s' \in S} P(s' \mid \hat{s}, a) v_{t+1}(s')$$

olur. Bu eşitsizliği önermenin (a) durumu ile birleştirirsek,

$$\begin{aligned} v_t(s) &\leq r_t(s, a_t^*(s)) + \sum_{s' \in S} P(s' \mid s, a) v_{t+1}(s') \\ &\leq \max_{a \in A} \left\{ r_t(\hat{s}, a_t^*(s)) + \sum_{s' \in S} P(s' \mid \hat{s}, a) v_{t+1}(s') \right\} \\ &= v_t(\hat{s}) \end{aligned}$$

eşitsizliğini elde ederiz. Bu sonuçla da, önerme 2' nin (b) durumunu göstermiş oluruz.

Önerme 2: Eğer

a. $q_t(s,a) = \sum_{s' \geq \bar{s}} P(s' \mid s,a)$ süpermodüler ise

b. $v(s)$ azalan bir fonksiyon ise

$\sum_{s' \in S} P(s' \mid s,a)v(s')$ ifadesi süpermodüler bir fonksiyondur.

İspat: Geçiş fonksiyonunun süpermodüler olmasını aşağıdaki gibi ifade edebiliriz:

$$\sum_{s' \geq \bar{s}} P(s' \mid s^+,a^+) + \sum_{s' \geq \bar{s}} P(s' \mid s^-,a^-) \geq \sum_{s' \geq \bar{s}} P(s' \mid s^+,a^-) + \sum_{s' \geq \bar{s}} P(s' \mid s^-,a^+)$$

Yukarıdaki ifadeleri kısaca aşağıdaki gibi yazarsak,

$$p_{\bar{s}} = \sum_{s' \geq \bar{s}} P(s' \mid s^+,a^+) + \sum_{s' \geq \bar{s}} P(s' \mid s^-,a^-)$$
$$p'_{\bar{s}} = \sum_{s' \geq \bar{s}} P(s' \mid s^+,a^-) + \sum_{s' \geq \bar{s}} P(s' \mid s^-,a^+)$$

yardımcı önerme 1' i kullanarak

$p_{\bar{s}} v(s') \geq p'_{\bar{s}} v(s')$

diyebiliriz. Bu da $\sum_{s' \in S} P(s' \mid s,a)v(s')$ ifadesinin süpermodüler olduğunu gösterir.

Sonuç olarak $r(s,a)$ gelir fonksiyonu ve $\sum_{s' \in S} P(s' \mid s,a)v(s')$ fonksiyonu süpermodüler olduğundan, $v_t(s)$ fonksiyonu süpermodülerdir.

Yardımcı Önerme 2: $v_t(s)$ süpermodüler ise, $a_t^*(s) = \text{maks}\left\{ a' \in \underset{a}{\text{argmaks}}(v_t(s)) \right\}$ monoton ve s' e göre azalmayan bir ifadedir. $v_t(s)$ fonksiyonu her s için tek bir optimal karara sahipse, bu ifadeyi aşağıdaki gibi yazabiliriz:

$$a_t^*(s) = \underset{a}{\text{maks}}(v_t(s))$$

İspat: $s^+ \geq s^-$ olsun ve $a^*(s^-) \geq a$ olan bir karar seçtiğimi farz edelim. $a_t^*(s)$, s verildiğinde a' nın en iyi değeri olduğundan,

$$v(s^-, a^*(s^-)) - v(s^-, a) \geq 0$$

olacaktır. Süpermodüler özelliği aşağıdaki özelliği gerektirdiğinden

$$\begin{aligned} v(s^-, a) + v(s^+, a^*(s^-)) &\geq v(s^-, a^*(s^-)) + v(s^+, a) \\ v(s^+, a^*(s^-)) &\geq \underbrace{v(s^-, a^*(s^-)) - v(s^-, a)}_{>0} + v(s^+, a) \\ &\geq v(s^+, a) \end{aligned}$$

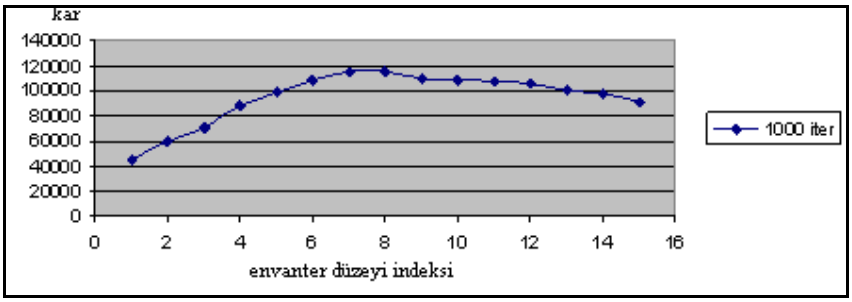
açıkça, $v(s^+, a^*(s^+)) \geq v(s^+, a^*(s^-))$ olduğunu görebiliriz çünkü $a^*(s^+)$ kararı $v(s^+, a)$ değer fonksiyonunu eniyiler. Bu demek oluyor ki $a^*(s^+) \geq a^*(s^-)$' dır.
Sonuç olarak, $v(s)$ değer fonksiyonu süpermodülerdir.

5.2.1 SARSA algoritması analiz sonuçları

Modelimizin amacı karı enbüyükleyen optimal indirimli fiyat politikasına karar vermektir. Karı enbüyüklemekte öncelikle ürünlerin optimal başlangıç envanter düzeylerinin bilinmesi önem taşımaktadır. Analizler iki ve üçlü ürün grupları için yapılmış, ikame ve zaman faktörlerinin optimal politika ve gelir üzerindeki etkileri incelenmiştir.

5.2.1.1 İkili ürün grubu için analiz sonuçları:

Başlangıç envanter düzeylerine karar vermek için farklı envanter düzeylerine (Çizelge 5.1) göre algoritma çalıştırılmıştır. Şekil 5.1'de elde edilen karın konkav bir yapı sergilediği gözlenmektedir. Kar belli bir envanter düzeyine göre artmakta, sonrada azalmaya başlamaktadır. Karın en yüksek elde edildiği (8). envanter düzeyi 4500 ve 1700 adet envantere sahip olunan düzeydir.



Şekil 5.1 : Sistemin konkav yapısı.

Çizelge 5.1 : Şekil 5.1' e ait envanter düzeyleri.

Envanter düzeyi	Ürün 1'e ait envanter düzeyi	Ürün 2'ye ait envanter düzeyi
1	1000	1000
2	1500	1100
3	2000	1200
4	2500	1300
5	3000	1400
6	3500	1500
7	4000	1600
8	4500	1700
9	5000	1800
10	5500	1900
11	6000	2000
12	6500	2100
13	7000	2200
14	7500	2300
15	8000	2400

Dolayısıyla ürünlerin başlangıç envanterleri sırasıyla 4500 adet ve 1700 adet olarak alınmıştır. Ürünlerin başlangıç fiyatları 30 TL/adet ve 20 TL/adet' tir. 10 haftalık bir periyot değerlendirilmiştir. SAS'tan regresyon ile elde edilen talep dağılımlarının ortalamaları aşağıdaki gibidir:

$$D_{1t}(Fiyat_{1t}, Fiyat_{2t}) = 950 - 19Fiyat_{1t} + 15Fiyat_{2t} - 25t + \varepsilon_{1t}$$
$$D_{2t}(Fiyat_{1t}, Fiyat_{2t}) = 700 + 10Fiyat_{1t} - 10Fiyat_{2t} - 15t + \varepsilon_{2t}$$

Talep dağılımları daha öncede değinildiği gibi fiyat ve zamana bağlı olarak değişmektedir. D_{it} , t anında i. ürünün talep dağılımının ortalamasını göstermektedir. Standart sapma, ortalamalarının 15%'i olarak alınmıştır. Her dönem her durum için ya hiç indirim yapılamayacağı (0%) ya da 10%, 30% ve 50% indirim yapılabileceği

kararları değerlendirilmiştir. Sistemin durumu $S_t = (s_{1t}, s_{2t}, a_{1,t-1}, a_{2,t-1})$, her k ürününün envanter düzeyi s_{kt} ve bir önceki dönemde verilen $a_{k,t-1}$ kararlarından oluştuğundan her durum için olası karar kümesi $A(S_t)$, $4^2 = 16$ tane karardan oluşmaktadır. Her iterasyonda örnek bir patika izleneceğinden, her t anında aynı durumlara gelmek algoritmanın daha çabuk yakınsaması açısından önem taşımaktadır. Toplam $4500 \times 1700 \times 16 = 12,24 \times 10^7$ tane durum olacağından tüm durumları ziyaret edebilmek için çok fazla sayıda iterasyon yapmak gerekmektedir. Bu da çok fazla zaman gerektirdiğinden durumları kümeleme (aggregation) yoluna gidilmiştir. Envanter sayısına göre 50'lik ve 100'lük kümeler yapılmıştır. Böylelikle 50'lik kümeleme yapıldığında durum sayısı $(4500/50) \times (1700/50) \times 16 = 48.960$ adet, 100'lük bütünleştirme yapıldığında da durum sayısı $(4500/100) \times (1700/100) \times 16 = 12.240$ adet olmuştur. Mesela birinci ürünün envanter düzeyi 1800 adet, ikinci ürünün 1000 olsun. 50'lik kümelemeye göre ziyaret edilecek küme $[(1800/50)-1] \times (1700/50) + (1000/50)+1=1245$. küme, 100'lük kümelemeye göre ise $[(1800/100)-1] \times (1700/100)+(1000/100)+1=300$. küme olacaktır. Kümenin büyüklüğü arttıkça oraya ziyaret daha da artacak, fakat ortalama ve standart sapmalarında değişiklik olacaktır. Her kümenin ortalama ve standart sapmaları hesaplanmış, 95% güven aralıkları bulunmuştur.

Güven Aralığı

Güven aralığı yeterince yüksek olduğunda 1. tür hata olasılığı yok denecek kadar azdır, böylece sonuçlar $(1-\alpha)\%$ olasılıkla gerçekten önemlidir. L ve U güven aralığının alt ve üst sınırlarını göstersin. 95% güven aralığı bulunduğundan $\alpha = 5\%$ olmaktadır. Bu aralıkta bulunma olasılığı $P(L \le \mu \le U) = 1-\alpha$ olarak ifade edilirse, bu aralığa μ parametresinin $100(1-\alpha)\%$ güven aralığı denilir. Örneklem dağılımı $\bar{X}$, ortalaması μ, standart sapması $\sigma/\sqrt{n}$ olan normal dağılıma uyduğu için istatistik dağılımı

$$Z = \frac{\bar{X} - \mu}{\sigma/\sqrt{n}} \qquad \textbf{(5.24)}$$

standart normal dağılımdır. Z dağılımı $P\{-z_{\alpha/2} \le Z \le z_{\alpha/2}\} = 1-\alpha$ olduğundan

$$P\left\{-z_{\alpha/2} \le \frac{\bar{X}-\mu}{\sigma/\sqrt{n}} \le z_{\alpha/2}\right\} = 1-\alpha \qquad \textbf{(5.25)}$$

şeklinde ifade edilir. Gerekli düzenlemeler yapıldığında **(5.26)** elde edilir:

$$P\left\{\bar{X} - z_{\alpha/2}\sigma/\sqrt{n} \le \mu \le \bar{X} + z_{\alpha/2}\sigma/\sqrt{n}\right\} = 1-\alpha \qquad \textbf{(5.26)}$$

Sonuç olarak μ parametresinin $100(1-\alpha)\%$ güven aralığı **(5.27)'** deki gibidir.

$$\bar{X} - z_{\alpha/2}\sigma/\sqrt{n} \le \mu \le \bar{X} + z_{\alpha/2}\sigma/\sqrt{n} \qquad \textbf{(5.27)}$$

Böylece güven aralığının alt sınırı $L = \bar{X} - z_{\alpha/2}\sigma/\sqrt{n}$, üst sınırı da $U = \bar{X} + z_{\alpha/2}\sigma/\sqrt{n}$ 'dır. **(5.28)**

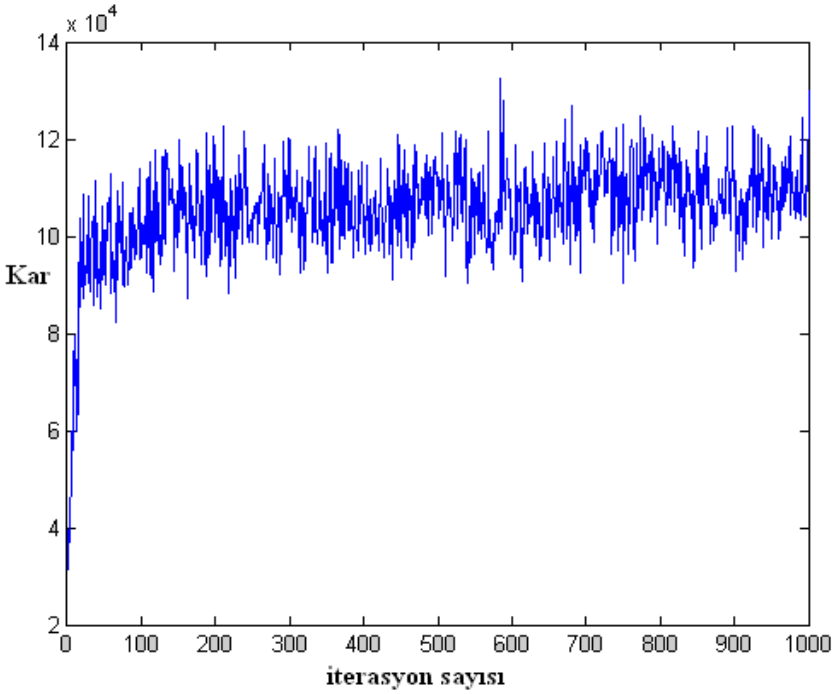
İkame etkisini görmek için iki analiz yapılmıştır: Birincisi ikamenin olduğu ikincisi ikamenin olmadığı durum. Bu bölümde öncelikle 50'lik ve 100'lük kümelemelere göre elde edilen sonuçlar verilmiş, daha sonra karışık kümeleme kullanıldığında ne gibi değişiklikler olduğu ve optimal politikayı nasıl etkilediği gözlenmiştir.
Algoritmanın çıktıları olarak şunlar belirlenmiştir:

1. Her hafta kaç farklı durumun incelendiği
2. Her hafta hangi kümelerin kaç kez ziyaret edildiği
3. Ziyaret edilen kümelerin standart sapması
4. Ziyaret edilen kümelerin beklenen ortalaması
5. Ziyaret edilen kümelerin güven aralıkları
6. Ziyaret edilen durumların optimal politikaları
7. Örnek bir patika için optimal politika ve getirisi
8. Yakınsama grafiği

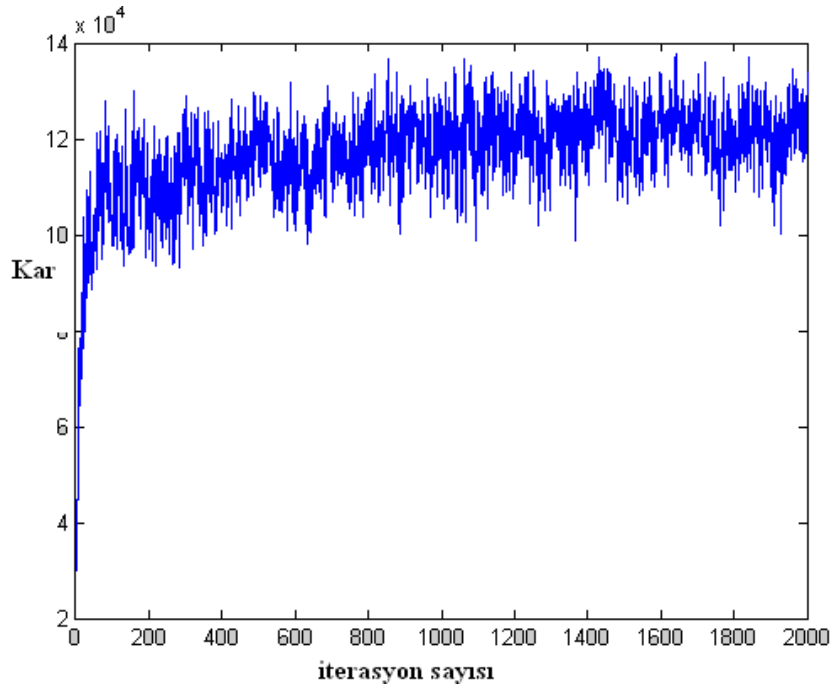
50'lik kümeleme

İterasyon sayısının ne olacağı algoritmanın yakınsaması açısından önem taşımaktadır. Sırasıyla 1000, 2000, 3000 ve 4000 iterasyon olmak üzere algoritma

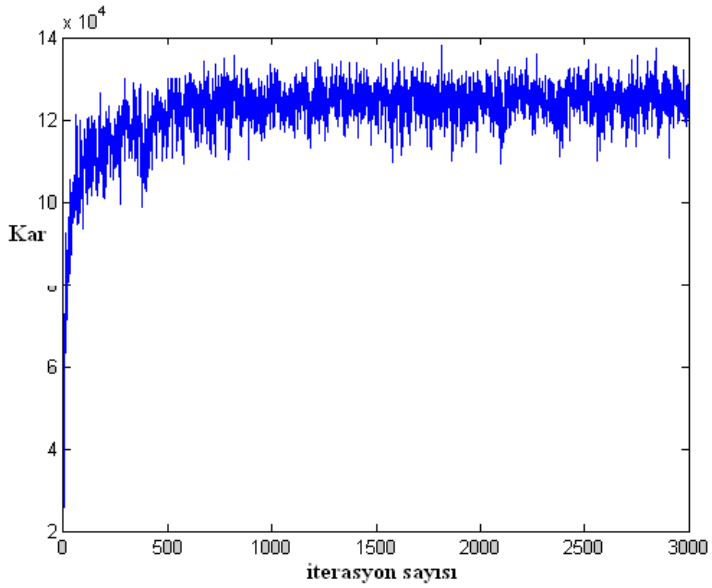
çalıştırılmış, sonuçları kıyaslanmıştır. Şekil 5.2 - 5.5'te görüldüğü gibi sistem 500 iterasyondan sonra yakınsamaya başlamış ve sonrasında da bir değişiklik gözlenmemiştir.



