

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL

**COMPARATIVE STUDY OF FEDERATED LEARNING FOR
CREDIT RISK ASSESSMENT AND FAIRNESS EVALUATION**

M.Sc. THESIS

Mustafa AKTAŞ

Department of Computer Sciences

Computer Sciences Programme

JULY 2025

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL

**COMPARATIVE STUDY OF FEDERATED LEARNING FOR
CREDIT RISK ASSESSMENT AND FAIRNESS EVALUATION**

M.Sc. THESIS

**Mustafa AKTAŞ
(704221001)**

Department of Computer Sciences

Computer Sciences Programme

**Thesis Advisor: Prof. Dr. Behçet Uğur TÖREYİN
Co-advisor: Dr. Ruşen AKKUŞ HALEPMOLLASI**

JULY 2025

**FEDERE ÖĞRENMENİN KARŞILAŞTIRMALI ÇALIŞMASI:
KREDİ RİSK DEĞERLENDİRMESİ VE ADALET ÖLÇÜMÜ**

YÜKSEK LİSANS TEZİ

**Mustafa AKTAŞ
(704221001)**

Bilgisayar Bilimleri Anabilim Dalı

Bilgisayar Bilimleri Programı

**Tez Danışmanı: Prof. Dr. Behçet Uğur TÖREYİN
Eş Danışman: Dr. Ruşen AKKUŞ HALEPMOLLASI**

TEMMUZ 2025

Mustafa AKTAŞ, a M.Sc. student of ITU Graduate School student ID 704221001 successfully defended the thesis entitled “COMPARATIVE STUDY OF FEDERATED LEARNING FOR CREDIT RISK ASSESSMENT AND FAIRNESS EVALUATION”, which he/she prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor : Prof. Dr. Behçet Uğur TÖREYİN
Istanbul Technical University

Co-advisor : Dr. Ruşen AKKUŞ HALEPMOLLASI
TÜBİTAK BİLGEM

Jury Members : Doç. Dr. Yusuf YASLAN
Istanbul Technical University

Doç. Dr. Sefer BADAY
Istanbul Technical University

Doç. Dr. Ahmet GÜNEŞ
Gebze Technical University

Date of Submission : 30 May 2025

Date of Defense : 18 July 2025





To my family,



FOREWORD

Foremost, I extend my deeply gratitude to my advisor, Prof. Dr. Behçet Uğur TÖREYİN, for his extensive knowledge and valuable guidance throughout my thesis study.

I owe a special debt of gratitude to my advisor, Dr. Ruşen AKKUŞ HALEPMOLLASI, whose unwavering support and guidance extended far beyond the thesis process into my professional life and personal development. Her contributions to both my academic and personal growth have been critical in making this study possible.

I would also like to thank all the professors whose courses I have attended and who contributed to the completion of this thesis, my colleagues for their support, and my family for their constant encouragement. Finally, I am grateful to the open-source communities whose projects provided essential resources for this research.

July 2025

Mustafa AKTAŞ
Data Scientist

TABLE OF CONTENTS

	<u>Page</u>
FOREWORD	ix
TABLE OF CONTENTS	xii
ABBREVIATIONS	xiii
SYMBOLS	xv
LIST OF TABLES	xvii
LIST OF FIGURES	xix
SUMMARY	xxi
ÖZET	xxiii
1. INTRODUCTION	1
2. LITERATURE REVIEW	5
2.1 Central Machine Learning in Finance.....	5
2.2 Federated Learning in Finance	7
2.3 Fairness in Federated Learning	9
3. DATASETS & EXPLORATION	13
3.1 Default of Credit Card Clients Dataset.....	13
3.1.1 Data analysis of DCCC	14
3.2 Home Credit Default Risk Dataset	18
3.2.1 Data analysis of HCDR	21
3.3 Home Equity Dataset	23
3.3.1 Data analysis of HMEQ	24
3.4 Data Partitioning for Federated Learning	26
3.5 Synthetic Noise Generation for Fairness Studies	29
4. METHODOLOGY	31
4.1 Machine Learning Algorithms	31
4.1.1 Logistic Regression	31
4.1.2 Multi Layer Perceptron.....	32
4.1.3 Support Vector Machine	32
4.1.4 eXtreme Gradient Boosting	32
4.1.5 Random Forest.....	32
4.2 Data Sampling Techniques	33
4.2.1 Synthetic Minority Over-sampling.....	33
4.2.2 Random Under Sampling	33
4.3 Feature Reduction Techniques.....	34
4.3.1 Principal Component Analysis.....	34
4.3.2 Select k-Best	34
4.4 Federated Learning.....	34
4.5 Aggregation Techniques.....	36
4.5.1 Federated Averaging	36
4.5.2 F1 Score Weighted Aggregation	37
4.6 Fairness in Machine Learning	38
4.6.1 Group fairness	39
4.6.2 Individual fairness	39
4.6.3 Fairness mitigation techniques	40
4.6.4 Fairness in federated learning: FairFed approach.....	41
4.7 Evaluation Metrics	43
4.7.1 Performance metrics for machine learning	43
4.7.2 Measuring fairness and bias	44
4.7.2.1 Statistical Parity	45
4.7.2.2 Statistical Parity Difference	45
4.7.2.3 Equal Opportunity Difference	45
4.7.2.4 Equalized Odds.....	45
4.7.2.5 Disparate Impact	46

4.7.2.6 Predictive Parity	46
4.8 Implementation Framework	46
5. EXPERIMENTS	47
5.1 Federated Learning Comparative Study For Credit Risk Assessment	47
5.1.1 System architecture.....	48
5.1.2 Train test configuration	49
5.2 Federated Learning Fairness In Finance: An Application To Credit Risk Assessment	50
5.2.1 System architecture.....	50
6. RESULTS & DISCUSSION	53
6.1 Federated Learning Comparative Experiment Results.....	54
6.1.1 Overall performance review: Top models summary	54
6.1.2 Comparison of centralized and federated learning approaches....	57
6.1.3 Analysis of aggregation methods: Comparison of FedAvg and FedF1	61
6.1.4 Impact of data preprocessing techniques on performance.....	64
6.2 Fairness in Federated Learning Experiment	71
7. CONCLUSION	77
REFERENCES	79
APPENDICES	87
Appendix A: Model Performance on DCCC Dataset	89
Appendix B: Model Performance on HCDR Dataset	91
Appendix C: Model Performance on HMEQ Dataset.....	93
Appendix D: Application Architecture and Coding Details	95
D.1 Project Structure and Dependencies	95
D.2 Development of Custom Aggregation Strategies	95
D.2.1 Implementing the FedAvg strategy	95
D.2.2 Implementing the FedF1 strategy	97
D.2.3 Implementing the FairFed strategy	97
D.3 Integration with Different Algorithms and Model Parameters	99
D.3.1 Natively compatible models.....	100
D.3.2 Tree based models	100
CURRICULUM VITAE	103

ABBREVIATIONS

CML	: Centralized Machine Learning
DCCC	: Default of Credit Card Clients
EOD	: Equal Opportunity Difference
FairFed	: Fair Federated Learning
FedAvg	: Federated Averaging
FedF1	: F1 Score Weighted Aggregation
FL	: Federated Learning
HCDR	: Home Credit Default Risk
HMEQ	: Home Equity
ML	: Machine Learning



SYMBOLS

$\mathbf{D}$	: Dataset
$\mathbf{F1}_t$	: Sum of all clients' F1 scores in round t
$\mathbf{f1}_k$	: Client k 's F1 score
$\mathbf{K}$	: Number of clients
$\mathbf{k}$	: Current client
$\mathbf{n}$	: Total number of data points in a dataset
$\mathbf{n}_k$	: Data points in client k 's dataset
$\mathbf{w}_t^k$	: Model parameters of client k in round t
$\oplus$	: Privileged class



LIST OF TABLES

	<u>Page</u>
Table 3.1 : Label distributions according to data sets.	13
Table 3.2 : Null values distributions of DCCC.....	15
Table 3.3 : Null values distributions of HMEQ	24
Table 4.1 : Summary of datasets used in the study in terms of fairness. ...	39
Table 4.2 : Confusion matrix definition.....	43
Table 4.3 : Evaluation metrics and their formulas.	44
Table 5.1 : Sex distribution rates in each client.....	51
Table 5.2 : Different approaches for fairness experiments.	52
Table 6.1 : Best performance results and models by dataset	55
Table 6.2 : EOD scores for different approaches in fairness experiments....	71
Table 6.3 : Scores for FedAvg, FedF1 and fairness experiments with LR ...	73
Table A.1 : Comparative experimental performance on the DCCC dataset .	90
Table B.1 : Comparative experimental performance on the HCDR dataset.	92
Table C.1 : Comparative experimental performance on the HMEQ dataset	94



LIST OF FIGURES

	<u>Page</u>
Figure 3.1 : Distribution of default payment status.....	15
Figure 3.2 : Education status among clients.....	16
Figure 3.3 : Marriage status among clients	16
Figure 3.4 : Sex status among clients	17
Figure 3.5 : Heatmap of features	18
Figure 3.6 : Interrelated structure of files with each other. (Source: [1]) ...	19
Figure 3.7 : Distribution of loan payment status.....	22
Figure 3.8 : Heatmap of features	23
Figure 3.9 : Distribution of loan payment status.....	25
Figure 3.10 : Loan reason status among clients.....	25
Figure 3.11 : Loan reason status among clients.....	26
Figure 3.12 : Heatmap of features	27
Figure 3.13 : Label distribution and percentage after using <i>DirichletPartitioner</i> . (Green as label 1, while red as label 0.).....	28
Figure 3.14 : Kernel Density Estimation distributions of the synthetic Interest attribute by gender and payment status.	30
Figure 4.1 : Federated Learning system architecture.....	35
Figure 4.2 : Federated Averaging Algorithm [2]	37
Figure 4.3 : FedF1 server and client update procedures	38
Figure 5.1 : Illustration of the proposed experimental setup.	48
Figure 5.2 : Illustration of the proposed FL fairness experimental setup. ...	52
Figure 6.2 : Comparison of CML, FedAvg, and FedF1 on the DCCC (purple), HCDR (red), and HMEQ (green) datasets using metrics: (a) Precision, (b) Recall, (c) F1-Score, and (d) Accuracy.	59
Figure 6.3 : Performance comparison of FedAvg and FedF1 methods according to metrics: (a) Precision, (b) Recall, (c) F1 and (d) Accuracy	62
Figure 6.4 : Percentage impact of preprocessing techniques on metrics: (a) Precision, (b) Recall, (c) F1 and (d) Accuracy (Relative to the baseline model)	66
Figure 6.1 : Heatmap of Best Performance Results and Models According to Dataset	75



COMPARATIVE STUDY OF FEDERATED LEARNING FOR CREDIT RISK ASSESSMENT AND FAIRNESS EVALUATION

SUMMARY

In this study, we proposed a horizontal federated learning based risk model study for credit card default prediction for data privacy requirements and fairness issues. In finance, healthcare and IoT fields where data collection on central servers is becoming increasingly difficult, information sharing is provided by sharing model parameters while user sensitive data remains on the local device. In addition, thanks to federated learning, communication load is reduced by sharing only model parameters instead of sharing all data. We used three different datasets in our studies. These are Default of Credit Card Clients (DCCC) dataset containing 30,000 samples, 23 features and 22.1% default rate; Home Credit Default Risk (HCDR) dataset containing 356,255 samples, 122 features and 8.1% default rate; and Home Equity (HMEQ) dataset containing 5,960 samples, 12 features and 19.9% default rate.

In the preprocessing phase of the datasets; scaling with StandardScaler and MinMaxScaler, categorical coding with One-Hot Encoding, SMOTE and RandomUnderSampler for imbalance, PCA and Select-kBest for feature reduction were applied. In addition, five machine learning algorithms were compared for our experiments: Logistic Regression, Multilayer Perceptron, SVM, XGBoost and Random Forest. We built the Federated Learning infrastructure with Flower framework and created five clients with non-IID distribution based on α parameter using DirichletPartitioner in the simulation environment. Each client data was divided into 70% training, 10% validation and 20% test subsets and local preprocessing was performed.

We designed 2 experiments in our study; performance comparison and fairness analysis. In our first experiment, we conducted experiments with different combinations of model, data sampling and feature reduction methods for each data set. This means that a total of 405 different experiments were conducted for three data sets, and each experiment was repeated 10 times to increase the consistency of the experiments and the results were averaged. Here, we tested the performance of the F1 Score Weighted Aggregation method, which we developed as an alternative to FedAvg. According to the results we obtained, aggregation methods generally achieved good or similar results compared to the central method and at the same time protected data privacy. In addition, FedF1 gave good results that could compete with FedAvg and in some cases, it performed better, allowing more consistent results to be produced with lower variance.

For our fairness analysis experiment, we created the gender-dependent "Interest" feature on DCCC. Accordingly, we distributed the data as non-IID among five

clients with different gender ratios. In our fairness experiments, we used three different scenarios to make the results easier to understand; Local, Central and FairFed. In these scenarios, the deviations from Local to Central and FairFed with the Equal Opportunity Difference (EOD) metric were 0.1444, 0.1332 and 0.0283, indicating an increase in fairness of over 80%.



FEDERE ÖĞRENMENİN KARŞILAŞTIRMALI ÇALIŞMASI: KREDİ RİSK DEĞERLENDİRMESİ VE ADALET ÖLÇÜMÜ

ÖZET

Son yıllarda veri gizliliği ve güvenliği gereksinimlerinin ve farkındalığın artmasıyla, verilerin merkezi sunucularda toplanıp işlenmesi giderek zorlayıcı hale gelmiştir. Bu konuda hem kullanıcılar daha bilinçli hale gelmiştir hem de devletler kanunlar ile vatandaşlarının kişisel verilerini koruyucu önlemler almıştır. Bu noktada federe öğrenme yaklaşımı, yerel cihazlarda kullanıcıdan toplanan hassas verilerin merkezi bir sunucuya veya günümüz teknolojisi ile bir bulut sunucusuna aktarılmasına gerek kalmadan makine öğrenmesi modelleri geliştirilmesine olanak sağlar. Bu yaklaşım özellikle sağlık, finans ve IoT gibi alanlarda gelecek vaad etmektedir. Böylece kullanıcının hassas verilerinin sahibinin kontrolünden çıkmasını engellerken makine öğrenimi modellerinin geliştirilmesine imkân sağlanmış olur. Buna ek olarak giderek maliyetli hale gelen iletişim yükünü tüm veriler yerine sadece eğitilmiş makine öğrenmesi model parametrelerini merkezi sunucu ile paylaşarak azaltmayı hedefler. Makine öğrenmesinin yenilikçi bir yaklaşımı olan federe öğrenmesi günlük hayat sorunlarımıza çözüm getirirken bir diğer yaşanan sorun ise bu modellerdeki adalet sorunudur. İdeal bir durumda makine öğrenmesi sistemlerinin aldığı kararların çeşitli demografik gruplar için adaletli ve tarafsız olması gerekmektedir. Bir makine öğrenmesi modelinin ne kadar adil olduğunu ise "equal opportunity difference" (EOD) gibi metrikler ile cinsiyet, ırk veya yaş gibi hassas öznitelikler üzerinde belirliyoruz. Adil olmayan bir modelin ürettiği sonuçlar kurumların etnik ve yasak sorunlar ile yüzleşmesine sebebiyet verebilir ve uzun vadede kullanıcılarının güvenini sarsabilir. Buna örnek olarak bir banka müşterilerinin deri renginden bağımsız bir şekilde kredi skorunu tahmin etmelidir veya bir işe alım sürecinde işe alınacak adayın belirlenmesi konusunda cinsiyet bir önyargı oluşturmamalıdır. Bunlara ek olarak finans sektörü genel olarak kredi skoru tahmini, sahtekârlık tespiti ve portföy optimizasyonu gibi uygulamalarda makine öğrenimine yoğun şekilde bel bağlamaktadır ve ödeme geçmişi ve demografik bilgiler gibi son derece hassas verileri model eğitiminde kullanılmaktadır. Son yıllarda yaşanan dijital dönüşüm bu süreçleri hem hızlandırmış hem de dolandırıcılık araçlarının kolayca erişilebilir olması nedeniyle yeni güvenlik ve adalet gereksinimlerini gündeme getirmiştir. Yaptığımız bu çalışmada ise müşterilerin ham verilerini paylaşmadan kredi kartı temerrütlerini tahmin eden bir federated learning tabanlı risk modeli geliştirdik. Geliştirilen modeller merkezi makine öğrenimi yöntemleri ile FedAvg ve geliştirdiğimiz FedF1 federe yaklaşımları arasında performans karşılaştırması yapıldı. Buna ek olarak FairFed yaklaşımını ve EOD metriğini kullanarak adalet performansını detaylı biçimde analiz ettik.

Çalışmamızda yaptığımız deneylerin hem sonuçlarını hem de tutarlılığını daha iyi gösterebilmek adına üç farklı veriseti kullanılmıştır; Default of Credit Card Clients (DCCC), Home Credit Default Risk (HCDR) ve Home Equity (HMEQ). DCCC veriseti UCI ML Repository’den elde edilmiştir, Nisan–Eylül 2005 dönemine ait 30.000 kredi kartı müşterisinin geri ödeme kayıtlarını içerir. Toplam 23 bağımsız öznitelik bulunurken 5 tanesi demografik hassas bilgiler içerir (yaş, cinsiyet, eğitim, medeni durum vb.). Veriseti ikili olarak kodlanmış olup, %22,12 oranında temerrüt (1) ve %77,88 oranında sorunsuz geri ödeme (0) vakası barındırır. HCDR ise 307.511 eğitim ve 48.744 test örneği olmak üzere toplam 356.255 kredi başvurusu içerir. Bu veriseti de cinsiyet, eğitim, aile durumu gibi çeşitli hassas bilgiler içermektedir. Hedef değişkeni üzerinden ölçülen temerrüt oranı %8,07’dir ve bu da HCDR’yi en yoğun veri dengesizliği yaşanan veriseti yapmaktadır. Son olarak kullanılan Home Equity (HMEQ) veriseti ise toplamda 5.960 ipotekli konut kredisi kaydından oluşur ve 12 öznitelik barındırır. ancak bu özniteliklerden hiç birinde hassas veri yoktur. Veriseti %19,95 oranında temerrüt (BAD=1) ve %80,05 oranında başarılı ödeme vakası içerir.

Kullandığımız verisetleri için uygulanan ön işleme adımlarında veri ölçekleme (StandardScaler, MinMaxScaler), kategorik değişkenlerin kodlanması için One-Hot Encoding ve dengesiz sınıf dağılımını dengelemek amacıyla SMOTE ile Sentetik Azınlık Aşırı Örnekleme ve RandomUnderSampler (RUS) yöntemleri kullanıldık. Ayrıca boyut indirgeme ve özellik seçimi için Temel Bileşen Analizi (PCA) ve Select k-Best yöntemleriyle veri karmaşıklığı ve öznitelik sayısı azalttık. Kullanılan makine öğrenimi modelleri; parametrik lineer yaklaşımlar sunan Lojistik Regresyon (LR), çok katmanlı yapay sinir ağlarıyla çalışan Çok Katmanlı Algılayıcı (MLP), kernel tabanlı SVM, gradyan artırmalı XGBoost (XGB) ve ansamble tabanlı Rastgele Orman (RF) olmak üzere beş farklı algoritmayı kapsamaktadır.