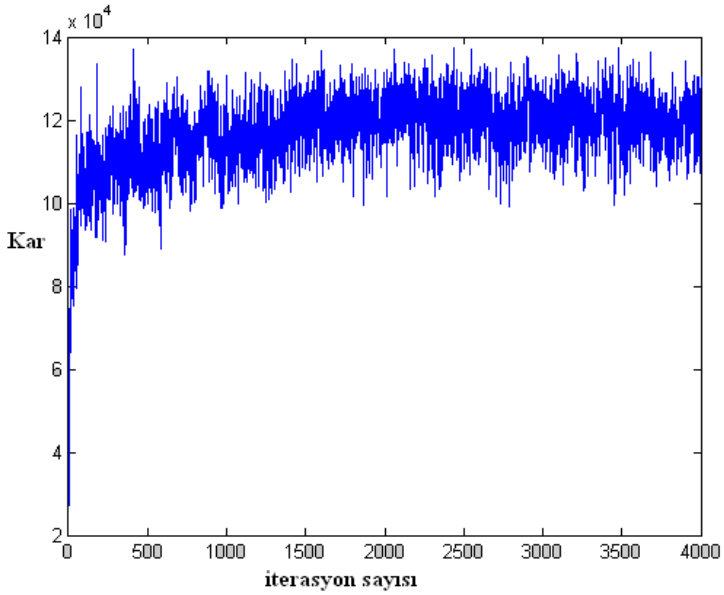
Şekil 5.2 : 1000 iterasyon için yakınsama grafiği.



Şekil 5.3 : 2000 iterasyon için yakınsama grafiği.



Şekil 5.4 : 3000 iterasyon için yakınsama grafiği.



Şekil 5.5 : 4000 iterasyon için yakınsama grafiği.

Fakat başlangıç durumun beklenen ortalamaları ve standart sapmaları kıyaslandığında (Çizelge 5.2), beklenen ortalamalar yaklaşık aynı olmasına rağmen standart sapmalarının iterasyon sayısı arttıkça azaldığı gözlenmiştir. Çünkü aynı durum daha çok ziyaret edildiğinden beklenen değere daha da yaklaşılmış, standart sapma azalmıştır.

Çizelge 5.2 : Başlangıç durumun iterasyon sayısına bağlı sonuçları.

	Başlangıç Kümesine ziyaret sayısı	Standart sapma	Başlangıç Kümesinin beklenen karı	95% güven aralığı Alt Sınır (L)	95% güven aralığı Üst Sınır (U)
1000 iter.	1000	10218	106.502	105.868	107.136
2000 iter.	2000	8820	115.809	115.422	116.196
3000 iter.	3000	8671	114.345	114.035	114.655
4000 iter.	4000	7889	121.345	121.100	121.590

Her iterasyona her zaman aynı durumla (aynı envanter düzeyi) başlandığından, kümeyi ziyaret sayısı iterasyon sayısı kadardır. Bu kümelerin beklenen değerleri, standart sapmaları ve 95% güven aralıkları hesaplanmıştır. İterasyon sayısı arttıkça standart sapma azaldığından güven aralıkları daralarak daha hassas hale gelmiştir.

Deterministik modelden elde ettiğimiz örnek patika için algoritmadan elde edilen optimal indirim politikaları kıyaslandığında, iterasyon sayısı arttığında politikanın aynı değerlere yakınsadıkları görülmüştür (Çizelge 5.3). Sistem 1000 iterasyon çalıştırıldığında birinci ürün için 2. hafta 10% indirim kararı, ikinci ürün ise 3. hafta 10%, 4. hafta 50% indirim kararları alınırken, iterasyon sayısı arttıkça politikalar yakınsamıştır.

Çizelge 5.3 : 1000, 2000, 3000 ve 4000 iterasyon için optimal politikalar.

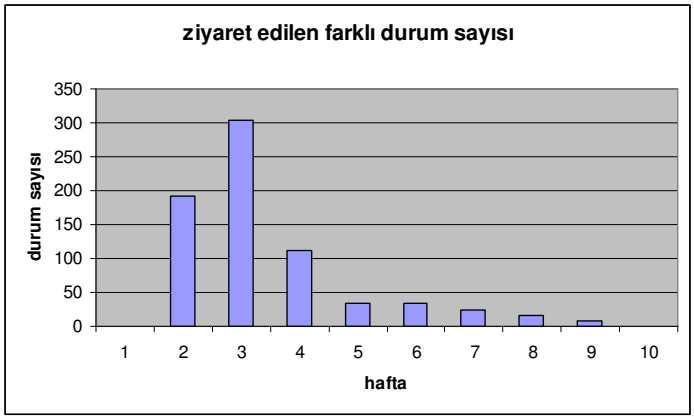
İterasyon sayısı			Hafta									
			1	2	3	4	5	6	7	8	9	10
	Envanter	Ürün1	4500	3845	2930	2070	1235	425	0	0	0	0
	düzeyleri	Ürün2	1700	915	295	0	0	0	0	0	0	0
1000	Optimal	Ürün1	27	24,3	24,3	24,3	24,3	24,3	24,3	24,3	24,3	24,3
	politika	Ürün2	20	20	18	9	9	9	9	9	9	9
2000	Optimal	Ürün1	27	27	27	24,3	24,3	24,3	24,3	24,3	24,3	24,3
	politika	Ürün2	20	20	20	18	18	18	18	18	18	18
3000	Optimal	Ürün1	30	30	30	30	30	30	30	30	30	30
	politika	Ürün2	18	18	18	18	18	18	18	18	18	18
4000	Optimal	Ürün1	30	30	30	30	30	30	30	30	30	30
	politika	Ürün2	18	18	18	18	18	18	18	18	18	18

3000 ve 4000 iterasyon yapıldığında politikalar arasında farklılık olmamasının yanında, beklenen ortalama karları ve standart sapmaları da yaklaşık olarak aynı hesaplanmıştır. İterasyon sayısı arttıkça algoritmanın koşum süresi arttığından, analizler 3000 iterasyon için gerçekleştirilmiştir. 3000 iterasyona göre, örnek patika için 153.432 TL kar elde edilmiş, tüm dönemler boyunca hiç indirim kararı alınmamıştır. Başlangıç durumun beklenen değeri ise 114.000 TL civarında olup deterministik modelden elde edilen karla (110.000 TL) hemen hemen aynıdır. Sonuçlar, talep fonksiyonundan da görüldüğü gibi ikame ve zaman etkilerinin dahil edildiği durumlar için alınmıştır. GAMS yazılımı kullanılarak elde edilen deterministik çözüme göre optimal politika 1. ürün için 30 TL ile başlamış 2. hafta yapılan 50% indirimle periyot sonuna kadar aynı şekilde devam etmiştir. 2. ürün için ise politika 20 TL ile başlamış ve 3. hafta yapılan 10% indirim kararı ile periyot sonuna kadar devam etmiştir (Çizelge 5.3). Sonuçta, SARSA algoritması sonuçlarının gerçek sonuçlardan çok uzak olmadığı görülmüştür. Çizelge 5.4'te 3000 iterasyon için elde edilen sonuçlar verilmiştir. Her periyotta ziyaret edilen durumlardan sadece 1 tanesine yer verilmiştir. Mesela 3. periyotta 304 farklı durum ziyaret edilmiş, bunlardan bir tanesi 29 kez ziyaret edilmiştir. Bu kümenin beklenen ortalaması 64.981 TL'dir.

Çizelge 5.4 : İkamenin olduğu durumda SARSA algoritması analizi.

Periyot	Ziyaret edilen kümeye ziyaret sayısı	Standart sapma	Ziyaret edilen farklı durum sayısı	Ziyaret edilen Kümenin beklenen karı	95% güven aralığı Alt sınır (L)	95% güven aralığı Üst sınır (U)
1	3000	8671	1	114.345	114.035	114.655
2	91	8032	193	89.373	87.723	91.023
3	29	9734	304	64.981	61.438	68.524
4	267	6770	113	43.820	43.008	44.632
5	250	5064	35	30.838	30.210	31.466
6	255	3930	35	16.562	16.080	17.044
7	255	1996	25	6757	6512	7002
8	142	1530	16	6366	6114	6618
9	39	601	8	2029	1840	2218
10	3000	0	1	0	0	0

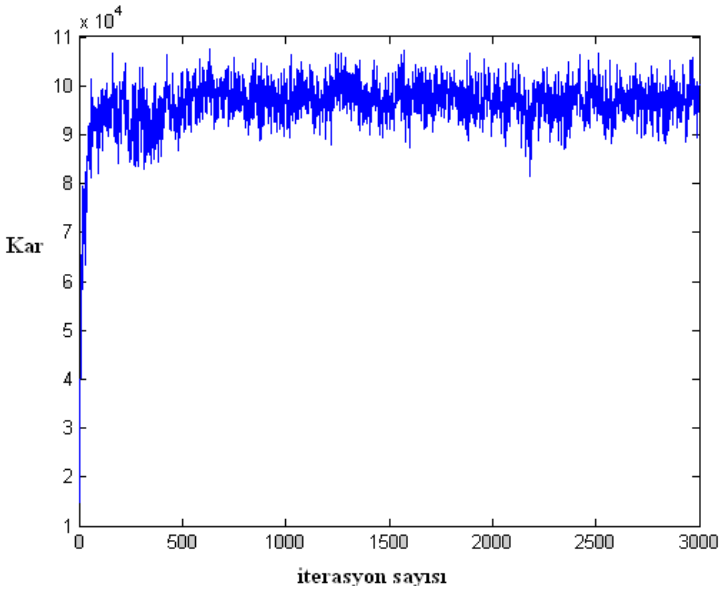
Birinci haftada ziyaret edilen durum sayısının 1 olmasının nedeni her iterasyonda aynı başlangıç envanterleri ile başlanmasıdır. Aynı başlangıç envanteri ile başlandıktan sonra her iterasyonda gelen farklı taleplerle farklı durumlara gidilmekte ve sonuçta algoritma yakınsadığında bu başlangıç envanter düzeyi için beklenilen değer elde edilmektedir. Şekil 5.6'da haftalar bazında ziyaret edilen farklı durum sayıları gösterilmektedir.



Şekil 5.6 : Haftalara göre ziyaret edilen farklı durum sayısı.

Talepler birbirinden bağımsız olduğunda, yani ikame etkisinin ihmal edildiği durumda, talep fonksiyonu aşağıdaki gibi değişecektir. Şekil 5.7' de görüldüğü gibi sistem yine 500 iterasyondan sonra yakınsamaya başlamış, ama beklenen kar düşmüştür.

$$D_{1t}(Fiyat_{1t}) = 950 - 19Fiyat_{1t} - 25t + \varepsilon_{1t}$$
$$D_{2t}(Fiyat_{1t}) = 700 - 10Fiyat_{2t} - 15t + \varepsilon_{2t}$$



Şekil 5.7 : İkamenin olmadığı durumda sistemin yakınsama grafiği.

Çizelge 5.5' te ikamenin olmadığı durumda optimal indirimli politika sonuçları görülmektedir. Sistem 30 TL ve 18 TL başlangıç fiyatlarıyla sezona başlamış, birinci ürün için 3. hafta 30% indirim kararı, 5. hafta 10% indirim kararı alınmıştır. İkinci ürün ise 4 ve 5. haftalarda 10% indirim kararı, 7. hafta 50% indirim kararı alınmıştır. İkame olmadığı durumda talepler azaldığından, sistem fiyatları düşürerek envanteri boşaltmaya çalışmaktadır. Fiyatlar düştüğünden dolayı karda da bir azalma gözlenmiş, kar ikame olduğu durumda 153.432 TL iken ikame olmadığı durumda 111.990 TL'ye düşmüştür.

Çizelge 5.5 : İkame olmadığı durumda optimal politika.

		Hafta									
		1	2	3	4	5	6	7	8	9	10
Envanter düzeyleri	Ürün1	4500	4145	3530	2940	2375	1835	1320	830	365	0
	Ürün 2	1700	1215	745	290	0	0	0	0	0	0
Optimal politika	Ürün 1	30	30	21	21	18,9	18,9	18,9	18,9	18,9	18,9
	İndirim kararı	-	-	30%	-	10%	-	-	-	-	-
	Ürün 2	18	18	18	16,2	14,58	14,58	7,29	7,29	7,29	7,29
	İndirim kararı	-	-	-	10%	10%	-	50%	-	-	-

Başlangıç durumunun beklenilen değeri ise 96.129 TL, standart sapması ise 3307 olarak hesaplanmıştır. Bu durum deterministik çözümle kıyaslandığında, politikanın çok farklı olmadığı karın ise 97.000 TL civarında olduğu gözlenmiştir (Çizelge 5.6). Sonuç olarak ikamenin fiyatlar üzerinde dolayısıyla kar üzerinde pozitif etkilerinin olduğu gözlenmiştir.

Çizelge 5.6 : Deterministik çözümden elde edilen optimal politikalar.

			Hafta									
			1	2	3	4	5	6	7	8	9	10
Temel durum	Optimal politika	Ürün1	30	15	15	15	15	15	15	15	15	15
		Ürün2	20	20	18	18	18	18	18	18	18	18
İkamenin olmadığı durum	Optimal politika	Ürün1	30	15	15	15	15	15	15	15	15	15
		Ürün2	20	20	20	20	20	14	14	14	14	14
Zaman etkisinin olmadığı durum	Optimal politika	Ürün1	30	15	15	15	15	15	15	15	15	15
		Ürün2	20	20	20	20	20	20	20	20	20	20

Zamanın talep üzerinde negatif etkisi olduğu talep dağılımlarından görülmektedir. Çünkü tüketiciler genellikle ihtiyaçlarını sezon başında karşılarlar ve bu istekleri sezon sonuna doğru azalmaktadır. Dolayısıyla zaman etkisini kaldırdığımızda talep ortalamalarında bir artış gözlenecektir. Talep arttığında politikada Çizelge 5.7' deki gibi bir değişim olmuştur. Talep dağılımları aşağıdaki gibidir:

$$D_{1t}(Fiyat_{1t}, Fiyat_{2t}) = 950 - 19Fiyat_{1t} + 15Fiyat_{2t} + \varepsilon_{1t}$$
$$D_{2t}(Fiyat_{1t}, Fiyat_{2t}) = 700 + 10Fiyat_{1t} - 10Fiyat_{2t} + \varepsilon_{2t}$$

Görüldüğü gibi talepler arttığı için fiyatlarda artmıştır. Hatta birinci üründe hiç indirime gidilmemiş ve kar yaklaşık 156.000 TL'ye yükselmiştir. Dolayısıyla zamanın optimal politika ve kar üzerindeki pozitif etkileri gözlenmiştir.

Çizelge 5.7 : Zaman etkisinin olmadığı durumda optimal politika.

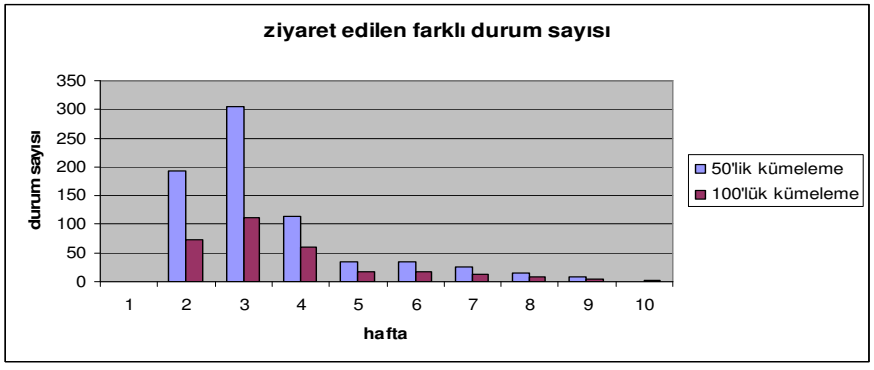
		Hafta									
		1	2	3	4	5	6	7	8	9	10
Envanter düzeyleri	Ürün 1	4500	3820	2855	1890	925	0	0	0	0	0
	Ürün 2	1700	900	250	250	0	0	0	0	0	0
Optimal politika	Ürün 1	30	30	30	30	30	30	30	30	30	30
	İndirim kararı	-	-	-	-	-	-	-	-	-	-
	Ürün 2	18	18	16,2	16,2	16,2	16,2	16,2	16,2	16,2	16,2
	İndirim kararı	-	-	10%	-	-	-	-	-	-	-

100'lük kümeleme:

Oluşturulan kümelere ziyaret ne kadar fazla olursa, istatistiksel açıdan daha doğru sonuçlar elde etmiş oluruz. Bunun için küme büyüklüğü 100' e çıkarılmıştır.

İkame ve zaman etkilerinin dahil olduğu temel durum algoritması çalıştırıldığında aşağıdaki sonuçlar elde edilmiştir.

Bu durumda başlangıç kümesinin beklenen değeri 117.970 TL, 95% güven aralığı ise $117.638 \le \mu \le 118.302$ şeklindedir. Küme büyüklüğü büyüdüğü için ziyaret edilen farklı küme sayısı azalırken bu kümelere olan ziyaret sayısı artmıştır (Şekil 5.8) .



Şekil 5.8 : Her iki kümeleme düzeyine göre ziyaret edilen durum sayıları.

Dolayısıyla standart sapmada artış gözlenmiştir. Mesela, 2. periyotta toplam 73 farklı küme ziyaret edilmiş olup bunlardan biri 334 kez ziyaret edilmiştir, aynı şekilde 6. periyotta ziyaret edilen bir küme 639 kez ziyaret edilmiştir.

Çizelge 5.8 : İkame olduğunda 100'lük kümeleme için SARSA Algoritması sonuçları.

Periyot	Ziyaret edilen kümeye ziyaret sayısı	Standart sapma	Ziyaret edilen farklı küme sayısı	Ziyaret edilen kümenin beklenen karı	95% güven aralığı Alt sınır (L)	95% güven aralığı Üst sınır (U)
1	3000	9275	1	117.970	117.638	118.302
2	334	7953	73	92.778	91.925	93.631
3	186	6911	111	68.280	67.287	69.273
4	477	4419	61	45.182	44.785	45.579
5	503	5545	18	29.721	29.236	30.206
6	639	4056	18	15.866	15.552	16.180
7	544	1892	12	8819	3903	4165
8	250	939	8	4428	4312	4544
9	22	804	5	4685	4349	5021
10	2998	3	2	0	0	0

Optimal politikaya bakıldığında 50'lik kümelemedeki sonuçlarla yaklaşık aynı olduğu görülmektedir (Çizelge 5.9). Sadece 3. dönemde ikinci ürün için 10% indirim kararı verilmiştir. Fakat bu elde edilen gelirlerde çok farklılık yaratmamıştır.

Çizelge 5.9 : 50'lik ve 100'lük kümeleme yapıldığında optimal politika.

			Hafta									
			1	2	3	4	5	6	7	8	9	10
Kümeleme düzeyi	Envanter düzeyi	Ürün1	4500	3845	2930	2070	1235	425	0	0	0	0
		Ürün2	1700	915	295	0	0	0	0	0	0	0
50'lik	Optimal politika	Ürün1	30	30	30	30	30	30	30	30	30	30
		Ürün2	18	18	18	18	18	18	18	18	18	18
100'lük	Optimal politika	Ürün1	30	30	27	27	27	27	27	27	27	27
		Ürün2	18	18	18	18	18	18	18	18	18	18

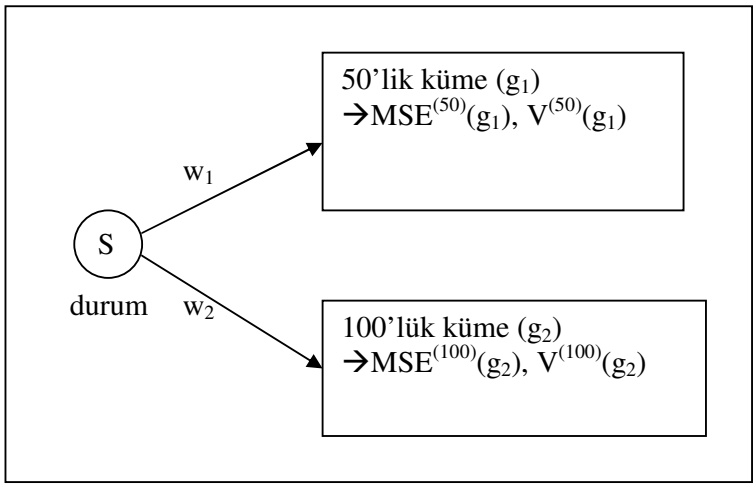
İkame ve zaman etkileri kaldırıldığında örnek patika için oluşan optimal politika ise Çizelge 5.10'da verilmiştir. 50'lik kümelemede elde edilen optimal politika ile kıyaslandığında yaklaşık aynı sonuçlar olduğu görülmektedir. Diğer durumda değerlendirilen örnek patika için; ikame olmadığında talep azaldığından fiyatlarda düşüş, zaman etkisi olmadığında ise talep arttığından fiyatlarda artış gözlenmiştir.

Çizelge 5.10 : 100'lük kümeleme için optimal politika sonuçları.

			Hafta									
			1	2	3	4	5	6	7	8	9	10
İkame etkisi olmadığında	Optimal politika	Ürün1	30	21	21	21	21	21	21	21	21	21
		Ürün2	20	20	18	18	18	18	9	9	9	9
Zaman etkisi olmadığında	Optimal politika	Ürün1	30	27	27	27	27	27	27	27	27	27
		Ürün2	20	20	20	20	20	20	20	20	20	20

Karışık Kümeleme

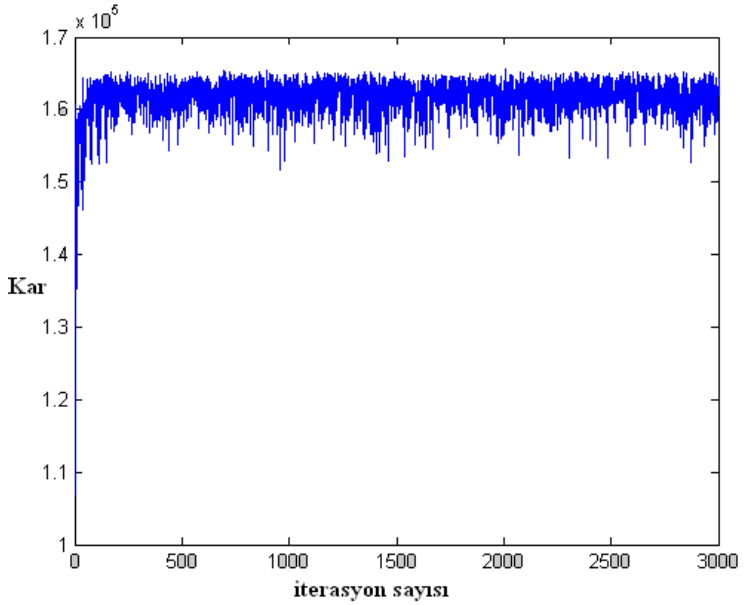
Bu kümeleme 50'lik ve 100'lük kümeleme tekniklerini birlikte içermektedir. 50'lik kümeleme yapıldığında oluşan kümelere 100'lük kümelemede olduğundan daha az ziyaret olacağından standart sapmalarda da farklılık olmuştur. Bu kümelemeye göre, eğer ziyaret edilen S durumu, hem 50'lik kümede (g_1), hem de 100'lük bütünleşik kümede (g_2) yer alıyorsa varyanslarıyla doğru orantılı ağırlıklı ortalamaları alınarak yeni değeri oluşturulmuştur (Şekil 5.9). Bulunan yeni değerle her iki kümenin standart sapması ve beklenen ortalama değeri değişmiştir.



Şekil 5.9 : Karışık kümeleme.

Her iki kümelemenin avantajları birlikte sağlanıldığından, kümelerin ziyaret sayısını artarken standart sapmada da düşüşler gözlenmiştir. Öncelikle sistemin yakınsamasına bakıldığında, sistemin 500 iterasyondan sonra yakınsadığı görülmüştür (Şekil 5.10). Çizelge 5.11'e bakıldığında ise 100'lük kümeleme

durumunda ilk ziyaret edilen durumun standart sapmasının 9275'ten 6220'ye, 50'lik kümeleme durumunda ise 8671'den 5718'e düştüğü gözlenmiştir. Beklenen değerler ise yine aynı civarda oluşmaktadır.



Şekil 5.10 : Karışık kümeleme için yakınsama grafiği.

Çizelge 5.11 : Karışık kümeleme sonuçları.

100'lük kümeleme:						
Kümeleme düzeyi	Periyot	Başlangıç kümesinin kaç kez ziyaret edildiği	Standart sapma	Başlangıç kümesinin beklenen karı	95% güven aralığı Alt sınır L	95% güven aralığı Üst sınır U
Tekil	1	3000	9275	117.970	117.638	118.302
Karışık	1	3000	6220	117.258	117.035	117.481
50'lik kümeleme:						
Kümeleme düzeyi	Periyot	Başlangıç kümesinin kaç kez ziyaret edildiği	Standart sapma	Başlangıç kümesinin beklenen karı	95% güven aralığı Alt sınır L	95% güven aralığı Üst sınır U
Tekil	1	3000	8671	114.345	114.035	114.655
Karışık	1	3000	5718	127.660	127.455	127.865

Optimal politikaya bakıldığında (Çizelge 5.12) yaklaşık aynı olduğu görülmekle birlikte 155.953 TL kar sağlanmıştır. 3. periyotta ikinci ürün için 10% indirim kararı verilmiş ve dönem sonuna kadar aynı fiyatla devam etmiştir.

Çizelge 5.12 : İkame olduğunda karışık kümelemeye göre optimal politika.

		Hafta									
		1	2	3	4	5	6	7	8	9	10
Envanter düzeyi	Ürün1	4500	3845	2930	2070	1235	425	0	0	0	0
	Ürün2	1700	915	295	0	0	0	0	0	0	0
Optimal politika	Ürün1	30	30	30	30	30	30	30	30	30	30
	Ürün2	18	18	16,2	16,2	16,2	16,2	16,2	16,2	16,2	16,2
	indirim kararı	-	-	10%	-	-	-	-	-	-	-

İkame etkisini incelemek için ürünler arasındaki korelasyon ortadan kaldırılmıştır. Diğer kümeleme düzeylerinde analiz edilen örnek patika için elde edilen optimal politika sonuçları Çizelge 5.13'te verilmiştir. İkame olmadığında aynı şekilde fiyatlarda düşüş olduğu görülmektedir. Aynı talepler için analiz gerçekleştirildiğinden, fiyatlar düşünce dolayısıyla kar 155.953 TL'den 124.210 TL' ye düşmüştür. Başlangıç durumun beklenilen karı ise 95.253 TL'ye düşerken deterministik modelden 99.000 TL kar elde edilmiştir. Böylece, ikamenin optimal politika ve kar üzerinde pozitif etkileri olduğu görülmüştür.

Çizelge 5.13 : İkame olmadığında karışık kümeleme için optimal politika.

			Hafta										
			1	2	3	4	5	6	7	8	9	10	Kar
indirim kısıtı olmadığı durum	Optimal politika	ürün1	30	27	27	24,3	21,87	21,87	21,87	21,87	21,87	21,87	124.210 TL
		indirim kararı	-	10%	-	10%	10%	-	-	-	-	-	
		ürün2	20	20	14	12,6	11,34	10,20	5,10	5,10	5,10	5,10	
		indirim kararı	-	-	30%	10%	10%	10%	50%	-	-	-	
indirim kısıtı olduğu durum	Optimal politika	ürün1	30	30	21	18,9	18,9	18,9	18,9	18,9	18,9	18,9	113.944 TL
		indirim kararı	-	-	30%	10%	-	-	-	-	-	-	
		ürün2	20	20	14	14	14	14	9,8	9,8	9,8	9,8	
		indirim kararı	-	-	30%	-	-	-	30%	-	-	-	

Buraya kadar incelenen durumlarda indirim sayısında bir kısıtlama olmadığı farz edilmiştir. "Sezon sonuna kadar en fazla 3 indirim uygulanabilir" şeklinde bir indirim kısıtı getirildiğinde sonuçlar doğal olarak değişecektir. Çizelge 5.12'ye bakıldığında her iki ürün için iki indirim kararı alınmışken, kısıtlamanın olmadığı durumda (Çizelge 5.13), birinci ürün için 3 indirim, ikinci ürün için ise 5 indirim uygulandığı görülmüştür.

Bu durumda beklenen karın 113.944 TL'ye düştüğü görülmektedir. Çünkü indirim kısıtı konulduğunda sistem bir an önce envanteri boşaltma eğiliminde olmuş ve daha büyük indirimler yaparak bunu sağlamaya çalışmıştır. Dolayısıyla fiyatlarda düşüş olduğundan karda da bir düşüş gözlenmiştir. Birinci ürünün politikasında bu durum daha açık görülmektedir.

Çizelge 5.14 : İkame olmadığı durumda ve indirim kısıtı koyulduğunda optimal politika.

		Hafta									
		1	2	3	4	5	6	7	8	9	10
Envanter düzeyi	Ürün1	4500	4145	3530	2940	2375	1835	1320	830	365	0
	Ürün2	1700	1215	745	290	0	0	0	0	0	0
Optimal politika	Ürün1	30	30	21	18,9	18,9	18,9	18,9	18,9	18,9	18,9
	karar	-	-	30%	10%	-	-	-	-	-	-
	Ürün2	20	20	14	14	14	14	9,8	9,8	9,8	9,8
	Karar	-	-	30%	-	-	-	30%	-	-	-

Zaman etkisi kaldırıldığında, aynı şekilde, talepler arttığından fiyatlar da artmış daha az indirim yoluna gidilmiştir (Çizelge 5.15). Fiyatlar ve talepler arttığından kar 155.953 TL'den 161.800 TL'ye yükselmiştir. Zaman etkisi kaldırıldığında ortalama talep dağılımları aşağıdaki gibi olmuştur:

$$D_{1t}(Fiyat_{1t}, Fiyat_{2t}) = 950 - 19Fiyat_{1t} + 15Fiyat_{2t} + \varepsilon_{1t}$$
$$D_{2t}(Fiyat_{1t}, Fiyat_{2t}) = 700 + 10Fiyat_{1t} - 10Fiyat_{2t} + \varepsilon_{2t}$$

Talep üzerinde negatif etkisi olan zamanın optimal politika ve kar üzerinde pozitif etkileri olduğu görülmüştür.

Çizelge 5.15 : Zaman etkisi kaldırıldığında karışık kümeleme için optimal politika.