Federe öğrenme ile merkezi makine öğrenmesi modellerini karşılaştırdığımız deneyimizde Flower framework’ün DirichletPartitioner bileşeni kullanılarak, α parametresi aracılığıyla veri heterojenitesinin ayarlandı ve 5 istemci arasında non-IID veri dağılımı oluşturuldu. Burada amaç, bahsettiğimiz 3 verisetini de mümkün olduğunca gerçek hayat senaryosuna benzetmektir. Her bir istemcinin veri kümesi, yerel gizliliği korumak amacıyla yalnızca o istemcinin verisini kapsayacak şekilde %70 eğitim, %10 doğrulama ve %20 test alt kümelerine ayrıldı. Ayrıca bu adımda ön işleme (ölçekleme, kodlama, örnekleme) işlemleri yalnızca istemci bazında uygulandı. Kurduğumuz bu yapı sayesinde farklı etiket oranlarına sahip gerçek dünya non-IID senaryolarını başarıyla simüle ederek yatay FL’nin temel prensipleri doğrultusunda hem model performansı karşılaştırılmış oldu. Çalışmamızda yaptığımız diğer bir deney olan adalet deneyi için ise DCCC verisetini 5 istemci arasında cinsiyet hassas özniteliği üzerinde önceden belirlediğimiz oranlara göre böldük. Burada amacımız farklı oranlardaki veri dengesizliğinin nasıl adaleti etkileyeceğini görmekti.

Yaptığımız çalışmada federe öğrenme altyapısı, açık kaynaklı Flower framework (flwr) kullanılarak simüle edilmiş ve yatay FL senaryosu, beş istemcinin her birinin kendi verisi üzerinde bağımsız eğitim yapıp model parametrelerini

sunucuya iletmesi esasına dayanacak şekilde uygulanmıştır. Buna göre her turda istemciler yerel modellerini kendi verileri ile eğitip güncel ağırlıkları sunucuya gönderir; sunucu gelen tüm ağırlıkları toplar ve küresel modeli güncelleyen bir agregasyon işlemi uygular. Ardından güncellenen küresel modeli tekrar istemcilere iletir. Bu döngü belirlenen iterasyon sayısı kadar devam eder. Agregasyon adımı en yaygın yöntem olan Federated Averaging (FedAvg), her bir istemcinin yerel ağırlık güncellemelerini, o istemcinin eğitimde kullandığı örnek sayısına orantılı olarak ağırlıklandırıp toplar. Buna ek olarak, dengesiz veri setlerinde performans tutarlılığını artırmak amacıyla bu çalışmada geliştirdiğimiz F1 Score Weighted Aggregation (FedF1) yöntemi, her istemciden dönen ve istemcinin kendi doğrulama veriseti ile elde ettiği F1 skorlarını küresel F1 skoru toplamına oranlayarak küresel modeli oluşturur. Bu sayede, yüksek F1 skoruna sahip istemcilerin model güncelleme sürecindeki etkinliği artırılarak genel performans dengesi iyileştirilmektedir.

Çalışmamızda yaptığımız ilk deney olan performans karşılaştırması deneyinde, üç farklı finansal veri seti (DCCC, HCDR, HMEQ) üzerinde işlenen veriler; dengesizlik sorununu çözmek için SMOTE ve RUS örnekleme yöntemlerine, boyut indirgemeyi sağlamak üzere PCA ve kBest özellik seçimine tabi tutuldu; beş makine öğrenimi algoritması (Lojistik Regresyon, Çok Katmanlı Algılayıcı, Destek Vektör Makineleri, XGBoost, Rastgele Orman) hem merkezi ML (CML) hem de iki federated learning agregasyon yöntemi (FedAvg, FedF1) çerçevesinde karşılaştırıldı. Her veriseti model, ön işleme ve agregasyon kombinasyonu ile 135 deney (toplamda 405) yapıldı. Her bir deney, sonuçların tekrarlanabilirlik ve güvenilirliğini arttırmak için 10’ar kez farklı random seed sayıları ile test edildi. Elde edilen sonuçlar doğruluk, kesinlik, duyarlılık ve F1 metrikleri kullanılarak merkezi ve federe öğrenme yaklaşımlarının performans sonuçları değerlendirildi.

Çalışmamızda yaptığımız ikinci deney olan adalet ölçümü deneyinde ilk olarak DCCC verisetinde adalet etkisini daha net bir biçimde gösterebilmek adına “Interest” özniteliği Gauss dağılımlarından türetilerek ekledik ve bu sayede cinsiyetin model çıktıları üzerindeki etkisi suni olarak daha bariz olmasını sağladık. 5 istemciden her birinin farklı derecede non-IID ve adaletsizlik barındırması adına %20/80’den %70/30’a kadar değişen önceden belirlenmiş cinsiyet oranlarına göre manuel olarak ayırdık. Ancak her istemci yine kendi veriseti ile rastgele train-test ve validation olarak ayırdı. Bu deneyde adalet performansının etkisinin iyileşmesi için kullandığımız FairFed yaklaşımının etkisinin daha iyi görülmesi adına 3 farklı senaryo kullandık; Local, Merkezi ve Federe. Local yaklaşımda her istemci, yalnızca kendi verisiyle model ve cinsiyet adaleti ilişkisini ölçerek eğitim ve değerlendirme yaptı. Merkezi yaklaşımda ise tüm istemcilerin kullandığı training veri seti ortak bir havuzda birleştirildi ancak her istemci kendi yerel verisiyle test yaptı. Son olarak federe öğrenme yaklaşımında ise yine federe öğrenme mantığı izlenildi ancak agregasyon kısmı için FairFed kullanıldı.

Yaptığımız deneyler sonucunda ilk olarak vardığımız bulgu, federe öğrenme yöntemlerinin veri gizliliğini korurken aynı zamanda merkezi makine öğrenmesi yöntemleriyle büyük ölçüde benzer veya bazen daha iyi sonuçlar verdiğidir. Kendi

geliştirdiğimiz FedF1 agregasyon yöntemi FedAvg ile karşılaştırıldığında eşdeğer veya daha yüksek değerler elde ederken model varyansını da en düşük seviyeye indirmiştir. Model varyansının düşük olması FedF1'nın modeller arasında daha tutarlı sonuçlar ürettiğini göstermektedir. Bunun yanında modellere eklenen SMOTE yöntemi genel olarak duyarlılık ve F1 skorunu arttırırken doğruluk ve kesinlik değerlerini düşürmüştür. RUS yöntemi ise veri dengesizliği olduğu durumda denge sağladığından F1 skorunda ve doğrulukta artış sağlamıştır. PCA ve kBest gibi boyut indirgeme yöntemleri ise kimi durumlarda karmaşıklığı azalttığı için performans artışı sağlarken kimi durumlarda veri kaybına yol açtığından performans kaybına sebep olmuştur. Bunun yanında model bazlı karşılaştırmalarda ise genel olarak en başarılı sonuçlara sırasıyla XGB, RF ve LR modellerinin ulaştığı söylenebilir. Adalet analizi deneylerinde ise FairFed yöntemi, Yerel yaklaşımın ortalama EOD değerinin %0.1444'ten federe senaryosunda %0.0283'e indirerek %80'in üzerinde iyileşme sağlamıştır. Bu durum demografik gruplar arası adaletsizliği anlamlı biçimde azaltmaktadır. FairFed ayrıca modelde adaleti sağlamaya çalışırken aynı zaman model performansında da merkezi yaklaşım kadar iyi sonuç vermiştir.

Çalışmamızın özgün katkıları şunlardır;

- Merkezi ML yöntemleriyle rekabet ederken veri gizliliğini koruyan, kredi kartı temerrüt tahmini için yeni bir FL tabanlı yaklaşım geliştirdik.
- Her ML modeli için toplam 405 deneyde üç veri kümesiyle hem merkezi hem de federasyonlu yaklaşımlar altında beş ML algoritmasının kapsamlı bir karşılaştırma yaptık.
- FedAvg'a alternatif olarak geliştirdiğimiz, her turdaki F1 skorlarını ağırlıklandırarak küresel güncellemeyi optimize eden FedF1 algoritmasını sunduk.
- EOD temelli ağırlıklı adalet optimizasyonu sunan FairFed metodolojisiyle model çıktılarında grup adaletini sağladık ve deneylerimizle kapsamlı analiz yaptık.
- Tüm deney altyapısını ve kodlarını GitHub'da açık kaynak olarak erişime sunduk.

1. INTRODUCTION

As awareness and requirements for data privacy and security have increased in recent years, collecting and processing data on centralized servers has become increasingly cumbersome. The federated learning approach, which enables sensitive data accumulated on local devices to be incorporated into model training without being transferred to the cloud or data centers, is leading the way. Thus, in areas such as healthcare, finance and the Internet of Things (IoT), personal data can be prevented from circulating outside the owner's control, while simultaneously developing machine learning models. Academic studies and industrial applications have shown that federated learning offers critical advantages in protecting data privacy [3,4].

Fairness in machine learning systems focuses on ensuring that models produce unbiased decisions across demographic groups. Metrics such as "equal opportunity difference" (EOD) (Section 4.7.2.3) have been developed to measure discrimination based on sensitive characteristics such as gender, race or age. Group fairness approaches aim to equalize the advantageous class acceptance rates of different demographic groups, while individual fairness approaches aim for similar outcomes for individuals with similar characteristics. Unfair decisions put organizations at risk both ethically and legally, and in the long run may also damage the trust of model users. Therefore, ensuring fairness is not only an ethical imperative, but also critical for the sustainability of model performance and the competitiveness of organizations. The financial industry relies heavily on machine learning in applications such as credit scoring, fraud detection and portfolio optimization. This kind of data, which includes income, payment history and demographic information, is highly sensitive. Unfair scoring systems can victimize individuals and pose serious risks to the reputation and legal obligations of financial institutions.

Digital transformation has driven modern financial practices, significantly improving traditional banking systems [5]. While financial consumers increasingly prefer digital systems and credit cards for daily transactions [6], the easy accessibility of tools used for fraudulent credit card transactions raises concerns about the adoption of digital systems [7]. Moreover, increasing complexity in the financial market and the diversity of customer demands pose significant challenges for service providers [8].

Financial risk refers to the possibility that an organization’s financial results may deviate from expected business objectives. These deviations, which are often the result of unforeseen and uncontrollable circumstances, can lead to economic losses or lower than expected earnings. One of the most critical risks in the financial sector is credit card default, which occurs when borrowers are unable to fulfill their repayment obligations. The rapid development of the digital environment makes it easier for users to accumulate debt, further complicating default prediction.