		Hafta									
		1	2	3	4	5	6	7	8	9	10
Envanter düzeyi	Ürün1	4500	3820	2855	1890	925	0	0	0	0	0
	Ürün2	1700	900	250	250	0	0	0	0	0	0
Optimal politika	Ürün1	30	30	27	27	27	27	27	27	27	27
	İndirim kararı	-	-	10%	-	-	-	-	-	-	-
	Ürün2	20	20	20	20	18	18	18	18	18	18
	İndirim kararı	-	-	-	-	10%	-	-	-	-	-

Ayrı ayrı çapraz fiyat esnekliğinin (E), ürünün kendi fiyatından kaynaklanan esnekliğin (O) ve zaman esnekliğinin (T) talep üzerindeki etkileri ayrıştırılabilir.

Esneklik hesapları şu şekilde yapılmıştır: Her ürün bir fayda fonksiyonuna sahip ve fayda fonksiyonunun parametrelerini ürünlerin fiyatları ve o dönemin etkisi oluşturmaktadır. Dolayısıyla birinci ürünün temsili fayda fonksiyonu **(5.29)**'deki gibidir.

$$v_{1t} = \beta_{0t} + \beta Fiyat_{1t} + \beta Fiyat_{2t} + \beta_{3t} t \qquad \textbf{(5.29)}$$

Daha önceki bölümlerde de anlatıldığı gibi fayda fonksiyonu değerleri satış verilerinden elde edilmişti. Fayda fonksiyonu bilindiği takdirde ürünlerin seçilme olasılıkları (logit olasılıkları) **(5.30)** hesaplanabilmektedir. P_{1t} birinci ürünün t dönemindeki logit olasılığını göstermektedir.

$$P_{1t} = \frac{e^{v_{1t}}}{e^{v_{1t}} + e^{v_{2t}}} \qquad \textbf{(5.30)}$$

Fayda fonksiyonu parametreleri ve logit olasılıkları bulunduktan sonra esneklikler hesaplanmıştır. Örneğin ikinci ürünün fiyatını düşürdüğümüzde birinci ürünün talebindeki değişim oranı (çapraz fiyat esnekliği) aşağıdaki gibi elde edilir:

$$E_{1t} = -\beta_{2t} P_{1t} P_{2t}$$

β_{2t} birinci ürünün fayda denkleminde ikini ürünün fiyatına ilişkin katsayıyı göstermektedir. Birinci ürünün fiyatı değiştiğinde kendi talebindeki değişim oranı (O_{1t});

$$O_{1t} = \beta_{1t} P_{1t} (1 - P_{1t})$$

ile hesaplanır, zaman etkisini değiştirdiğimizde ise zaman esnekliği aşağıdaki gibi hesaplanır.

$$T_{1t} = \beta_{3t} P_{1t} (1 - P_{1t})$$

Çizelge 5.16 ve 5.17' de değerlendirilen örnek patika (talepler) için hafta bazındaki esneklikler görülmektedir.

Çizelge 5.16 : Birinci üründe görülen esneklikler.

Hafta	İlk talep	Çapraz Esneklik	Kendi esnekliği	Zaman esnekliği
1	655	17,32%	-11,42%	-2,84%
2	915	17,89%	-11,80%	-2,94%
3	860	-18,76%	6,70%	-3,76%
4	835	0	0	0
5	810	0	0	0
6	425	0	0	0

Çizelge 5.17 : İkinci üründe görülen esneklikler.

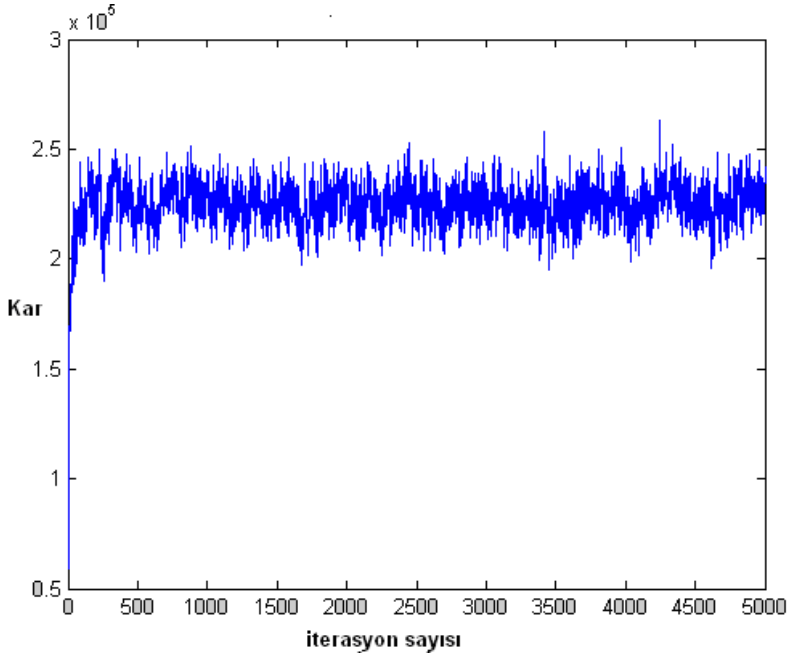
Hafta	İlk talep	Çapraz Esneklik	Kendi esnekliği	Zaman esnekliği
1	785	-17,32%	11,42%	2,84%
2	620	-17,89%	11,80%	2,94%
3	295	18,76%	-6,70%	3,76%
4	0	0	0	0
5	0	0	0	0
6	0	0	0	0

Çapraz esnekliklerde (-) olarak ifade edilen oranlar incelenen üründen diğer ürüne o oranda geçiş olacağını, (+) olanlar ise diğer üründen incelenen ürüne o oranda geçiş olacağını göstermektedir. Birinci ürüne birinci hafta 655 talep gelmiş, ikinci ürünün fiyatındaki değişimden dolayı ikinci üründen birinci ürüne 17,32% oranında ürün geçişi olmuş, kendi fiyatındaki değişimden dolayı kendi talebinde 11,42% oranında azalma, zaman faktörünün değişiminden dolayı yine talebinde 2,84% oranında azalma gözlenmiştir. İkinci üründe 3. haftadan sonra elde kalmamış, bu nedenle her iki ürünün esneklikleri 0' dır.

5.2.1.2 Üçlü ürün grubu için SARSA analiz sonuçları

Benzer analizler üçlü ürün grubu için de gerçekleştirilmiştir. Fakat ürün sayısı arttığından incelenecek olan durum sayısı artmıştır. Daha çok durum ziyaret edebilmek için iterasyon sayısı 5000' e çıkarılmıştır. Bu durumda da algoritmanın koşum süresi artmıştır. İkili ürün grubunda 5000 iterasyon için koşum süresi yaklaşık

30 dk. iken, üçlü ürün grubunda bu süre 2,5 saate kadar çıkmaktadır. 5000 iterasyon için yakınsama grafiği Şekil 5.11'de verilmiştir. Yaklaşık 500 iterasyondan sonra sistem yakınsama göstermiştir.



Şekil 5.11 : Üçlü ürün grubu için yakınsama grafiği.

Her ürünün envanter düzeyi 3000 adet olarak alınmıştır. Başlangıç fiyatları sırasıyla 31 TL, 31 TL ve 36 TL'dir. Oluşturulan çoklu ürün grubu için regresyonla elde edilen talep tahminleri aşağıdaki gibidir:

$$D_{1t}(Fiyat_{1t}, Fiyat_{2t}, Fiyat_{3t}) = 400 - 10Fiyat_{1t} + 5Fiyat_{2t} + 11Fiyat_{3t} - 20t + \varepsilon_{1t}$$
$$D_{2t}(Fiyat_{1t}, Fiyat_{2t}, Fiyat_{3t}) = 950 - 20Fiyat_{1t} - 8Fiyat_{2t} + 17Fiyat_{3t} - 15t + \varepsilon_{2t}$$
$$D_{3t}(Fiyat_{1t}, Fiyat_{2t}, Fiyat_{3t}) = 800 - 12Fiyat_{1t} + 12Fiyat_{2t} - 9Fiyat_{3t} - 10t + \varepsilon_{3t}$$

Ürünlerin talep tahminlerine bakıldığında, ürün 1'in ürün 2 ve 3 ile negatif ilişkiye sahip olduğu yani birbirine ikame ürünler olduğu görülmektedir. Çünkü ürün 2 ve 3'ün fiyatları arttığında ürün 1'in talebinde artış gözlenecek, düştüğünde ise talepte azalış gözlenecektir. Ürün 2'nin ikamesi ürün 3, ürün 1'in ise tamamlayıcı ürün olduğu görülmektedir. Çünkü ürün 1'in fiyatı arttığında ürün 2'nin talebinde düşüş gözlenecektir. Ürün 3'e bakıldığında ürün 1'in tamamlayıcı ürün ürün 2'nin de ikame ürün olduğu görülmektedir. Tüketicilerin talepleri sezon sonuna doğru genellikle azaldığından zamanın etkisi ise negatif olarak görülmektedir.

İkili ürün grubunda olduğu gibi 4 indirim kararı verilebileceği farz edilmiştir. Bu durumda toplam karar sayısı $4\times4\times4=64$ olmaktadır. Her birinin envanter düzeyi 3000 adet olduğundan ve sistem durumu envanter düzeyleri ve kararlardan oluştuğu için toplamda $3000\times3000\times3000\times64=17,28\times10^{11}$ adet durum oluşmaktadır. Kümeleme yöntemi kullanılmadığında her bir durumu ziyaret etmek ya da ziyaret edilen durumlara tekrar gelebilmek için çok sayıda iterasyon yapılması gerekecektir. Bu yüzden 50'lik ve 100'lük kümeleme ve karışık kümeleme tekniği uygulanarak sonuçlar elde edilmiştir. Karışık kümeleme ile standart sapmada düşüş ve ziyaret edilen durumlara ziyaret sayılarının arttığı gözlenmiştir (Çizelge 5.18)

Çizelge 5.18 : Üçlü ürün grubunda her kümeleme düzeyi için analiz sonuçları.

Kümeleme düzeyi		Periyot	Başlangıç kümesinin kaç kez ziyaret edildiği	Std. sapma	Başlangıç kümesinin beklenen karı	95% güven aralığı Alt sınır L	95% güven aralığı Üst sınır U
Tekil	50	1	5000	7422	230.907	230.701	231.113
	100	1	5000	7357	224.319	224.115	224.523
Karışık	50	1	5000	7356	230.391	230.187	230.595
	100	1	5000	6492	218.227	218.047	218.407

Her iterasyona aynı durumla yani envanter düzeyleri ile (3000 adet/ürün) ile başlandığından ve 5000 iterasyon gerçekleştirildiğinden başlangıç kümesinin ziyaret sayısı 5000'dir. Beklenen değerler yaklaşık aynı olmakla birlikte ilk duruma olan ziyaret sayısı fazla olduğundan istatistiksel açıdan 95% olasılıkla güvenilir bir aralıkta olduğu görülmektedir.

Optimal politikalara bakıldığında Çizelge 5.19'daki sonuçlar elde edilmiştir. Karışık kümelemede standart sapma daha az olduğu için bu kümeleme durumu için analizler yapılmıştır.

Çizelge 5.19 : Karışık kümeleme düzeyi için ikame, tamamlayıcı ve zaman etkisi olduğu temel durumda optimal politika.

		Hafta									
		1	2	3	4	5	6	7	8	9	10
Talep	ürün 1	751	718	685	652	619	586	553	436	-	-
	ürün 2	1282	1208	1134	1060	316	-	-	-	-	-
	ürün 3	651	631	611	591	571	551	531	511	352	-
Optimal politika	ürün 1	27,9	25,11	25,11	22,59	22,59	22,59	22,59	22,59	22,59	22,59
	Karar	-	10%	-	10%	-	-	-	-	-	-
	ürün 2	31	27,9	25,11	25,11	25,11	25,11	25,11	25,11	25,11	25,11
	Karar	-	10%	10%	-	-	-	-	-	-	-
	ürün 3	36	32,4	29,16	29,16	26,24	26,24	26,24	26,24	26,24	26,24
	Karar	-	10%	10%	-	10%	-	-	-	-	-

İkame, tamamlayıcı ve zaman etkisinin olduğu temel durumda sezona 27,9 TL, 31 TL, ve 36 TL fiyatlarla başlanmış, birinci ürün için 2 ve 4. haftalarda 10% indirim kararı, ikinci ürün için 2 ve 3. haftalarda 10% indirim kararı ve üçüncü ürün için 2, 3 ve 4. haftalarda 10% indirim kararı verildiği görülmektedir. Beklenen kar ise 230.200 TL olarak hesaplanmıştır. Deterministik modelle kıyasladığımızda 280.000 TL civarında hesaplanan kardan biraz farklı olduğu görülmekte fakat bunun normal olduğu bilinmektedir. Çünkü algoritma ile her kümeleme düzeyi için bir beklenilen değer hesaplanmaktadır. İncelenen her envanter düzeyi bu kümelerden birinde yer almakta ve o kümenin beklenilen değerine sahip olmaktadır. Yani o envanter düzeyine ilişkin hesaplanmış gerçek değer değildir.

İkame etkisi kaldırıldığında ürünlerin taleplerinde azalma olduğundan fiyatlarda düşüş, dolayısıyla karda düşüş gözlenmiştir. Ayrıca, indirimlerin daha büyük oranlarda yapıldığı görülmektedir. Çünkü talep azaldığı için sistem bir an önce elinde kalan envanteri boşaltmak istiyor. Beklenilen kar 112.890 TL olmakla birlikte deterministik modelde bu durum için kar 140.000 TL olarak hesaplanmıştır. Çizelge 5.20'de ikame olmadığı durumda elde edilen optimal politika görülmektedir. Deterministik modelde elde edilen optimal politika ise Çizelge 5.22' de verilmiştir.

Çizelge 5.20 : Karışık kümeleme için ikame olmadığı durumda optimal politika.

		Hafta									
		1	2	3	4	5	6	7	8	9	10
Talep	Ürün 1	70	50	30	10	-	-	-	-	-	-
	Ürün 2	687	672	657	642	342	-	-	-	-	-
	Ürün 3	628	618	608	598	548	-	-	-	-	-
Optimal politika	Ürün 1	21,7	21,7	15,19	15,19	10,63	5,31	5,31	5,31	5,31	5,31
	Karar	-	-	30%	-	30%	50%	-	-	-	-
	Ürün 2	21,7	21,7	21,7	21,7	19,53	13,67	13,67	13,67	13,67	13,67
	Karar	-	-	-	-	10%	30%	-	-	-	-
	Ürün 3	36	18	18	18	18	9	9	9	9	9
	Karar	-	50%	-	-	-	50%	-	-	-	-

Zaman etkisini kaldırdığımızda tüketicilerin alma eğilimi arttığından ayrıca ikame ve tamamlayıcı etkilerden dolayı fiyatlarda artış gözlenmiş ve kar artmıştır. Bu durumda optimal politika sonuçları Çizelge 5.21'de verilmiştir. Oluşan kar ise 250.000 TL civarındadır.

Çizelge 5.21 : Karışık kümeleme için zaman etkisi olmadığında optimal politika.

		Hafta									
		1	2	3	4	5	6	7	8	9	10
Talep	Ürün 1	641	641	641	641	436	-	-	-	-	-
	Ürün 2	694	694	694	694	224	-	-	-	-	-
	Ürün 3	476	476	476	476	476	476	144	-	-	-
Optimal politika	Ürün 1	27,9	27,9	25,11	25,11	25,11	25,11	25,11	25,11	25,11	25,11
	karar	-	-	10%	-	-	-	-	-	-	-
	Ürün 2	31	27,9	27,9	27,9	27,9	27,9	27,9	27,9	27,9	27,9
	karar	-	10%	-	-	-	-	-	-	-	-
	Ürün 3	36	32,4	32,4	32,4	32,4	32,4	32,4	32,4	32,4	32,4
	karar	-	10%	-	-	-	-	-	-	-	-

Zaman etkisi olmadığında ikame ve tamamlayıcı etkilerden dolayı yeteri kadar talep oluşmakta ve çok fazla indirime gerek duyulmamaktadır. Ayrıca zamanın talepler üzerinde negatif etkisinden dolayı sistem envanter boşaltma amacını gerçekleştirmek için daha düşük fiyat kararları vermektedir.

Çizelge 5.22 : Her durum için deterministik model optimal politika sonuçları.

			Hafta									
			1	2	3	4	5	6	7	8	9	10
Temel durum	Optimal politika	Ürün1	31	31	31	31	31	31	31	31	31	31
		Ürün2	31	31	31	31	31	31	31	31	31	31
		Ürün3	36	36	36	36	36	36	36	18	18	18
Zaman etkisinin olmadığı durum	Optimal politika	Ürün1	31	31	31	31	31	31	31	31	31	31
		Ürün2	31	31	31	31	31	31	31	31	31	31
		Ürün3	36	36	36	36	36	36	36	36	36	18
İkamenin olmadığı durum	Optimal politika	Ürün1	31	31	31	31	31	31	31	31	31	31
		Ürün2	31	31	31	31	31	31	31	31	31	31
		Ürün3	18	18	18	18	18	18	18	18	18	18

5.2.2 Yaklaşık değer yineleme algoritması sonuçları

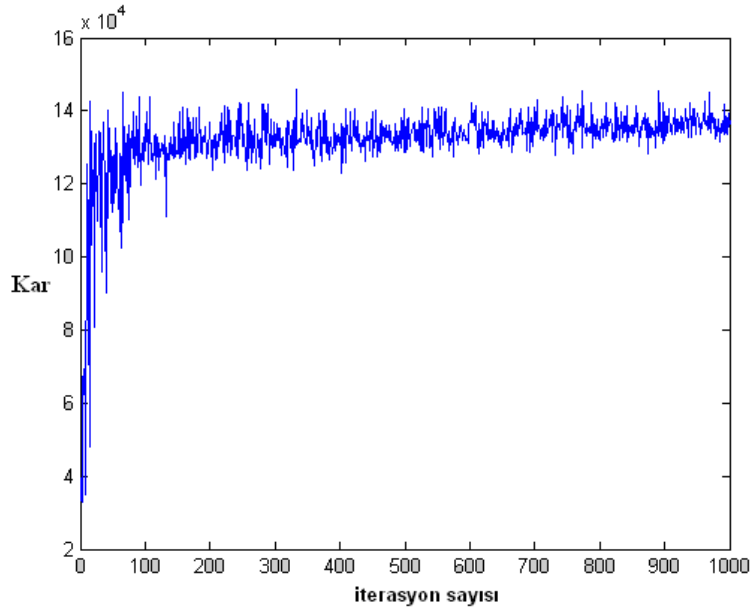
Aynı problem için analizler YDY algoritması ile de yapılmış, ikame ve zamanın optimal politika ve gelir üzerindeki etkileri değerlendirilerek SARSA algoritması ile karşılaştırılmıştır.

5.2.2.1 İkili ürün grubu için YDY sonuçları

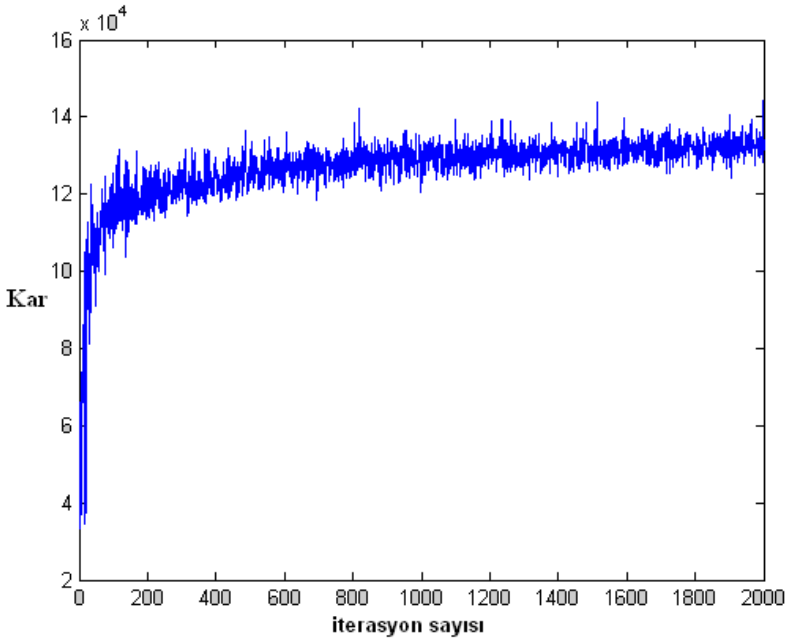
SARSA algoritmasında ikili ürün grubu için veriler bilgiler aynen bu algoritma içinde kullanılmıştır. SARSA algoritmasında her iterasyonda politika güncellenirken, YDY algoritmasında tüm iterasyonlar bittikten sonra politika oluşmaktadır.

50'lik kümeleme

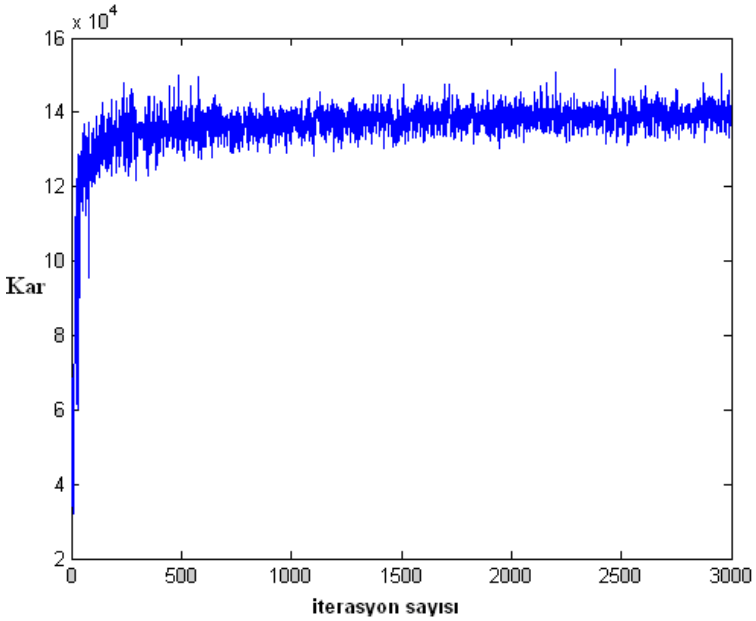
Öncelikle sistem farklı iterasyon sayılarına göre çalıştırılmış, yakınsama grafiklerine göre karşılaştırmaları yapılarak kaç iterasyon için analiz yapılacağı belirlenmiştir.



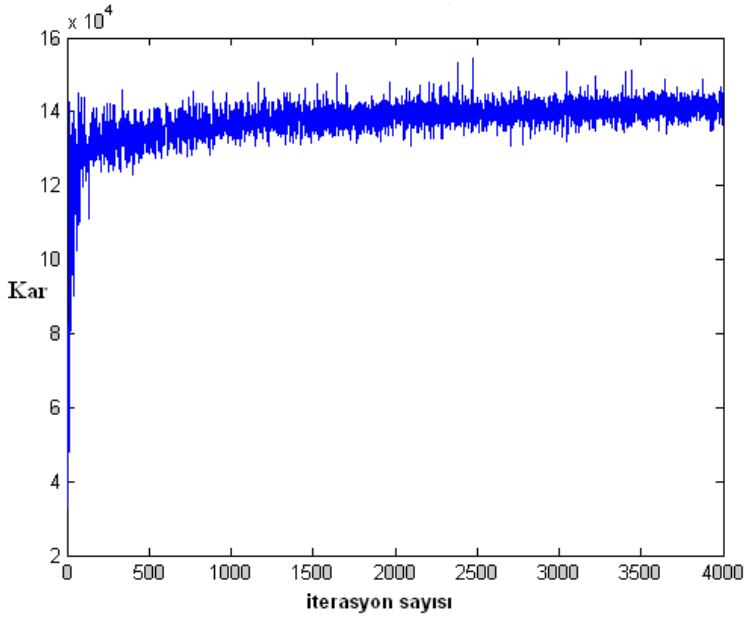
Şekil 5.12 : YDY -1000 iterasyon için yakınsama grafiği.



Şekil 5.13 : YDY – 2000 iterasyon için yakınsama grafiği.



Şekil 5.14 : YDY -3000 iterasyon için yakınsama grafiği.



Şekil 5.15 : YDY – 4000 iterasyon yakınsama grafiği.

YDY Algoritmasına göre yakınsama 500 iterasyondan sonra sağlanmıştır. SARSA Algoritmasıyla kıyasladığımızda sistemin varyansının daha az olduğu görülmüştür (Çizelge 5.23).

Çizelge 5.23 : Başlangıç durumun iterasyon sayısına bağlı sonuçları.

	Periyot	Başlangıç kümesini ziyaret sayısı	Standart sapma	Başlangıç kümesinin beklenen karı	95% güven aralığı Alt sınır L	95% güven aralığı Üst sınır U
1000 iter.	1	1000	4974	132.126	131.818	132.434
2000 iter.	1	2000	4615	135.108	134.906	135.310
3000 iter.	1	3000	4327	119.206	119.051	119.361
4000 iter.	1	4000	4233	124.469	124.338	124.600

İterasyon sayısı arttıkça kümelere olan ziyaret sayısı artmış, standart sapmalarda azalma görülmüştür. Ama bu azalma çok fazla olmadığından, süreden kazanmak için, SARSA algoritmasında olduğu gibi analizler 3000 iterasyon için yapılmıştır.

Aynı analizler bu algoritma için gerçekleştirilmiştir. İkamenin ve zaman etkisinin dahil olduğu temel durum, ikamenin olmadığı durum ve zaman etkisinin olmadığı 3 durum için elde edilen optimal politikalar Çizelge 5.24' te verilmiştir. Kıyaslanması açısından SARSA Algoritmasında kullanılan örnek patika değerlendirilmiştir.

Çizelge 5.24 : 50'lik kümelemeye göre 3 durum için elde edilen optimal politikalar.

			Hafta									
			1	2	3	4	5	6	7	8	9	10
Temel durum	Optimal politika	Ürün1	27	27	27	27	27	27	27	27	27	27
		Ürün2	20	20	20	20	20	20	20	20	20	20
İkame etkisi olmadığında	Optimal politika	Ürün1	27	24,3	24,3	24,3	24,3	24,3	24,3	24,3	24,3	24,3
		Ürün2	18	18	18	16,2	16,2	16,2	16,2	16,2	16,2	16,2
Zaman etkisi olmadığında	Optimal politika	Ürün1	30	30	30	27	27	27	27	27	27	27
		Ürün2	20	20	18	18	18	18	18	18	18	18

İkame ve zaman etkilerinin dahil olduğu durumda sezona 27 TL ve 20 TL fiyatlarla başlanmış ve hiç indirim uygulanmamıştır. Elde edilen kar ise örnek patika için 146.661 TL olup, başlangıç durumunun beklenilen değeri ise 119.000 TL civarındadır. SARSA'dan elde edilen karla hemen hemen aynıdır. İkame

olmadığında ürünlere olana talepler azalacağından fiyatlarda düşüş gözlenmiştir. Sezona 27 TL ve 18 TL fiyatlarla başlanmış, ikinci hafta birinci üründe 10% indirim kararı, 4. hafta ise ikinci üründe 10% indirim kararı verilmiş ve dönem sonuna kadar başka indirim uygulanmamıştır. Elde edilen kar ise 126.275 TL'ye düşmüştür. Zaman etkisinin ihmal edildiği durumda ise taleplerde zaman negatif etkiye sahip olduğundan bu etki kaldırıldığında taleplerde artış gözlenmiş dolayısıyla sistem fiyatları artırmıştır. Bu durumda da elde edilen kar 154.550 TL civarındadır.

100'lük kümeleme yapıldığında da benzer politikalar oluşmuş, fakat ziyaret edilen durumlara ziyaret sayısı artmıştır. Aynı SARSA algoritmasında yapıldığı gibi karışık kümeleme uygulayarak her iki kümeleme düzeyinin avantajlarından faydalanılmıştır.

Çizelge 5.25 : Karışık kümeleme yapıldığında 3 durum için elde edilen optimal politikalar.

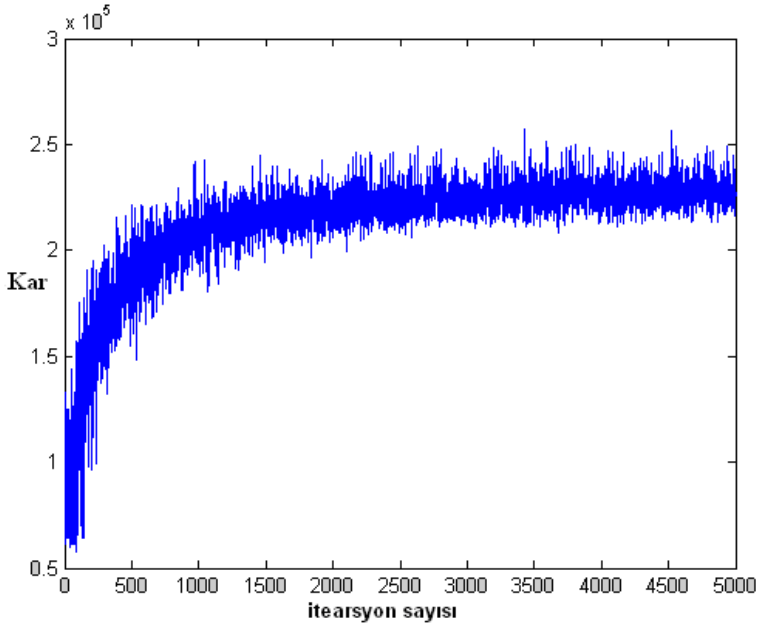
			Hafta									
			1	2	3	4	5	6	7	8	9	10
Temel durum	Optimal politika	Ürün1	27	27	27	27	27	27	27	27	27	27
		Ürün2	20	20	20	20	20	20	20	20	20	20
İkame etkisi olmadığında	Optimal politika	Ürün1	21	21	21	21	21	21	21	21	21	21
		Ürün2	20	20	20	20	20	20	20	20	20	20
Zaman etkisi olmadığında	Optimal politika	Ürün1	30	30	27	27	27	27	27	27	27	27
		Ürün2	20	20	20	20	20	20	20	20	20	20

Karışık kümeleme ile temel durumda standart sapma 4300'lü değerlerden 3700 civarında değerlere düşmüş ve ziyaret edilen durumlara ziyaret sayısı artmıştır.

Temel durumda değerlendirilen örnek patikaya göre birinci üründen 6. haftadan sonra, ikinci üründen 3. haftadan sonra elde kalmamaktadır. Dolayısıyla bu haftalardan sonra fiyat politikası son haftada belirlenen fiyatla devam etmektedir. SARSA algoritması ile kıyasladığımızda birinci ürünün aynı, ikinci üründe ise 3. haftada 10% indirim uygulandığı görülmüştür. Elde edilen kazanç ise 146.474 TL'dir. İkamenin olmadığı durumda birinci ürünün fiyatında belirgin bir düşüş gözlenmiş ve kar 120.280 TL'ye düşmüştür. Zaman etkisinin kaldırıldığı durumda ise fiyatlarda artış sağlanarak karda artış sağlanmış, 151.344 TL'ye yükselmiştir.

5.2.2.2 Üçlü ürün grubu için YDY algoritması sonuçları

SARSA Algoritmasında üçlü ürün grubu için veriler bilgiler bu algoritma içinde aynen kullanılarak analizler yapılmıştır. İkamenin ve diğer etkilerin olduğu temel durum için algoritma 50'lik, 100'lük kümeleme düzeyleri ve her iki düzeyin birlikte değerlendirildiği karışık kümeleme sonuçları Çizelge 5.26'da verilmiştir. Temel durumun yakınsama grafiği ise Şekil 5.16'da görülmektedir.



Şekil 5.16 : Üçlü ürün grubu için YDY algoritması yakınsama grafiği.

Çizelge 5.26 : Üçlü ürün grubunda her kümeleme düzeyi için YDY analiz sonuçları.

Kümeleme düzeyi		Periyot	Başlangıç kümesinin kaç kez ziyaret edildiği	Standart sapma	Başlangıç kümesinin beklenen karı	95% güven aralığı Alt sınır L	95% güven aralığı Üst sınır U
Tekil	50	1	5000	5648	213.123	212.966	213.279
	100	1	5000	6853	236.286	236.096	236.476
Karışık	50	1	5000	5460	223.938	223.787	224.089
	100	1	5000	6065	200.195	200.027	200.363

Karışık kümeleme yapıldığında standart sapmalarda düşüş gözlendiğinden sadece, karışık kümeleme için elde edilen sonuçlara yer verilmiştir.

İkame, tamamlayıcı ve zaman etkilerinin dahil olduğu temel durumda Çizelge 5.27'deki sonuçlar elde edilmiş, ikamenin olmadığı durum için elde edilen optimal politikalar ise Çizelge 5.28' de verilmiştir.

Çizelge 5.27 : Karışık kümeleme için temel durumda elde edilen optimal politika.