In this study, we propose a federated learning-based risk prediction model that predicts credit card default based on debt and payment data and provides collaborative model training without the need for data sharing. Our model is evaluated in terms of fairness metrics (EOD) with the FairFed method and compared with other federated learning strategies.

In our experiments, we used DCCC [9], HCDR [1] and HMEQ [10] datasets and applied scaling, encoding, SMOTE and RUS sampling techniques, PCA and kBest feature selection methods. Five machine learning algorithms were tested: Logistic Regression (LR), Multilayer Perceptron (MLP), Support Vector Machine (SVM), XGBoost (XGB) and Random Forest (RF). FedAvg and our proposed FedF1 methods are compared to evaluate the federated learning effect, while the Equal Opportunity Difference (EOD) metric is used over the FairFed method for fairness-oriented analysis. The results provide a comparative evaluation of FedAvg and FedF1 in terms of both performance and fairness metrics.

Our contributions are as follows:

- We propose a novel FL approach for credit card default prediction that maintains data privacy while ensuring competes with centralized ML methods.
- We perform an extensive comparison of five ML algorithms under both centralized and federated approaches with three datasets in a total of 405 experiments for each ML model.
- We introduce the F1 Score Weighted Aggregation (FedF1) method that achieves comparable or superior results to the traditional Federated Averaging (FedAvg) method.
- We developed fairer model using FairFed’s fairness perspective and compared it to other federated learning methods in terms of performance and fairness metrics.
- The code implementation of our experiments is publicly available for FL research reproducibility and exploration.

The structure of the thesis is as follows: the studies already done in the literature are explained in Section 2. Then, the datasets we used in our experiments are explained in detail in Section 3. In Section 4, all the technologies and methods we used throughout the thesis work are explained. The steps of the experiments we conducted are explained in Section 5 and the results are explained in Section 6. Finally, we interpreted the results we obtained throughout the thesis and shared our ideas for future studies in Section 7.

2. LITERATURE REVIEW

In this section, innovative approaches from related work are briefly reviewed. First, studies employing centralized machine learning (CML) methods for credit risk assessment are analyzed in Section 2.1; then, research adopting federated learning (FL) techniques instead of centralized models is examined in Section 2.2. This overview lays the groundwork for the comparative analysis presented in the study's first experiment. Finally, in Section 2.3, literature on fairness in federated learning is reviewed for the second part of the research.

2.1 Central Machine Learning in Finance

There are several studies in the literature which demonstrate ML applications for customer segmentation [11], financial risk prediction [12], fraud detection [13] and price anomaly detection [14,15] in financial operations that address growing data complexity. The authors used LR and RF to predict two main financial risks, which include interest rate risk and cash risk. They stated that ML methods are effective in accurately assessing financial risks and emphasized their expanding role in risk management strategies.

Financial fraud detection applications increasingly use ML techniques through ensemble learning methods which include RF and DT. The research by Guo et al. [16] showed how the RF algorithm works for financial transaction fraud detection through a comparison with XGB and DTs. The research used the Kaggle "Credit Card Fraud Detection" dataset [13] which includes 284,807 transactions with only 492 fraud cases. Initially, they achieved an Out-Of-Bag (OOB) score of 0.924 and an AUC of 0.967 using default parameters. With hyperparameter tuning, they improved those metrics to 0.933 for OOB and 0.978 for AUC and emphasized the importance of hyperparameter optimization in ML applications [16].

Modern banking systems depend on financial customer segmentation to enable organizations to provide tailored services while properly managing risks. In a recent study, Qiu and Wang [17] utilized a ML approach to segment credit card customers based on the behavioral data. Their dataset comprises information from approximately 9,000 active credit card users and includes 18 behavioral variables such as balance, shopping frequency, and credit limit. The authors employed the K-means clustering algorithm for its ability to generate interpretable and actionable segments and compared its performance with various clustering techniques including hierarchical clustering, DBSCAN, Birch, and Gaussian mixture models. To evaluate results, they used the Davies-Bouldin Index, Silhouette Score, and Calinski-Harabasz Index metrics. K-means outperformed other methods with a Davies-Bouldin Index of 0.80, a Silhouette Score of 0.87, and a Calinski-Harabasz Index of 5823. By contrast, DBSCAN produced a Calinski-Harabasz Index of 3749 and a Silhouette Score of 0.43, therefore highlighting the limitations of density-based clustering for this dataset [17].

Identifying pricing anomalies is crucial in e-commerce as demonstrated by Walmart Labs. They developed an ML-based anomaly detection system to detect potential pricing inconsistencies in their online platforms [15]. They conducted the analysis using 4.6 million real retail data samples with severe class imbalance. According to their results, RF and GBM significantly outperformed other models with F1 scores of 94.16% and 94.38%, respectively. Additionally, implementing log-based feature transformations significantly improved model performance by handling data skewness [15].

Wahab et al. [9] evaluated the performance of Deep Learning (DL) and ML techniques for credit card default prediction. Authors compared AdaBoost, DT, and Artificial Neural Network (ANN) models using credit card default data [18] from the UCI ML dataset. During the data preprocessing, they eliminated missing data and redundant columns and conducted exploratory data analysis. Their findings indicated that AdaBoost and DT achieved the highest accuracy performance with 82%. They further investigated the impact of DL parameters,

such as the number of hidden layers, activation functions, and optimization techniques on models' performance. They observed that the ANN with 5 hidden layers and Adam optimizer achieved the highest accuracy score [9].

The selection of features plays a vital role in credit card default assessment because the chosen variables determine the final model outputs. The "FDAF-score" feature selection model developed by Zhou et al. [19] integrates Fisher discriminant analysis (FDA) with F-score methodologies. Thus, authors aimed to ensure that feature combinations have maximum discrimination ability, contrasting with single feature selection methods. They evaluated commonly employed models including SVM, ANN, LR, DT, K-Nearest Neighbor (KNN), Linear Discriminant Analysis (LDA), and RF. Their results revealed that FDAF outperformed single feature selection approaches on the Default of Credit Card Clients dataset [18].

These studies highlight the effectiveness of centralized models. However, they rely on the collection of sensitive financial data on a single server, which poses significant data privacy and security challenges. These concerns have motivated the exploration of alternative approaches like Federated Learning, which will be discussed in the next section.

2.2 Federated Learning in Finance

Research in healthcare [20,21], IoT [22], wireless communication networks [23] and energy demand prediction [24] demonstrates FL's ability to improve analytical capabilities while maintaining data privacy and security.

Authors [25] developed an FL-based prototype to investigate the performance of FL-based credit risk assessments and investigate its potential to reduce disparities in risk calculations between financial institutions. The authors highlighted the potential of FL in credit risk assessment, allowing financial institutions to collaboratively train models without sharing raw data, thus preserving privacy. The authors tested the prototype to assess how information sharing affects risk models across different scenarios. The research showed that FL technology

enhances the performance of smaller financial institutions which have restricted internal data resources.

Imteaj and Amini [26] proposed an FL-based application to predict financial issues and constructed a global model that combines locally trained models from distributed agents without sharing sensitive data externally. They performed asynchronous FL to mitigate delays from resource-constrained agents and adapted FedAvg to customize local epochs. Their findings demonstrated that the FL model achieves comparable F1 scores to centralized models and outperforms state-of-the-art FL methods, in particular, when managing resource-limited agents in a distributed setting.

Awosika et al. [7] proposed a new FL-based fraud detection system which unites FL and eXplainable AI (XAI) methods to improve both data privacy and transparency in the financial sector. They employed a Deep Neural Network to detect fraudulent transactions across federated databases, ensuring high accuracy without compromising data confidentiality. Also, they incorporated XAI to enhance model prediction interpretability because this feature enables financial experts to trust and understand fraud detection results better.

Li and Wen [27] proposed an FL-based method for credit risk management across multiple banks and solved the problems of data heterogeneity and privacy in credit risk modeling. They used heterogeneous data from a large joint-stock bank and a small city commercial bank in their experimental framework. The authors show that FL can achieve similar or better performance than centralized modeling while keeping data privacy. They show the potential of FL to improve credit risk assessment for smaller banks by using collective data.

The studies above demonstrate that FL can preserve data privacy in financial applications without a significant trade-off in performance. However, as these models are trained on distributed and often heterogeneous data from diverse demographic groups, a new and critical challenge arises: ensuring the fairness and equity of the resulting global model.

2.3 Fairness in Federated Learning

Khandpur Singh et al. (2021) [28] propose that security requirements and fairness requirements create conflicts when performing federated learning processes. They show that distance-based detection approaches with fixed thresholds have unfair filtering effects, excluding updates from minority groups while protecting against malicious attacks. The security measure functions to stop attacker models while simultaneously stopping the evaluation of updates from clients who have unfair data distribution. The proposed method resolves the unfairness problems by maintaining minority group data representation throughout the federated learning process. The microaggregation-based method reaches 95.11% model accuracy during UCI Adult Income dataset [29] tests. The system demonstrates its capability to preserve security and group fairness (e.g., Equal Opportunity and Statistical Equivalence differences) in federated environments by reducing false negative rates in minority group.

Thi Thu et al. [30] presents a comprehensive experimental study covering the applications of fairness-oriented ML in the credit scoring domain. The authors evaluated six fairness metrics including Statistical Parity, Equal Opportunity, Equalized Odds, Predictive Parity, Predictive Equality, and ABROCA alongside accuracy and balanced-accuracy metrics on six real-world datasets through Bayesian Network and data analysis methods. The results show that AdaFair, in particular, provides the best balance between accuracy and fairness on all datasets.