		Hafta									
		1	2	3	4	5	6	7	8	9	10
Talep	Ürün 1	751	718	685	652	619	586	553	436	-	-
	Ürün 2	1282	1208	1134	1060	316	-	-	-	-	-
	Ürün 3	651	631	611	591	571	551	531	511	352	-
Optimal politika	Ürün 1	15,5	15,5	15,5	15,5	15,5	15,5	15,5	15,5	15,5	15,5
	Ürün 2	27,9	27,9	27,9	27,9	27,9	27,9	27,9	27,9	27,9	27,9
	Ürün 3	32,4	32,4	32,4	32,4	32,4	32,4	32,4	32,4	32,4	32,4

YDY algoritmasına göre temel durumda sistem sezona sırasıyla 15,5 TL, 27,9 TL ve 32,4 TL fiyatlarla başlamış ve sezon sonuna kadar hiç indirime gitmemiştir. Elde edilen kar ise 212.370 TL'dir. 50'lik ve 100'lük kümeleme yapıldığında da yine aynı politikalar elde edilmiştir. Fakat ilk durumun standart sapması daha fazladır.

Çizelge 5.28 : Karışık kümeleme için ikame olmadığı durumda optimal politika.

		Hafta									
		1	2	3	4	5	6	7	8	9	10
Talep	Ürün 1	70	50	30	10	-	-	-	-	-	-
	Ürün 2	687	672	657	642	342	-	-	-	-	-
	Ürün 3	628	618	608	598	548	-	-	-	-	-
Optimal politika	Ürün 1	21,7	10,85	10,85	10,85	10,85	10,85	10,85	10,85	10,85	10,85
	Karar	-	50%	-	-	-	-	-	-	-	-
	Ürün 2	27,9	19,53	19,53	19,53	19,53	19,53	19,53	19,53	19,53	19,53
	Karar	-	30%	-	-	-	-	-	-	-	-
	Ürün 3	32,4	29,16	29,16	29,16	29,16	29,16	29,16	29,16	29,16	29,16
	Karar	-	10%	-	-	-	-	-	-	-	-

Ürünler arasında ikame olmadığı durumda, sistem sezona 21,7 TL, 27,9 TL ve 32,4 TL fiyatlarla başlamış, ikinci hafta ürünlere sırasıyla 50%, 30% ve 10% indirim kararları uygulanmış ve sezon sonuna kadar aynı fiyatlarla devam etmiştir. Elde

edilen kar ise 105.620 TL'dir. Görüldüğü gibi ürünlere ikame olmadığında talep azalmış, fiyatlarda düşüş olmuştur. Sonuç olarak da beklenen kar düşmüştür.

Çizelge 5.29 : Karışık kümeleme için zaman etkisi olmadığında optimal politika.

		Hafta									
		1	2	3	4	5	6	7	8	9	10
Talep	Ürün 1	641	641	641	641	436	-	-	-	-	-
	Ürün 2	694	694	694	694	224	-	-	-	-	-
	Ürün 3	476	476	476	476	476	476	144	-	-	-
Optimal politika	Ürün 1	21,7	21,7	21,7	21,7	21,7	21,7	21,7	21,7	21,7	21,7
	Ürün 2	27,9	27,9	27,9	27,9	27,9	27,9	27,9	27,9	27,9	27,9
	Ürün 3	32,4	32,4	32,4	32,4	32,4	32,4	32,4	32,4	32,4	32,4

Zaman etkisi kaldırıldığında aynı şekilde yine fiyatlarda artış olmuştur. Dolayısıyla kar da artmıştır.

5.2.3 Fonksiyon yaklaşımı ile değer fonksiyonu tahmin etme

Kümeleme tekniği ile ziyaret edilmeyen durum değerlerinin tahmin edilmesi hedeflenmiştir. Fakat buradaki sorun kümelerin beklenilen değerlerinin doğru tahmin edilebilmesi için kümelere olan ziyaret sayılarının fazla olması gerekliliğidir. Bunun için çok fazla iterasyon gerçekleştirmek gereklidir ama iterasyon sayısı arttığı zaman algoritmanın koşum süresi artmaktadır. Bir diğer çözüm de küme büyüklüğünü artırmak olmuştur fakat bu kez de kümelerin standart sapmaları artmakta ve beklenilen değerlerin güven aralıkları büyümektedir. Karışık kümeleme ile bu sorunlar giderilmeye çalışılmış, yaklaşık sonuçlar elde edilmiştir.

Ziyaret edilmeyen durum değerlerinin tahmini için bir diğer yol da fonksiyon uydurmaktır. Fonksiyon oluştururken SHAPE algoritması kullanılmıştır. Algoritmaya iyi bir başlangıç fonksiyonu ile başlamak daha iyi sonuçlar doğuracağından ilk 1000 iterasyon normal SARSA algoritması çalıştırılmış, regresyon ile fonksiyon katsayıları tahmin edilmiştir. Daha sonra ise SHAPE algoritmasına göre fonksiyon adım adım güncellenmiştir.

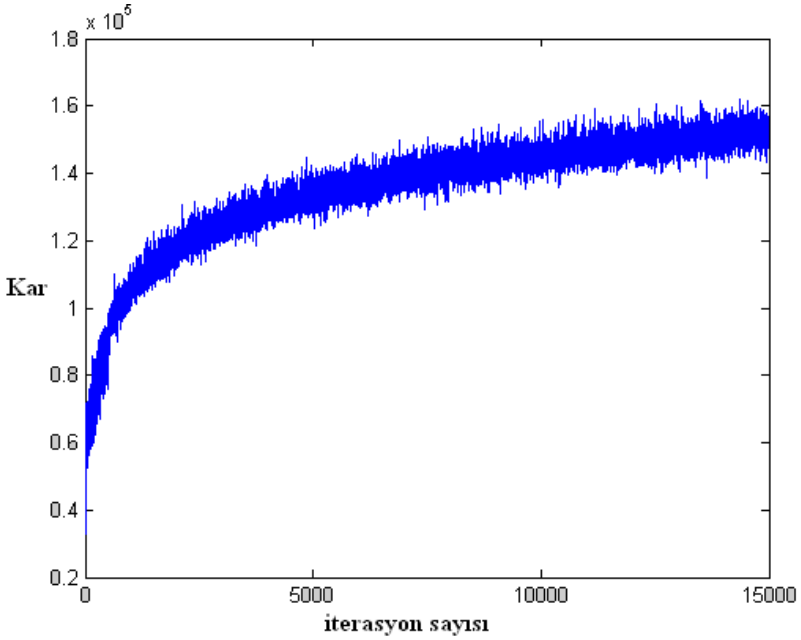
Fonksiyonumuzun parametrelerini ürünlerin fiyat ve envanter düzeyleri oluşturmakta ve her dönem için bir fonksiyon oluşturulmaktadır. Böylelikle sistem parçalı doğrusal fonksiyonlardan (piecewise lineer functions) meydana gelmiştir. Çoklu ürün

grubunda k tane ürün varsa t dönemindeki f fonksiyonu aşağıdaki gibi ifade edilmiştir:

$$f_t(Fiyat_t, Envanter_t) = \theta_{0t} + \sum_{i=1}^{k} \theta_{it} Fiyat_{it} + \sum_{i=1}^{k} \theta_{i+k,t} Envanter_{it}$$

5.2.3.1 İkili ürün grubu için analizler

Aynı ikili ürün grubu için SHAPE algoritması çalıştırılmış ve aynı analizler gerçekleştirilmiştir. Öncelikle sistemin yakınsamasına bakıldığında yaklaşık 5000 iterasyondan sonra yakınsama gözlenmiştir. Görüldüğü gibi sistemdeki durum sayısı arttığı için sistemin yakınsaması için daha fazla iterasyon gerekmiştir. Durumlara olan ziyaret sayısını artırmak ve istatistiksel açıdan daha doğru sonuçlar elde etmek için 15.000 iterasyon gerçekleştirilmiştir.



Şekil 5.17 : Temel durum için yakınsama grafiği.

İkamenin ve zaman etkisinin dahil olduğu temel durumun bazı durumlara ait sonuçları Çizelge 5.30'da verilmiştir.

Çizelge 5.30 : Temel durum için SHAPE algoritması analiz sonuçları.

Periyot	Ziyaret edilen duruma ziyaret sayısı	Standart sapma	Ziyaret edilen farklı durum sayısı	Ziyaret edilen durumun beklenen karı	95% güven aralığı Alt sınır L	95% güven aralığı Üst sınır U
1	15.000	523	1	152.701	152.693	152.709
2	3	5241	14.316	130.691	124.760	136.622
3	2	1741	14.595	96.157	93.744	98.570
4	21	337	2201	70.204	70.060	70.348
5	15	557	1946	55.630	55.348	55.912
6	16	515	1978	29.367	29.115	29.619
7	7	223	1683	13.925	13.760	14.090
8	6	741	1336	8557	7964	9150
9	6	489	977	7497	7106	7888
10	4	624	677	4992	4380	5604

Fonksiyon uydurma ile artık kümelere değil de tek tek durumlara ziyaret sağlanmıştır. Mesela ikinci periyotta 15.000 iterasyonda 14.316 tane farklı duruma ziyaret gerçekleşmiş, bunlardan bazılarına 1'den fazla gelinmiştir. Bunlardan bir tanesine 3 kez gelinmiş ve standart sapması 5241 olarak hesaplanmıştır. İndeksler aynı şekilde hesaplanmıştır: Başlangıç durumu, birinci ürünün 4500 adet, ikinci ürünün 1700 adet olduğu durumdur. Her bir envanter düzeyi incelenebileceği için toplam $4500 \times 1700 + 1 = 7.650.001$ durum vardır.

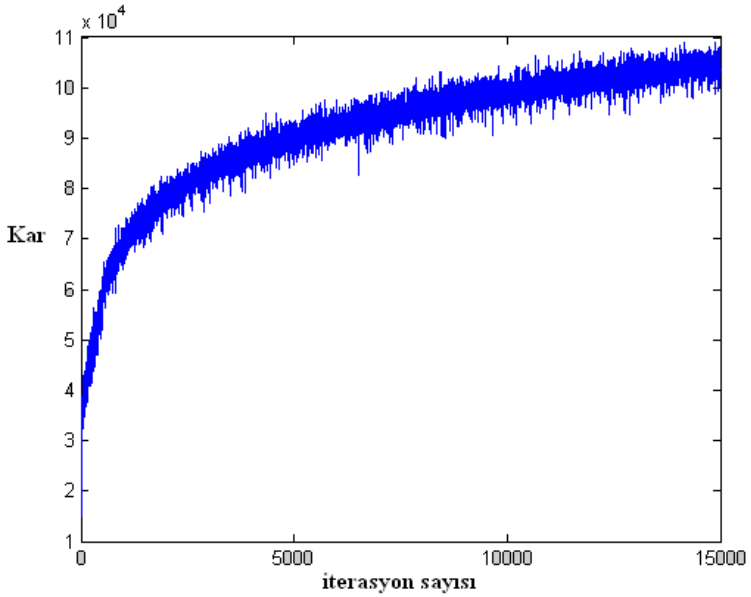
Çizelge 5.31' de ikame ve zaman etkilerinin dahil olduğu temel durumda, ikame olmadığında ve zaman etkisinin olmadığı durumlarda elde edilen yaklaşık optimal politikalar verilmiştir. Temel durumda değerlendirilen örnek patika için 159.850 TL kar elde edilmiştir. Kümeleme Tekniği ile hemen hemen aynı sonuçlar elde edilmiş ama standart sapma çok daha düşük bulunmuştur.

İkame olmadığı durumda talepteki azalmalardan dolayı birinci ürünün fiyatında düşüş gözlenmiştir. Dolayısıyla karda da düşüş gözlenmiş ve 114.890 TL olarak hesaplanmıştır. Deterministik modelde elde edilen 99.000 TL kardan çok farklı olmadığı görülmektedir.

Çizelge 5.31 : Tüm durumlar için elde edilen optimal politika.

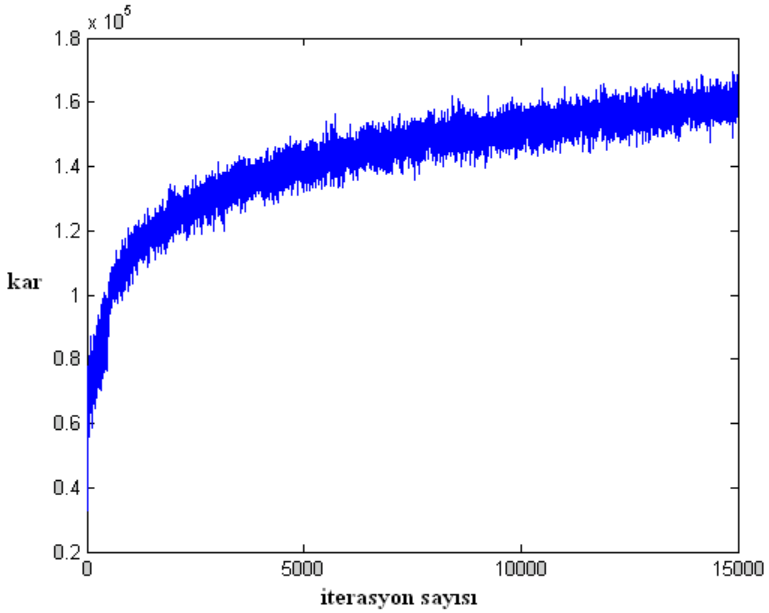
			Hafta									
			1	2	3	4	5	6	7	8	9	10
Temel durum	Optimal politika	Ürün1	30	30	30	30	30	30	30	30	30	30
		Ürün2	20	20	20	20	20	20	20	20	20	20
İkame etkisi olmadığında	Optimal politika	Ürün1	21	21	21	21	21	21	21	21	21	21
		Ürün2	20	20	20	20	20	20	20	20	20	20
Zaman etkisi olmadığında	Optimal politika	Ürün1	30	30	30	30	30	30	30	30	30	30
		Ürün2	20	20	20	20	20	20	20	20	20	20

15.000 iterasyon için yakınsama grafiği Şekil 5.18'de görülmektedir.



Şekil 5.18 : İkame olmadığı durumda sistemin yakınsama grafiği.

Zaman etkisinin olmadığı durumda ise talepler arttığı için fiyatlarda indirime gidilmemiş, 160.510 TL kar elde edilmiştir. Deterministik modelde elde edilen kar ise 106.255 TL'dir. Bu durumun yakınsama grafiği de Şekil 5.19'da görülmektedir.



Şekil 5.19 : Zaman etkisi olmadığı durumda sistemin yakınsama grafiği.

Analiz edilen bu üç senaryo için ilk durum sonuçları Çizelge 5.32'de verilmiştir.

Çizelge 5.32 : Üç senaryo için başlangıç durumu analiz sonuçları.

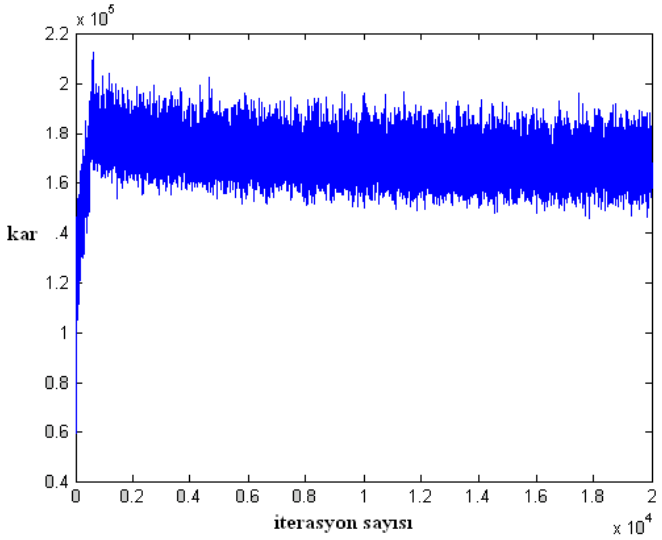
	İterasyon sayısı	Standart sapma	Başlangıç durumunun beklenen karı	95% güven aralığı (L)	95% güven aralığı (U)	Örnek patika için elde edilen kar
Temel durum	15.000	523	152.701	152.693	152.709	159.850
İkame etkisi olmadığında	15.000	402	103.073	103,067	103.079	114.890
Zaman etkisi olmadığında	15.000	257	159.508	159.504	159.512	160.510

Çizelge 5.32'de görüldüğü gibi ikame olmadığında taleplerde düşüş olduğundan kar azalmış, zaman etkisi olmadığında ise taleplerde artış olduğundan kar artmıştır.

5.2.3.2 Üçlü ürün grubu için analizler

Aynı üçlü ürün grubu için analizler SHAPE algoritması ile gerçekleştirilmiş, optimal politika ve gelir sonuçları elde edilmiştir. İkamenin ve zamanın optimal politika ve gelir üzerindeki etkileri ayrıca incelenmiştir. Her bir ürün 3000 envanter düzeyine

sahip olduğundan toplam $3000\times3000\times3000+1=27.000.000.001$ tane durum vardır. Sistem 20.000 iterasyon çalıştırılmış, yakınsama grafiği Şekil 5.20'de verilmiştir.



Şekil 5.20 : Temel durum yakınsama grafiği.

Temel durum sonuçları Çizelge 5.33'te görülmektedir. 8, 9 ve 10. haftalarda envanter kalmadığı için ziyaret edilen durum indeksi 1 ile ifade edilmektedir.

Çizelge 5.33 : Temel durum için elde edilen analiz sonuçları.

Periyot	Ziyaret edilen duruma ziyaret sayısı	Standart sapma	Ziyaret edilen farklı durum sayısı	Ziyaret edilen durumun beklenen karı	95% güven aralığı Alt Sınır (L)	95% güven aralığı Üst sınır (U)
1	20.000	260	1	150.116	150.112	150.120
2	2	9379	19.999	137.439	124.440	150.438
3	1	0	20.000	112.334	112.334	112.334
4	2	907	19.993	100.005	98.748	101.262
5	2	405	11.488	55.613	55.052	56.174
6	6	161	1773	19.838	19.709	19.967
7	4	213	363	1133	924	1342
8	19.985	60	16	0	0	0
9	20.000	60	1	0	0	0
10	20.000	60	1	0	0	0

Temel durumda elde edilen optimal politika ise Çizelge 5.34'te verilmiştir. Buna göre ikinci ve üçüncü ürünler için hiçbir dönem indirim yapılmamış, birinci üründe

ise ikinci dönem 30% indirim kararı alınmış ve sezon sonuna kadar aynı devam etmiştir. Daha fazla iterasyon gerçekleştirildiğinde optimal politikaya daha da yaklaşılacaktır. Çünkü 2, 3 ve 4. dönemlere bakıldığında hemen hemen her iterasyonda farklı durumlara gidilmiştir. Yani ziyaret edilen durumlara tekrar ziyaret sağlanmamıştır. Bu durum sonucunda 256.007 TL kar elde edilmiştir.

İkame etkisi kaldırıldığında, talep azaldığından fiyatlarda düşüş gözlenmiş, dolayısıyla kar 181.397 TL'ye düşmüştür. İkinci ve üçüncü üründe yine indirim uygulanmamış, birinci üründe ise ikinci ve üçüncü dönemlerde 50% indirim uygulanmış ve sezon sonuna kadar bir daha indirim uygulanmamıştır.

Zaman etkisi kaldırıldığında ise fiyatlarda indirime gidilmemiş, talep arttığından kar 278.660 TL'ye yükselmiştir.

Çizelge 5.34 : Her üç durum için elde edilen optimal politikalar.

			Hafta									
			1	2	3	4	5	6	7	8	9	10
Temel durum	Optimal politika	Ürün1	31	21,7	21,7	21,7	21,7	21,7	21,7	21,7	21,7	21,7
		Ürün2	31	31	31	31	31	31	31	31	31	31
		Ürün3	36	36	36	36	36	36	36	36	36	36
İkame etkisi olmadığında	Optimal politika	Ürün1	31	15,5	7,75	7,75	7,75	7,75	7,75	7,75	7,75	7,75
		Ürün2	31	31	31	31	31	31	31	31	31	31
		Ürün3	36	36	36	36	36	36	36	36	36	36
Zaman etkisi olmadığında	Optimal politika	Ürün1	31	31	31	31	31	31	31	31	31	31
		Ürün2	31	31	31	31	31	31	31	31	31	31
		Ürün3	36	36	36	36	36	36	36	36	36	36

Deterministik modelde ise temel durumda ve zaman etkisini kaldırdığımız durumda ürünlerde hiç indirime gidilmemiş, sırasıyla 283.995 TL ve 284.673 TL kar elde edilmiştir. İkame etkisini kaldırdığımızda ise üçüncü ürün sezona 18 TL ile başlamış ve sezon sonuna kadar hiç indirim uygulanmamıştır. Elde edilen kar ise 146.340 TL'dir.

Örnek patika için elde edilen ürünlerin çapraz esneklikleri, kendi fiyatlarındaki esneklikler ve zaman esneklikleri Çizelge 5.35-5.37 arasında verilmiştir. $E(i, j)$, i ürününün j ürününe göre çapraz esnekliğini, yani j ürünün fiyatı değiştiğinde i

ürününün seçim olasılığındaki değişimi göstermektedir. $O(i)$ i ürünün kendi fiyatındaki değişimden kaynaklanan esnekliği, $T(i)$ de i ürünün zaman esnekliğini göstermektedir.

Çizelge 5.35'te verilen birinci ürünün esnekliklerine göre, birinci hafta birinci ürüne, ikinci ürünün fiyat değişiminden dolayı ikinci ürünün 0,43%'ü kadar bir geçiş olmuş, üçüncü ürünün fiyat değişiminden dolayı üçüncü üründen 21,12% oranında bir geçiş olmuştur. Kendi fiyat değişiminden dolayı ise talebinin 20,87%'si kadar bir artış gözlenirken, zaman etkisinden dolayı ise talebinde 24,04% oranında bir kayıp olmuştur. Aynı şekilde diğer ürünlerin esneklikleri de Çizelge 5.36 ve 5.37' de görüldüğü gibidir.

Çizelge 5.35 : Birinci ürünün esneklikleri.

Hafta	İlk Talep	E(1,2)	E(1,3)	O(1)	T(1)
1	621	-0,43%	-21,12%	20,87%	-24,04%
2	601	14,05%	6,33%	-67,78%	20,36%
3	581	-2,86%	-0,86%	9,46%	-5,04%
4	561	15,89%	7,15%	-76,60%	23,01%
5	541	15,38%	8,36%	-78,94%	23,71%
6	95	0,00%	8,21%	5,90%	9,56%

Çizelge 5.36 : İkinci ürünün esneklikleri.

Hafta	İlk Talep	E(2,1)	E(2,3)	O(2)	T(2)
1	679	1,92%	14,95%	-1,92%	18,11%
2	664	9,30%	-1,38%	11,56%	-11,99%
3	649	-31,23%	-8,52%	34,01%	-26,56%
4	634	10,51%	-2,27%	13,75%	-14,26%
5	374	10,18%	-2,87%	13,96%	-14,49%
6	0	0,00%	0,00%	0,00%	0,00%

Çizelge 5.37 : Üçüncü ürünün esneklikleri.

Hafta	İlk Talep	E(3,1)	E(3,2)	O(3)	T(3)
1	466	-32,20%	-21,09%	-2,84%	-3,60%
2	456	16,89%	4,87%	-1,99%	-0,97%
3	446	4,83%	11,05%	15,08%	0,80%
4	436	19,07%	7,99%	-2,47%	-1,20%
5	426	22,29%	10,12%	-2,96%	-1,44%
6	416	-18,19%	-38,05%	-10,20%	1,29%

5.2.4 Kullanılan algoritmaların kıyaslamaları

Tez kapsamında değer fonksiyonlarının tahmini için 2 algoritma kullanılmıştır (SARSA, Yaklaşık Değer Yineleme algoritmaları) ve bu algoritmalarda değer

fonksiyonlarının tahmini için iki teknik kullanılmıştır: kümeleme tekniği ve fonksiyon uydurma tekniklerinden olan SHAPE algoritmalarıdır. Çizelge 5.38'de ikili ürün grubu için, Çizelge 5.39'da üçlü ürün grubu için algoritma kıyaslamaları yer almaktadır. Kıyaslamalar ikame ve zaman etkilerinin dahil olduğu temel durumlar için yapılmıştır.

Çizelge 5.38 : İkili ürün grubu için algoritma kıyaslamaları.

Algoritma	İterasyon sayısı	Koşum Süresi	Başlangıç durumunun std. sapması	Başlangıç durumunun beklenen karı	95% güven aralığı Alt sınır	95% güven aralığı Üst sınır	Örnek patika için elde edilen kar
SARSA_50	3000	14,6 dk.	8671	114.345	114.035	114.655	153.000
SARSA_100	3000	17 dk.	9275	117.970	117.638	118.302	140.580
SARSA_karışık	3000	18,33dk.	5718	127.660	127.455	127.865	155.953
YDY_50	3000	8,95 dk.	4327	119.206	119.051	119.361	146.661
YDY_100	3000	7,53 dk.	4116	138.857	138.710	139.004	158.040
YDY_karışık	3000	9,56 dk.	3729	133.111	132.978	133.244	156.474
SHAPE	3000	6,33 dk.	866	98.949	98.918	98.980	159.850
SHAPE	15.000	1,87 saat	523	152.701	152.693	152.709	159.850
SHAPE	30.000	6,8 saat	304	111.425	111.422	111.428	159.850
Deterministik	-	-	-	-	-	-	104.877

Algoritmalara bakıldığında başlangıç durum için beklenilen karın yaklaşık aynı olduğu görülmekte, standart sapmada ise YDY'de elde edilen sonucun SARSA'da elde edilenden daha az olduğu görülmektedir. 3000 iterasyon için koşum sürelerine bakıldığında SHAPE algoritması en az süreye sahiptir. Bu algoritmada tüm envanter düzeyleri incelendiğinden (kümeleme yapılmamakta) ve standart sapması da çok daha az olduğundan, algoritma 15.000 ve 30.000 iterasyon çalıştırılmış, iterasyon sayısı arttıkça sonuçların deterministik modele daha da yaklaştığı görülmüştür.

Çizelge 5.39 : Üçlü ürün grubu için algoritma kıyaslamaları.

Algoritma	İterasyon sayısı	Koşum Süresi	Başlangıç durumunun std. sapması	Başlangıç durumunun beklenen karı	95% güven aralığı Alt sınır	95% güven aralığı Üst sınır	Örnek patika için elde edilen kar
SARSA_50	5000	2,91 saat	7422	230.907	230.701	231.113	208.720
SARSA_100	5000	2,84 saat	7357	224.319	224.115	224.523	223.380
SARSA_karışık	5000	5,33 saat	6492	218.227	218.047	218.407	230.225
YDY_50	5000	2,53 saat	5648	213.123	212.966	213.279	241.100
YDY_100	5000	2,84 saat	6853	236.286	236.096	236.476	212.370
YDY_karışık	5000	5,01 saat	6065	200.195	200.027	200.363	212.370
SHAPE	5000	27 dk.	213	163.916	163.910	163.922	277.572
SHAPE	20.000	4,25 saat	260	150.116	150.112	150.120	256.007
Deterministik	-	-	-	-	-	-	283.995

Üçlü ürün grubu analizlerinde 5000 iterasyon için SHAPE algoritmasının koşum süresi ve standart sapma açısından diğerlerine göre çok daha iyi sonuçlar verdiği görülmüş, bu nedenle SHAPE algoritması 20.000 iterasyon için çalıştırılmıştır. Örnek patika için elde edilen beklenen gelirin deterministik çözümde elde edilen değere çok yakın olduğu görülmüştür.

6. SONUÇLAR

Bu çalışmada perakende sektöründe karşılaşılan bir kalıcı indirim eniyileme problemi ele alınmıştır. Bir hazır giyim perakende firmasına ait satış verileri, veri madenciliği algoritmaları ile analiz edilerek tekil ve çoklu ürün gruplarına karar verilmiştir. Talep tahmininde ekonomi ve pazarlama literatüründe sıklıkla kullanılan kesikli seçim modellerinden MNL Modeli kullanılmış ve ürün satışlarının etkileri fiyat indiriminden dolayı ikame etkisi ve zaman etkisi olmak üzere 2 gruba ayrıştırılmıştır. Çoklu ürün gruplarında ikame faktörünün optimal politika ve gelirler üzerindeki etkisi literatürden elde edindiğimzi bilgilere göre daha önce hiç çalışılmamıştır. Fakat biliniyor ki ürünler ararsındaki korelasyon optimal politikalara karar verirken değinilmesi gereken önemli bir konudur. Tekil ürünlerin optimal politikalarına klasik dinamik programlama yöntemlerinden Değer Yineleme Algoritması ile karar verilmiş, çoklu ürün grupların optimal politikalarına karar verirken ise durum uzayı çok boyutlu ve büyük olduğundan klasik dinamik programlama ile çözüm sağlanamamış, bu nedenle bu gibi problemlerin çözümünde iyi sonuçlar veren Ödüllü Öğrenme algoritmaları kullanılmıştır. Algoritma sonuçları her dönem talebin bilindiği farz edilerek oluşturulan deterministik modelle kıyaslanarak performansı değerlendirilmiştir. Deterministik modelden elde edilen optimal sonuçlar kıyaslama yapılabilmesi açısından geliştirdiğimiz algoritmalarda kullanılmış, ona göre bir değerlendirme yapılmıştır.

Problemin çözümünde Ödüllü Öğrenme algoritmalarından bir politika yineleme algoritması olan SARSA ve Yaklaşık Değer Yineleme algoritmaları kullanılmıştır. Her durumun öğrenilmesi çok fazla sayıda iterasyon gerektirdiğinden ve bu da çok fazla zaman aldığından, bu probleme çözüm olarak kümeleme tekniği uygulanmıştır. Kümeleme Tekniği ile benzer durumlar bir araya getirilmiş, her durumun kendi değeri yerine küme değerleri alınmıştır. Böylece kümelere olan ziyaret sayıları arttığından daha doğru sonuçlar elde edilmiştir. Kümeleme tekniğinde önemli olan küme büyüklüklerinin belirlenmesidir. Küçük kümeler oluşturulduğunda bu kümelere olan ziyaret sayıları daha az olacaktır. Büyük kümeler oluşturulduğunda ise

ziyaret sayıları artacak, ancak standart sapmalarda da artış olacaktır. Bunun için farklı kümeleme düzeylerini bir araya getiren karışık kümeleme tekniği kullanılmıştır. Karışık kümeleme ile beklenildiği gibi kümelere olan ziyaret sayıları artmış, standart sapmalarda düşüş gözlenmiştir. Bir diğer çözüm de durum değerlerinin tahmini için fonksiyon uydurma yöntemi olmuştur. Bu çözüm için SHAPE algoritması kullanılmıştır. SHAPE algoritmasında önemli olan başlangıç fonksiyonlarının iyi olmasıdır. Başlangıç fonksiyonlarının tahmini için SARSA algoritması 1000 iterasyon için çalıştırılmış, her t anında her envanter düzeyi için hangi fiyat kararları verildiği ve ne kadar gelir elde edildiği bilgileri edinilmiş, regresyon ile başlangıç fonksiyonlarına karar verilmiştir. SHAPE algoritması ile bu fonksiyonların değerleri adım adım iyileştirilmiş, sonuç olarak her dönem için oluşturulan parçalı doğrusal fonksiyonlarla her durumun değeri daha hızlı tahmin edilmiştir. Böylece koşum süresinden de büyük kazanç sağlanmıştır.

İkili ve üçlü ürün grupları için algoritma sonuçlarını kıyasladığımızda SARSA ve YDY algoritmaları için karışık kümeleme sonuçları, koşum süresi ve standart sapma açısından tekil kümeleme sonuçlarına göre daha iyi sonuçlar vermiştir. SHAPE algoritmasının ise, bu kriterler açısından, tüm diğer algoritmalara göre daha iyi olduğu görülmüştür.

Çoklu ürün gruplarında tezimizin asıl amaçlarından biri olan ikame ve zaman faktörlerinin optimal politika ve gelir üzerindeki etkileri analiz edilmiştir. Algoritma sonuçlarına göre ikamenin optimal politika üzerinde pozitif etkileri olduğu görülmüştür. İkame ürünlerde, bir ürünün fiyatını düşürdüğümüzde diğer ürünün talebinde artış olduğundan, sistem karı artırmak için fiyatları artırma yönüne gitmiş, dolayısıyla karda artmıştır. İndirimlere yönelik iki analiz yapılmıştır. Birincisi sezon boyunca indirim kısıtının olmaması, ikincisi indirim sayısının sınırlı olmasıdır. İndirim sayısına kısıt konulmadığında, sistem küçük küçük indirimler yaparak karı artırmaya çalışmıştır. Aniden büyük bir indirim yapılmadığından ilk dönemlerde fiyat daha yüksek olmuş ve daha büyük kar elde edilmiştir. Fakat indirim sayısında kısıt olduğunda, sistem elinde envanter kalmaması için daha büyük indirimlere karar vermiştir. Fiyatlar daha düşük olduğundan dolayı da diğer duruma göre daha az kar elde edilmiştir.