Yahya H. Ezzeldin et al. [31] proposed a server-side fairness-oriented aggregation mechanism to ensure group fairness in federated learning with their approach called FairFed. Each client calculates the local fairness metric (EOD or SPD) in each federation round after “debiasing” the model updates by itself with any local unfairness correction method it has previously chosen (e.g. reweighting [32], FairBatch [33]). The server-side obtains the global fairness metric from the clients with the secure aggregation protocol and assigns weights to each of them by comparing them with the local fairness metrics of the clients. Accordingly, the

clients whose global model EOD is not very different are weighted more. This actually shows that FairFed is a new aggregation method. This method improves EOD by 93% and 50% on the Adult and COMPAS datasets, respectively, when data heterogeneity is high, while losing only 0.3% in accuracy. Another advantage of FairFed is that it allows clients to use different debiasing methods; flexible implementation is possible on each client, since the server updates the weights based only on local and global fairness metrics.

Wasif et al. [34] examined the trade-offs between privacy, fairness, and accuracy in FL. FL offers the advantage of preserving data privacy while providing decentralized model training; however, when applied to critical domains such as healthcare and finance, a trade-off exists between privacy and fairness. The study examined the fairness effects of privacy-preserving techniques such as Differential Privacy (DP), Homomorphic Encryption (HE), and Secure Multi-Party Computation (SMC). The focus of the study was to determine how different privacy techniques affect fairness in IID and non-IID data distributions. In the experiments, FL aggregation techniques such as q-FedAvg, q-MAML, and Ditto were compared on Credit Card Fraud Detection (CCFD) datasets. The results indicate that the data privacy provided by the DP technique negatively impacted model accuracy in non-privileged group, increasing unfairness. The study also demonstrated that the relationship between privacy and fairness can be counterproductive. For example, optimizations aimed at improving fairness can weaken privacy protection methods. The study’s specific contribution to the financial sector is its examination of the balance between privacy and fairness in CCFD dataset. It has been demonstrated that because financial data generally has a non-IID structure, the DP method can negatively impact fairness.

In summary, the literature demonstrates a clear trajectory from centralized machine learning models in finance towards privacy-preserving federated learning approaches. While FL addresses critical data privacy concerns, recent studies increasingly highlight the challenge of fairness. Existing works have proposed various methods to measure and mitigate bias, such as the FairFed framework, and have begun to analyze the trade-offs between privacy, fairness, and model

accuracy. This study builds upon this foundation by conducting a comprehensive comparative study specifically within the credit risk domain. It not only compares the performance of centralized and federated approaches but also introduces a novel performance-based aggregation method, FedF1, and provides an in-depth fairness analysis using the FairFed methodology on real-world financial datasets.





3. DATASETS & EXPLORATION

In this chapter, we introduce three publicly available credit-risk datasets used to evaluate both CML and FL approaches. We employed the Default of Credit Card Clients (DCCC) (Section 3.1) dataset from the UCI Machine Learning Repository [9,35], Home Credit Default Risk (HCDR) (Section 3.2) dataset published by Home Credit B.V. on Kaggle [1], and Home Equity (HMEQ) (Section 3.3) dataset [10]. Together, these datasets contain more than 350,000 credit card data and a wide range of attributes from demographics to occupation. Each dataset exhibits a different degree of class imbalance (Table 3.1), which we address via targeted sampling techniques. Additionally, this chapter presents the exploratory data analysis (EDA) performed on each dataset (Section 3.1.1, 3.2.1, 3.3.1), describes how we partitioned the data among clients for our FL experiments (Section 3.4), and outlines the method used to introduce artificial unfairness for our fairness evaluations (Section 3.5).

Table 3.1 : Label distributions according to data sets.

	DCCC		HCDR		HMEQ	
Label	0	1	0	1	0	1
Count	23,364	6,636	282,686	24,825	4,771	1,189
Percentage (%)	77.88	22.12	91.93	8.07	80.05	19.95

3.1 Default of Credit Card Clients Dataset

Default of Credit Card Clients (DCCC) dataset from the UCI Machine Learning Repository [35], originally prepared by Yeh and Lien (2009) [9], comprises 30,000 credit card clients' payment records between April to September 2005 from Taiwanese banks. The target column in the dataset is "default.payment.for.next.month", which is binary classified (1 = default, 0 = non-default). An important consideration when working with this dataset is the class imbalance between default and non-default cases. Among these instances,

6,636 (22.12%) are categorized as default cases, while the remaining 23,364 (77.88%) are non-default cases (Table 3.1). In the dataset with 23 independent variables, the first 5 columns contain demographic information (education, gender, age, etc.), the next 6 columns contain past payment information, the next 6 columns contain past invoice amounts and the last 6 columns contain past payment amounts, while the other columns are ID and target columns. The entire dataset is provided as a single CSV file (UCI_Credit_Card.csv).

3.1.1 Data analysis of DCCC

The first intervention before the experiments in the DCCC dataset was to change the names of some columns. Accordingly, the 'default.payment.next.month' column was changed to 'def_pay' and the 'PAY_0' column was changed to 'PAY_1'. Then, in order to be able to process the data, the 'SEX', 'EDUCATION' and 'MARRIAGE' columns were encoded using OneHotEncoder. The remaining numerical data was scaled with StandardScaler. As a result of the feature importance studies, the columns to be used for kBest experiments were determined as 'PAY_1', 'PAY_2', 'PAY_3', 'PAY_4' and 'PAY_5' in order of importance (see Figure 3.5). According to the feature reduction analyses, it was determined that the most optimal value for the PCA value was 12. The rest of the study continued in this manner.

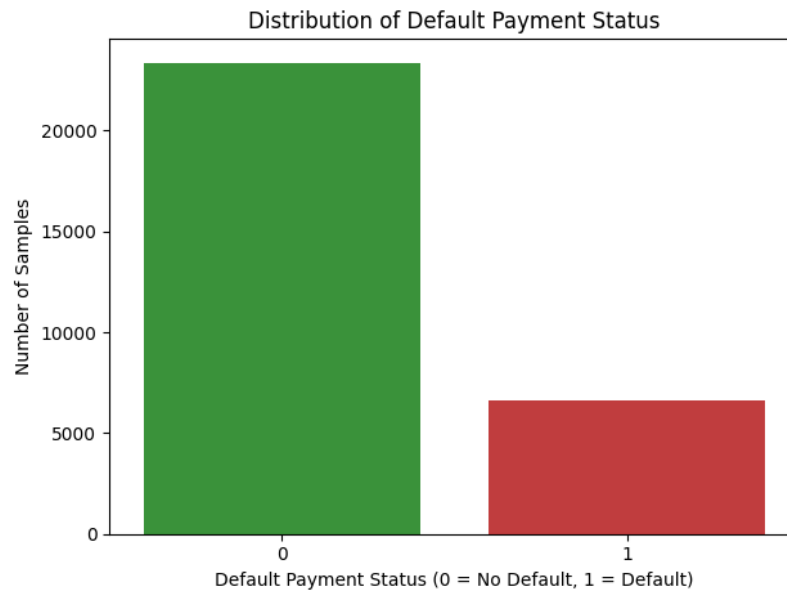
Table 3.2 shows the Non-Null values in the columns. Accordingly, there are no missing data in the dataset.

Figure 3.1 shows the label distribution in the target column in the dataset. As previously shown in Table 3.1, there is an imbalance in the data. Accordingly, 77.88% of the dataset is non-default and 22.12% is in the default state.

Figure 3.2a shows the distribution of users according to their educational background. Here, the numbers correspond to 1=graduate school, 2=university, 3=high school, 4=others, 5=unknown, and 6=unknown. The majority of the data were from university (46.8%), graduate school (35.3%) and high school (16.4%), respectively. In addition, Figure 3.2b shows the default payment status of users

Table 3.2 : Null values distributions of DCCC

#	Column	Non-Null Count	Dtype
0	ID	30000 non-null	int64
1	LIMIT_BAL	30000 non-null	float64
2	SEX	30000 non-null	int64
3	EDUCATION	30000 non-null	int64
4	MARRIAGE	30000 non-null	int64
5	AGE	30000 non-null	int64
6	PAY_1	30000 non-null	int64
7	PAY_2	30000 non-null	int64
8	PAY_3	30000 non-null	int64
9	PAY_4	30000 non-null	int64
10	PAY_5	30000 non-null	int64
11	PAY_6	30000 non-null	int64
12	BILL_AMT1	30000 non-null	float64
13	BILL_AMT2	30000 non-null	float64
14	BILL_AMT3	30000 non-null	float64
15	BILL_AMT4	30000 non-null	float64
16	BILL_AMT5	30000 non-null	float64
17	BILL_AMT6	30000 non-null	float64
18	PAY_AMT1	30000 non-null	float64
19	PAY_AMT2	30000 non-null	float64
20	PAY_AMT3	30000 non-null	float64
21	PAY_AMT4	30000 non-null	float64
22	PAY_AMT5	30000 non-null	float64
23	PAY_AMT6	30000 non-null	float64
24	def_pay	30000 non-null	int64

**Figure 3.1 :** Distribution of default payment status

according to each educational background. The groups with the most difficulty in payment are university, graduate school, and high school, respectively.

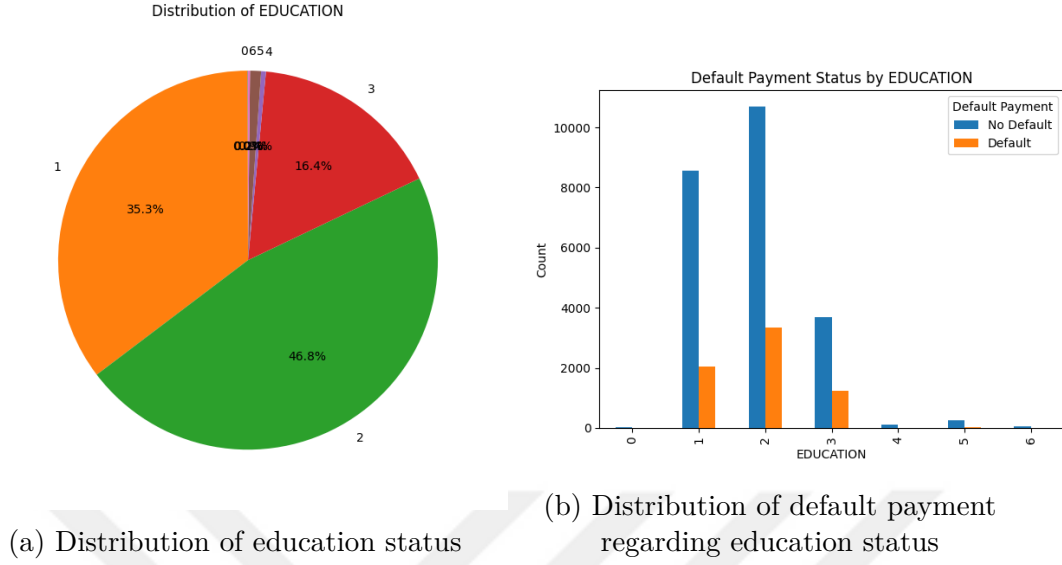


Figure 3.2 : Education status among clients

Figure 3.3a shows the distribution of data according to marital status. Marital status is coded as 1=married, 2=single, 3=others. Accordingly, 53.2% of the customers are single, while 45.5% are married. The rest are others. In addition, as seen in Figure 3.3b, there is no strong connection between marital status and financial difficulty.

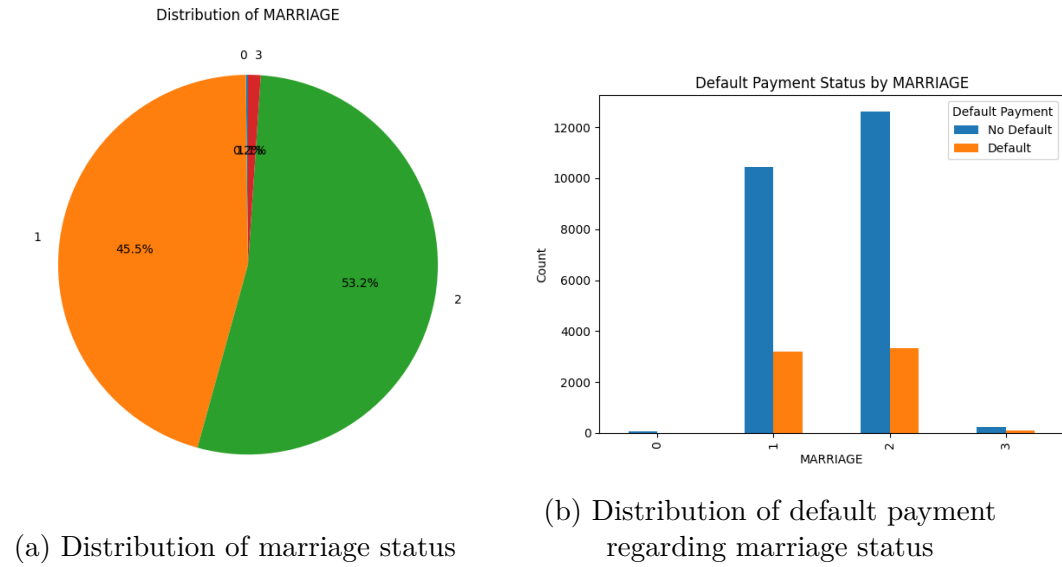


Figure 3.3 : Marriage status among clients

Figure 3.4a shows the data distribution by gender. Accordingly, 60.4% of the customers are female, while 39.6% are male. As seen in Figure 3.4b, there is a weak link between the difficulty in paying by gender. This situation, which is experienced in a sensitive attribute such as gender, may lead to injustice during model training, but the current situation is not a strong enough difference to reveal a significant difference in the experiments conducted (For details, see Section 3.5)

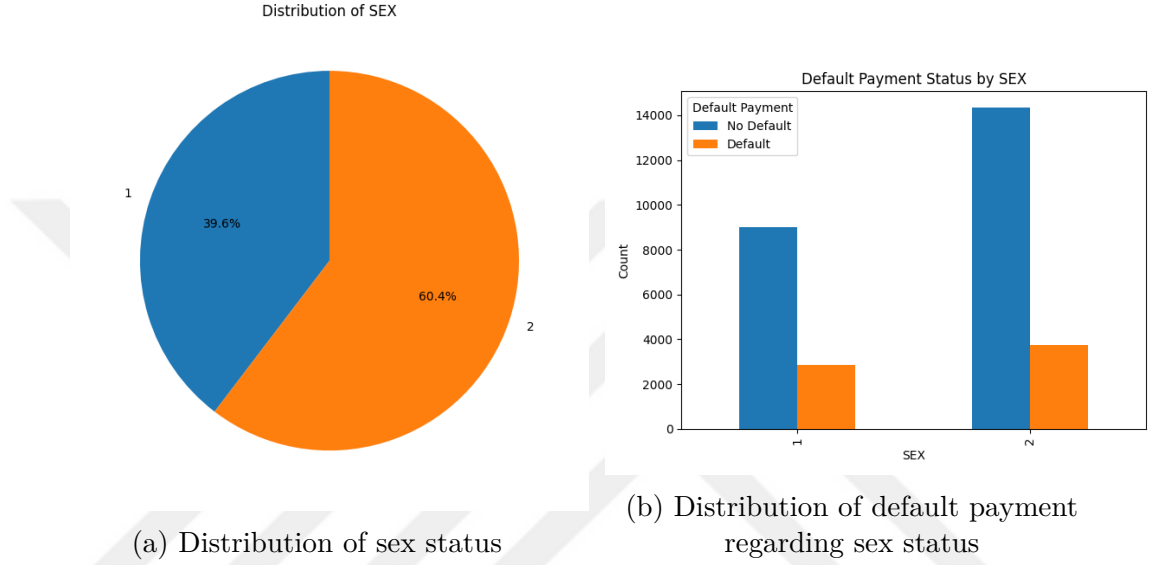


Figure 3.4 : Sex status among clients

Figure 3.5 shows the correlation between features. Accordingly, payment delay statuses (PAY_1, ..., PAY_6) have a very strong positive correlation with each other (between 0.5–0.8). In other words, your delay status in one month tends to continue in a similar way in the following months. Bill amounts (BILL_AMT1, ..., BILL_AMT6) behave almost as if they are the same (correlation 0.8–0.95). There is a serious multiple linear dependency between these variables. The feature importance status made for the kBest application is also noticeable here; the highest positive correlation with def_pay is between the first month's payment status (PAY_1: 0.32). In other words, the first month's payment performance is the strongest signal in determining the default risk. Invoice amounts and limit balance have a weak negative relationship with def_pay (between -0.02 and -0.15), and a high balance or invoice amount seems to slightly reduce the default

risk. Most of the demographic variables such as AGE, SEX, EDUCATION, MARRIAGE have no significant correlation with either payment or bill amounts.

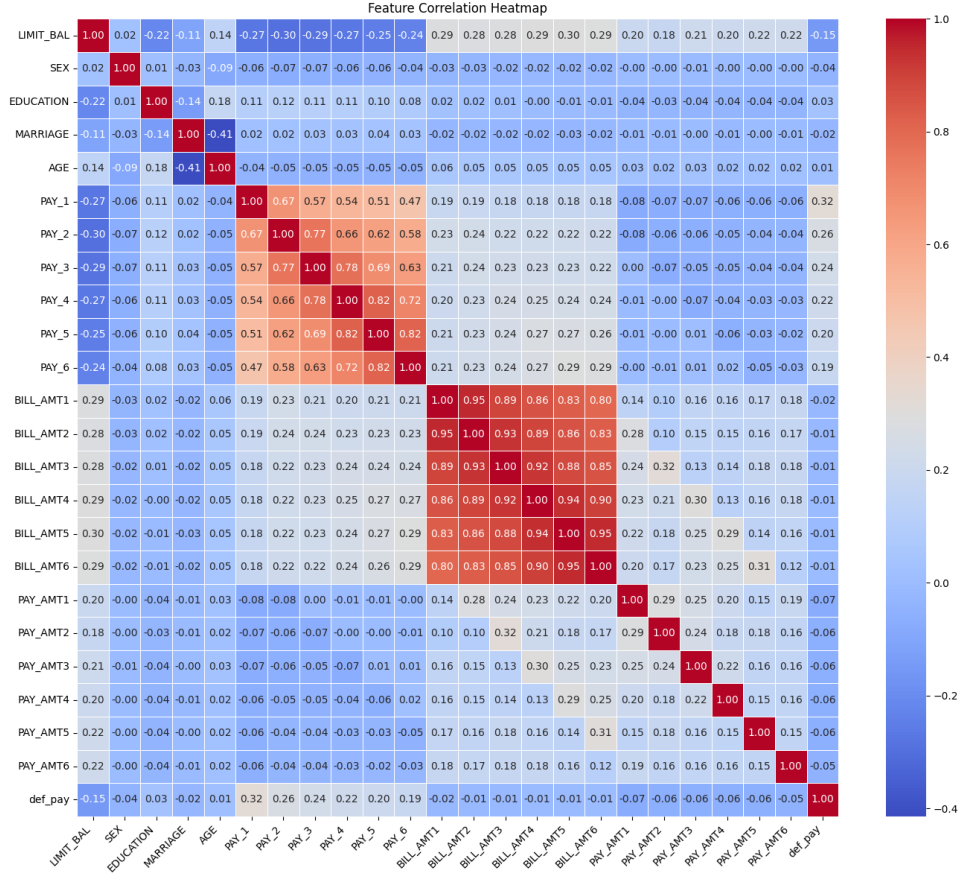


Figure 3.5 : Heatmap of features

3.2 Home Credit Default Risk Dataset

Home Credit B.V. was founded in 1997 in the Czech Republic and is now headquartered in the Netherlands [36]. Today, it operates in ten countries and partners with over 342,700 marketplaces (mostly in Russia, China, the Philippines, and Vietnam) to serve clients with little or no formal credit history [36,37]. Home Credit determines borrower risk through the analysis of financial information along with alternative sources such as data from telecom usage and transaction records. On Kaggle, they published the Home Credit Default Risk (HCDR) dataset [1], which contains 307,511 training and 48,744 test loan applications described by 122 features. These features include basic demographic

details (gender, education, family status) with information on employment and living conditions, such as years on the job and housing situation. The training dataset shows an 8.1% default ratio through its binary “TARGET” variable which indicates payment difficulty (1) or no issues (0) (Table 3.1). The data exists in multiple CSV files (application_train.csv, bureau_balance.csv, previous_application.csv) that we merged and these files are described in detail in the following sections.