Diğer incelenen bir etki ise zaman esnekliğidir. Zamanın talep üzerinde negatif etkisi olduğundan, zaman etkisini kaldırdığımız durumda taleplerde artış olmuştur. Talepler arttığı için sistem yine fiyatları artıma yönüne gitmiş, dolayısıyla beklenen kar artmıştır. Sonuç olarak zamanın optimal politika üzerinde negatif etkisi olduğu görülmüştür.

İkame ve zamanın optimal politika ve gelir üzerindeki etkileri deterministik modelle kıyaslandığında yaklaşık aynı sonuçlar elde edilmiştir. Böylece ÖÖ algoritmalarının büyük durum uzaylarına sahip problemlerde iyi sonuçlar verdiği görülmüştür.

Ürünleri birbirinden bağımsız kabul etmiş olsaydık, yanlış kararlar verip daha düşük gelirler elde edebilirdik. Bu yüzden, ürünler arası korelasyon kalıcı indirim eniyilemesinde dikkate alınması gereken önemli bir konudur.

KAYNAKLAR

Anupindi, R., Dada, M., Gupta, S., 1998: Estimation of consumer demand with stock-out based substitution: An application to vending machine products, *Marketing Science*, Vol. **4**, 406–423.

Aydin, G., Portesus, E. L., 2005: Joint Inventory and Pricing Decisions for an Assortment, *Working Paper*, Endüstri ve Yöneylem Mühendisliği Bölümü, Michigan Üniversitesi, Ann Arbor, MI.

Bell, D.R., Chiang, J., Padmanabhan, V., 1999: The Decomposition of Promotional Response: An Emprical Generalization, *Marketing Science*, Vol. **18**, No. 4, 504-526.

Ben-Akiva, M.E., Lerman, S.R., 1985: *Discrete Choice Analysis: Theory and Application to Travel Demand*, MIT Press

Bertsekas, D.D., Tsitsiklis, J.N., 1996: *Neuro Dynamic Programming*, Athena Scientific, Belmont, MA, USA

Bertsekas, D.D., Tsitsiklis, J.N., 1989: Adaptive aggregation methods for finite horizon dynamic programming, *IEEE Transactions Automatic Control*, Vol. **34**, No. 6, 589-598

Bitran, G., Caldentey, R., 2003: Dynamic Pricing in the Presence of Inventory Considerations: Research Overview, Current Practices, and Future Directions, *Management Science*, Vol. **5, No.** 3, 203-229

Bitran, G., Caldentey, R., Mondschein, S.V., 1998: Coordinating Clearance Markdown Sales of Seasonal Products in Retail Chains, *Operations Research*, Vol. **46**, 609–624

Bucklin, R.E., Gupta, S., 1992: Brand Choice, Purchase Incidence, and Segmentation: An Integrated Approach, *Journal of Marketing Research*, **29**, 201-215.

Bucklin, R.E., Gupta, S., Siddarth, S., 1998: Determining Segmentation in Sales Response Across Consumer Purchase Behaviors, *Journal of Marketing Research*, **35**, 189-197

Chan, L. M. A., Max Shen, Z.J., Simchi-Levi, D., Svann, J., 2004: Coordination of pricing and inventory decisions: A survey and classification.

Chan, T., Narasimhan, C., Zhang, Q., 2008: Decomposing Promotional Effects with a Dynamic Structural Model of Flexible Consumption, *Journal of Marketing Research*, Vol. **45**, 487-498

Chintagunta, P.K., Haldar, S., 1998: Investigating Purchase Timing Behavior in Two Related Product Categories, *Journal of Marketing Research*, **35** (1), 43-53.

Cooper, L.G., Nakanishi, M., 1988: *Market Share Analysis: Evaluating competitive marketing effectiveness*, Kluwer Academic Publishers

Elmaghraby, W., Keskinocak, P., 2003: Dynamic Pricing in the Presence of Inventory Considerations: Research Overview, Current Practices, and Future Directions, *Management Science*, Vol. **49**, No. 10, 1287-1309

Feng, Y., Gallego, G., 1995: Optimal starting times for end-of-season sales and optimal stopping times for promotional fares, *Management Science*,Vol. **41**, 1371–1391

Fisher, M.L., Hammond, J.H., Obermeyer, W. R., Raman, A., 1994: Making supply meet demand in an uncertain world, *Harvard Business Review*, Vol. **72**, No. 3, 83–93.

Gallego, G., van Ryzin, G., 1994: Optimal dynamic pricing of inventories with stochastic demand over finite horizons, *Management Science*, Vol. **40,** No. 8, 999-1020.

Gosavi, A., 2003: *Simulation-based optimization: Parametric Optimization Techniques and Reinforcement Learning*, Kluwer Academic Publishers, Hollanda

Guadagni, P., Little, J.D.C., 1983: A Logit Model of Brand Choice Calibrated on Scanner Data, *Marketing Science*, **2**(3), 203-238.

Gupta, S., 1988: Impact of Sales Promotions on When, What, and How Much to Buy, *Journal of Marketing Research*, **25**, 342-355.

Gupta, D., Hill, A.V., Chameeva, T., 2006: A Pricing Model for Clearing End-of-Season Retail Inventory, *European Journal of Operational Research*, Vol. **170**, 518-540

Harmon, M.E., Harmon, S.S., 2000: Reinforcement Learning: A Tutorial, http://www.nbu.bg/cogs/events/2000/Readings/Petrov/rltutorial.pdf

Heching, A., Gallego,G., van Ryzin, G., 2002: Mark-down Pricing: An Empirical Analysis of Policies and Revenue Potential at One Apparel Retailer, *Journal of Pricing and Revenue Management*, Vol. **1**, 139-160

Kaelbling, L.P., Littman, M.L., Moore, A.W., 1996: Reinforcement Learning: A Survey, *Journal of Artificial Intelligence Research*, Vol. **4**, 237- 285

Kök, A.G., Fisher, M.L., Vaidyanathan, R., 2006: Assortment Planning: Review of Literature and Industry Practice

Kök, A.G., Fischer, M.L., 2007: Demand Estimation and Assortment Optimization Under Substitution: Methodology and Application, *Operations Research*, **55**, No.6, 1001-1021

Krishnamurthi, L., Raj, S.P., 1988: A Model of Brand Choice and Purchase Quantity Price Sensitivities, *Marketing Science*, Vol. **7**, No. 1

Lazear, E. P., 1986: Retail pricing and clearance sales, *Amer. Econom. Rev*., Vol. **76**, 14–32,

Lee, T., Hersh, M., 1993: A model for dynamic airline seat inventory control with multiple seat bookings, *Transportation Science*, Vol. **27**, No. 3, 252–265

Little, J.D.C., Guadagni, P.M., 1986: A Logit Model of Brand Choice Calibrated on Scanner Data, *Marketing Science*, Vol. **2,** No. 3, 203-238

Luus, R., 2000: *Iterative Dynamic Programming* , Chapman & Hall/CRC, New York

Maddah, B., Bish, E.K., 2007: Joint pricing, Assortment, and Inventory Decisions for a Retailer's Product Line, Wiley Interscience

Maglaras, C., Meissner, J., 2006: Dynamic Pricing Strategies for Multi-product Revenue Management Problems, *Manufacturing & Service Operations Management*, Vol. **8**, No. 2, 136–148

Mantrala, M.K., Rao, S., 2001: A Decision support system that helps retailers decide order quantities and markdowns for fashion goods, *Interfaces*, Vol. **31**, No.3, 146-165

McFadden, D., 1974: *Conditional Logit Analysis of Qualitative Choice Behaviour*, Frontiers in Econometrics, Paul Zarembka, New York: Academic Press, Inc.

Monroe, K. B., Bitta, A. J. D., 1978: Models for pricing decisions, *Journal of Marketing Research*, Vol. **15**, 413–428.

Moriarty, D.E., Schultz, A.C., Grefenstette, J.J., 1999: Evolutionary Algorithms for Reinforcement Learning, *Journal of Artificial Intelligence Research*, Vol. **11**, 241-276

Neslin, S.A., Henderson, C., Quelch, J., 1985: Consumer Promotions and the Acceleration of Product Purchases, *Marketing Science*, **4**(2), 147-65.

Neslin, S.A., 2002: Sales Promotion, Relevant Knowledge Series, *Marketing Science* Institute, Cambridge, MA, **11**, 28, 62-63

Neslin, S.A., Ailawadi, K.L., Gedenk, K., Lutzky, C., 2007: Decomposition of Sales Impact of Promotion-Induced Stockpiling, *Journal of Marketing Research*, Vol. **44**, 450-467.

Netessine, S., Rudi, N., 2003: Centralized and competitive inventory models with demand substitution, *Operations Research*, **51**, 329–335.

Pashigian, B.P., 1988: Demand uncertainty and sales: A study of markdown pricing, *American Economy Review*, Vol. **78**, No. 5, 936-953

Patel, J.K., Read, C.B., 1982 : *Handbook of the Normal Distribution*, NewYork, Dekker

Phillips, R.L., 2005: *Pricing and Revenue Optimization*, Stanford University Press, Stanford, California

Powell, W.B., 2007: *Approximate Dynamic Programming: Solving the Curses of Dimensionality*, A John Wiley & Sons, Inc., New Jersey

Powell, W.B., 2010: Merging AI and OR to solve high dimensional stochastic optimization problems using approximate dynamic programming, *INFORMS Journal on Computing*, Vol. **22**, No. 1, 2-17

Puterman, M.L., 1994: *Markov Decision Processes: Discrete stochastic Dynamic Programming*, John Wiley & Sons, Inc.

Rao, R. C., 1984: Pricing research in marketing: The state of the art, *J. Bus.*,Vol. **57**, 39–64.

Rajan, A., Rakesh, R., S., 1992: Dynamic pricing and ordering decisions by a monopolist, *Management Science*, Vol. **38**, 240–262

Rajaram, K., Tang, C.S., 2001: The impact of product substitution on retail merchandising, *European Journal of Operation Research*, **135**, 582-601

Reda, S., 2003: Despite Early Positive Results, Retailers Haven't Jumped on Analytics Bandwagon, *Stores (KhiMetrics, Inc.)*, Vol. **85**, No.3, 203-238

Robbins, H., Monro, S., 1951: A stochastic approximation Method, *Ann. Math. Statistics*, **22**, 400-407

Rogers, D.F., Plante, R.D., Wong, R.T., Evans, J.R., 1991: Agggregation and disaggregation techniques and methodolgy in optimization, *Operations Research*, Vol. **39**, No.4, 553-582

Ryzin, G., Mahajan, S., 1999: On the Relationship between Inventory Costs and Variety Benefits in retail Assortments, *Management Science*, Vol. **45**, No.11, 1496-1509

Smith, S., Achabal, D., 1998: Clearance pricing and inventory policies for retail chains, *Management Science*, Vol. **44**, 285–300

Smith, S.A., Agrawal, N., 2000: Management of Multi-item Retail Inventory Systems with Demand Substitution, *Operations Research*, Vol. **48**, No.1, 050-064

Sutton, R.S., Barto, A.G., 1998: *Reinforcement Learning: An Introduction*, MIT Press, Cambridge

Train, K., 2003: *Discrete Choice Methods with Simulation*, Cambridge University Press

Vinod, B., 2004: Retail revenue management and the new paradigm of merchandise optimisation, *Journal of Revenue and Pricing Management*, Vol. **3**, No. 4

Yücel, E., Karaesmen, F., Salman, F.S., Türkay, M., 2008: Optimizing product assortment under customer-driven demand substitution, *European Journal of Operational Research,* Article in Press

Whatkins, C.J.C.H., 1989: Learning From Delayed Rewards, *PhD Thesis,* Cambridge

Zhao,W., Zheng, Y.S., 2000: Optimal dynamic pricing for perishable assets with nonhomogeneous demand, *Management Science*, Vol. **46**, No. 3, 375–388

ÖZGEÇMİŞ

Ad Soyad: Özlem COŞGUN

Doğum Yeri ve Tarihi: Kırıkkale, 1980

Lisans Üniversite: İstanbul Kültür Üniversitesi, Endüstri Mühendisliği (Burslu), 2004
İstanbul Kültür Üniversitesi, Bilgisayar Mühendisliği (ÇAP), (Burslu), 2004

Yüksek Lisans Üniversite : İstanbul Teknik Üniversitesi, Endüstri Mühendisliği, 2006

Yayın Listesi:

Ulusal sempozyumlarda basılan bildiriler:

1. **İnce, Ö**., Ayağ, Z., Özdemir, R.G., 2005: Montaj Hatlarında Ara Stok Alanı Kapasitesi Belirleme, YAEM 2005, Ulusal Yöneylem Araştırması ve Endüstri Mühendisliği Kongresi, Koç Üniversitesi, İstanbul
2. Özdemir, R.G., Ayağ, Z., Kula, U., **İnce, Ö.,** 2006: Hedef Programlama ile Tedarik Zinciri Yönetiminde bir Vaka Çalışması, YAEM 2006, Ulusal Yöneylem Araştırması ve Endüstri Mühendisliği Kongresi, Kocaeli Üniversitesi, 181-185, Kocaeli, Turkey
3. **İnce, Ö.,** Gültaş, İ., 2006: Makina Seçimi Problemine Bulanık AHP Yaklaşımı", ÜAS, İstanbul Kültür Üniversitesi, İstanbul
4. **İnce, Ö**., 2008: A Case Study on an Office Process for Reducing Lead Time by Applying Lean Tools, YAEM 2008, Ulusal Yöneylem Araştırması ve Endüstri Mühendisliği Kongresi, İstanbul
5. **İnce, Ö.,** Kula, U., Demiriz, A., 2008: Talep Değişkenliğinin Kalıcı Fiyat İndirimi Kararlarına Etkisi, YAEM 2008, Ulusal Yöneylem Araştırması ve Endüstri Mühendisliği Kongresi, İstanbul
6. **Coşgun, Ö**., Kula, U., Demiriz, A., 2010: Bir Hazır Giyim Perakende Zincirinde Rassal Talep Altında Kalıcı İndirim Politikalarının Belirlenmesi , YAEM 2010, Ulusal Yöneylem Araştırması ve Endüstri Mühendisliği Kongresi, İstanbul

Uluslararası sempozyumlarda basılan bildiriler:

1. Ayağ, Z., Özdemir, R.G., **İnce, Ö.,** 2005: A Simulation Model for Performance Evaluation in the Supply Chain, Uluslararsı Lojistik ve Tedarik Zinciri Kongresi, İstanbul, Türkiye

2. **İnce, Ö.,** 2007: Selection of an ERP Software System by using Fuzzy VIKOR, JCIS / FTT 2007, 1305-1311, Utah, ABD

3. **İnce, Ö.,** Serarslan, M.N., 2007: Deciding on the Optimal Timing and the Investment Policy for a Subway Line by Using Probabilistic Dynamic Programming, CIE 2007 International Conference on Computers and Industrial Engineering, İskenderiye, Mısır

4. **İnce, Ö.,** Behret, H., Çevikcan, E., Yenisey, M.M., 2007: TSP Based Flowshop Scheduling via Nearest Neighbour Algorithm, CEFIS' 07, IFAC Sempozyumu, İstanbul, Türkiye

5. Ekinci, Y., **İnce, Ö**., 2007: Supplier Evaluation and Selection by Using Data Envelopment Analysis and Goal Programming, Uluslararası Lojistik ve Tedarik Zinciri Kongresi, İstanbul, Türkiye

6. **İnce, Ö.,** Ünal, A., Yüksek, K., 2007: Data Mining Algorithms on the Web: An Application for Marketing Campaigns, CIE 2007 International Conference on Computers and Industrial Engineering, İskenderiye, Mısır

7. **İnce, Ö**., 2008: A Simulation Study on an Office Process by Applying Lean Tools, CIE 2008 International Conference on Computers and Industrial Engineering, 791-794, Vol 1, Pekin, Çin

8. **Coşgun, Ö.,** Kula, U., 2010: Markdown Optimization in a Retail Chain under Demand Substitution, EURO 24, Lizbon, Portekiz