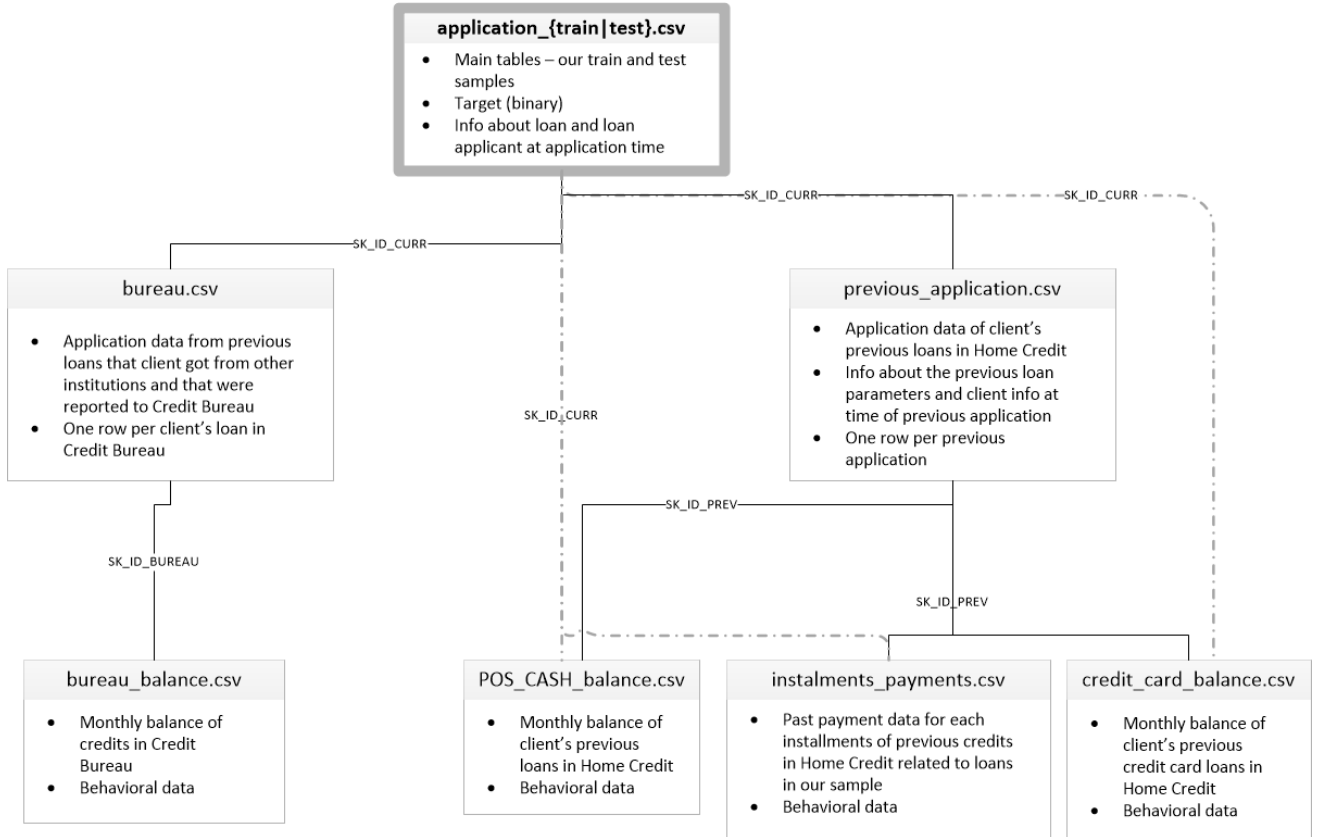


Figure 3.6 : Interrelated structure of files with each other. (Source: [1])

The dataset exhibits a hierarchical structure contains seven distinct but interrelated tables (Figure 3.6), each providing various information about the credit risk assessment process. The following details describe how these related data sources are connected in their architecture:

Primary Application Data (application_train|test.csv): These files constitute the core tables, which contain static information on loan applications. The training set comprises 307,511 samples, while the test set includes 48,744 samples. They consist of demographic information specific to the client, details

regarding the type and amount of loan requested, and a comprehensive set of 122 characteristics. Each client's loan status receives a binary encoding in the target variable where a value of 1 shows loan default and 0 indicates successful repayment. The test dataset lacks test set labels which prevents their use in this research because supervised learning techniques require labels.

Historical Credit Information (bureau.csv): The dataset containing historical credit information from credit bureaus serves as a supplement to the primary application data. The bureau.csv file contains aggregated data detailing clients' previous credit history, with 1,716,428 records spanning multiple credits from other financial institutions. Each client maintains records for different credit accounts.

Historical Credit Information-2 (bureau_balance.csv): It consists of the monthly situation of these previous credits, providing 27,299,925 records of monthly status data, with each row representing one month of a previous credit bureau.

Previous Loan Performance (previous_application.csv): The dataset gains more value through the addition of previous loan performance data in this file. This table contains 1,670,214 records detailing the outcomes of previous loan applications for Home Credit. The features in this file are similar to those in the current application data but also pertain to prior applications.

Transaction and Balance History (POS_CASH_balance.csv & credit_card_balance.csv): Transaction and balance history data are provided in the POS_CASH_balance.csv and credit_card_balance.csv files. The POS_CASH_balance.csv file contains 10,001,358 monthly balance records for past Point-of-Sale (POS) and cash loans that show the status of these loans during their active time. The credit_card_balance.csv file contains 3,840,312 records which reveal how clients use their credit.

Repayment Information (installments_payments.csv): It contributes repayment history data, with 13,605,401 records detailing payment information

for previous loans at HC. The payment records in this file show when borrowers made installment payments on time which helps evaluate their payment habits.

3.2.1 Data analysis of HCDR

In the HCDR dataset, the files "previous_application_preprocesd", "app_train_preprocesd", "bureau_preprocesd", "pos_cash_preprocesd", "credit_card_balance_preprocesd", and "installments_payments_preprocesd" used before the experiment were combined with each other with the "SK_ID_CURR" column, resulting in a single denormalized table. After that, the debt carrying capacity was determined with the ratio of monthly debt payments to income with the "AMT_INCOME_TOTAL" column to produce various features. In addition, various features were created for each family such as the number of children, the ratio of total credit to income, the ratio of credit amount to installments, the ratio of installments to income, the ratio of services price to credit amount, and the income per person in the family. In this de-normalized table, columns containing null values between 0-5% were filled with the median, columns containing null values between 5-25% were filled with the median, columns containing null values between 25-40% were filled with linear interpolation, and columns containing null values between 40-70% were filled with the *IterativeImputer* method. In addition, values containing null values over 70% were removed from the dataset. Then, the numerical values in the data were scaled with *MinMaxScaler*. Finally, 225 columns were selected for use in experiments using *VarianceThreshold(threshold = 0.01)* from the data preprocessing steps.

Figure 3.7 shows the data distribution of the "TARGET" label column. Accordingly, as mentioned in Table 3.1, there is a high rate of data imbalance (91.93% - 8.07%).

Figure 3.8 shows the correlation of features. Although everything is clearly visible in the table due to the large number of features, there is no specific correlation between the target column and the features. In addition, it is clear which parts of the other features are close to 1 and -1. In the detailed

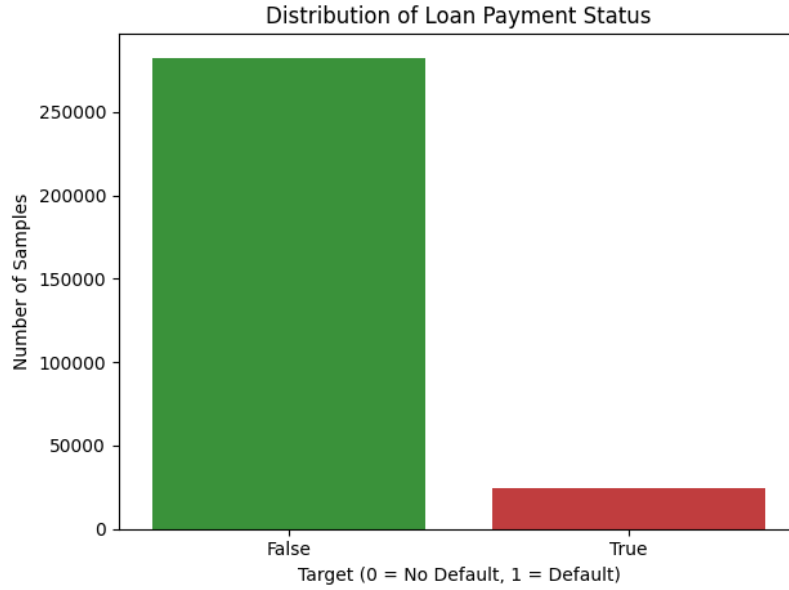


Figure 3.7 : Distribution of loan payment status

examination, the columns containing statistical features starting with "prep" have the highest positive and negative correlation with the target column. In addition, the columns starting with "clients" are also related to the target columns. Although not sufficient on their own, demographic variables such as "DAYS_BIRTH" and "REGION_RATING_CLIENT" are also weakly related to the target column. Another operation is to remove the features that have a correlation higher than 0.8 between each other. In addition, in the feature importance analysis conducted for kBest, the highest f1 value was reached with the following columns; 'DAYS BIRTH', 'DAYS ID PUBLISH', 'REGION RATING CLIENT', 'REGION RATING CLIENT W CITY', 'EXT SOURCE 1', 'EXT SOURCE 2', 'EXT SOURCE 3', 'DAYS LAST PHONE CHANGE', 'CODE GENDER F', 'CODE GENDER M', 'NAME INCOME TYPE Working', 'NAME EDUCATION TYPE Higher education', 'NAME EDUCATION TYPE Secondary / secondary special', 'preapp CODE REJECT REASON HC mean', 'preapp NAME PRODUCT TYPE walk-in mean', 'preapp NAME CONTRACT STATUS Refused mean', 'preapp NFLAG INSURED ON APPROVAL Missing mean', 'preapp NAME CONTRACT STATUS Approved mean', 'preapp CODE REJECT REASON XAP mean', 'preapp DAYS DECISION min', 'preapp DAYS FIRST DRAWING count', 'preapp DAYS FIRST DUE min', 'preapp DAYS

LAST DUE 1ST VERSION min', 'preapp DAYS LAST DUE min', 'client installments DAYS ENTRY PAYMENT min min', 'client installments DAYS INSTALMENT min min', 'client installments DAYS INSTALMENT mean min', 'client installments DAYS ENTRY PAYMENT mean min', 'client installments DAYS ENTRY PAYMENT max min', 'client installments DAYS INSTALMENT max min'



Figure 3.8 : Heatmap of features

3.3 Home Equity Dataset

The Home Equity (HMEQ) dataset comprises 5,960 home equity loan records, each annotated with 12 input features and a binary target variable BAD, which indicates loans where the borrower defaulted or became severely delinquent [10]. Among these 5,960 records, 1,189 (20%) are defaults (Table 3.1). The raw data is delivered in a single CSV file (hmeq.csv).

3.3.1 Data analysis of HMEQ

For HMEQ data, OneHotEncoder was applied for categorical columns in all experiments and StandardScaling method was applied for other numerical columns. In addition, as a result of our feature importance analysis, the following features were used in kBest application: 'DEROG', 'DELINQ', 'CLAGE', 'NINQ', and 'DEBTINC'. Table 3.3 shows the Non-Null values in the columns. In order to fill these null values, the columns 'MORTDUE', 'VALUE', 'YOJ', 'CLAGE' and 'DEBTINC' are imputed with their median values since they are continuous and right-skewed. In addition, the null values in the remaining columns are imputed with their mode values.

Table 3.3 : Null values distributions of HMEQ

#	Column	Non-Null Count	Dtype
0	BAD	5960 non-null	int64
1	LOAN	5960 non-null	int64
2	MORTDUE	5442 non-null	float64
3	VALUE	5948 non-null	float64
4	REASON	5708 non-null	object
5	JOB	5681 non-null	object
6	YOJ	5445 non-null	float64
7	DEROG	5252 non-null	float64
8	DELINQ	5380 non-null	float64
9	CLAGE	5652 non-null	float64
10	NINQ	5450 non-null	float64
11	CLNO	5738 non-null	float64
12	DEBTINC	4693 non-null	float64

Figure 3.9 shows the label distribution in the target column in the dataset. As previously shown in Table 3.1, there is an imbalance in the data. Accordingly, 80.05% of the dataset is loan paid and 19.95% is in the defaulted on loan state.

Figure 3.10a shows the reasons why users take out a home loan. Accordingly, 68.8% of the data took out a loan to buy a house, while 31.2% took out a loan for home renovation. According to Figure 3.10b, there is a small portion in both groups that cannot pay off the loan.

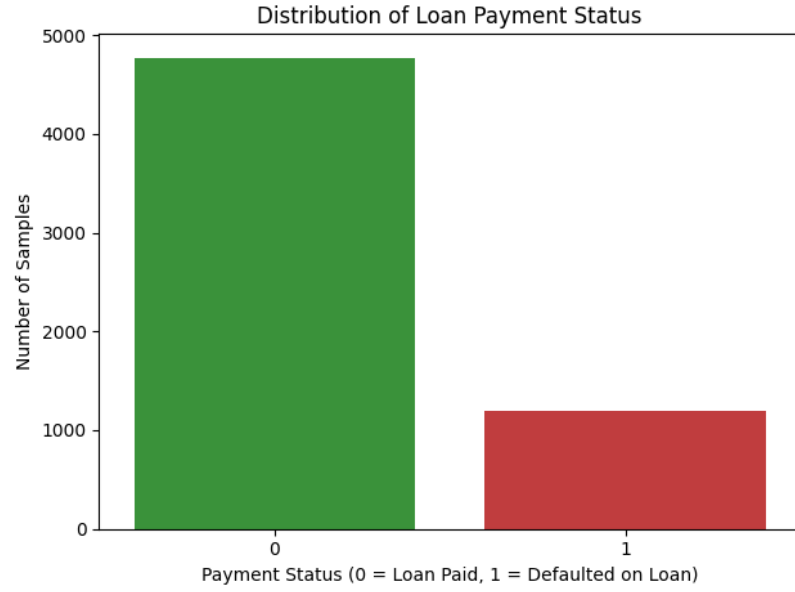
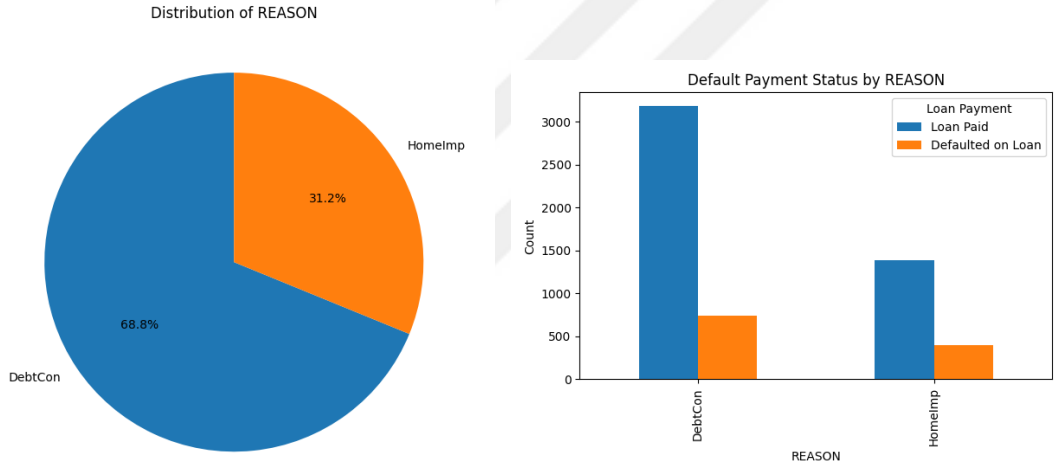


Figure 3.9 : Distribution of loan payment status



(a) Distribution of loan reason status

(b) Distribution of loan payment regarding loan reason status

Figure 3.10 : Loan reason status among clients

Figure 3.11a shows the occupational distribution of customers. While the Other class includes various occupational groups with 42%, it is followed by professional/executive (22.5%), office worker (16.7%), manager (13.5%), self-employed (3.4%) and sales (1.9%). In addition, 3.11b shows the repayment status of the loans they received according to their occupational groups. The class with the most difficulty in repaying is others and various occupational groups, followed by professional/executive and manager.

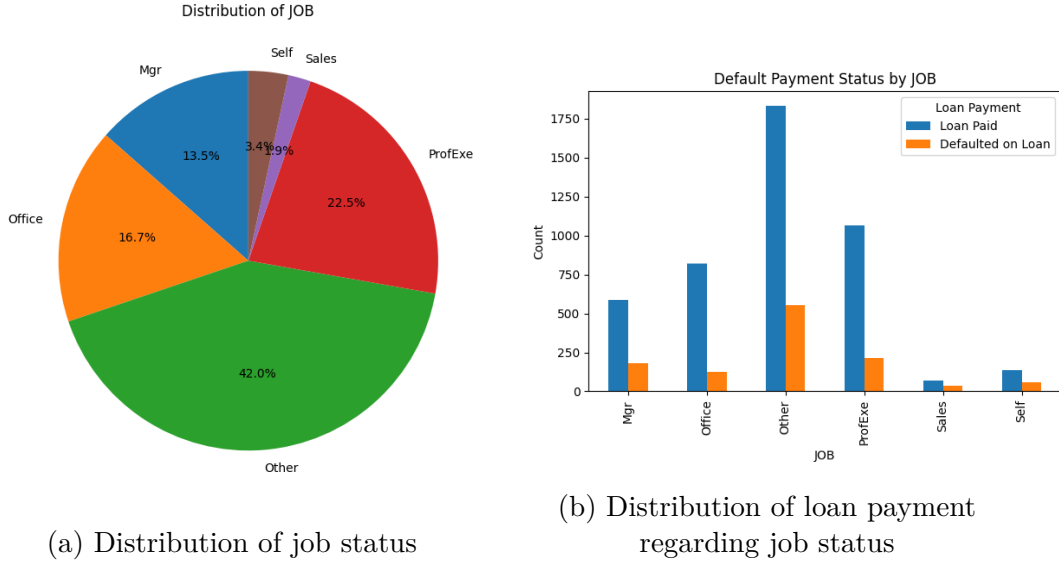


Figure 3.11 : Loan reason status among clients

Figure 3.12 shows the heatmap according to the features in the dataset. The strongest correlations with the BAD column come from past delays and negative credit records (DELINQ +0.35, DEROG +0.28). In addition, the debt/income ratio (DEBTINC +0.20) and the number of credit inquiries (NINQ +0.17) also increase the default risk. Accounts that have been used for a long time show that the risk decreases (CLAGE -0.17). These findings are consistent with the feature importance studies conducted for kBest.

3.4 Data Partitioning for Federated Learning

These three datasets, which we discuss in detail in previous sections, were initially prepared as a single file, as in CML models. However, the basic principle of FL is based on the principle that data stores with clients instead of being collected on a central server, and that no client has direct access to another client's data. Therefore, we had to divide the CML datasets between clients to simulate real world Non-IID scenarios.

To this end, we ran our FL experiments using the Flower Framework [38]. The *DirichletPartitioner* [39] method from Flower implements a partitioning strategy based on the Dirichlet distribution [40] to assign data shares with different class distributions to each client. In this method, the level of

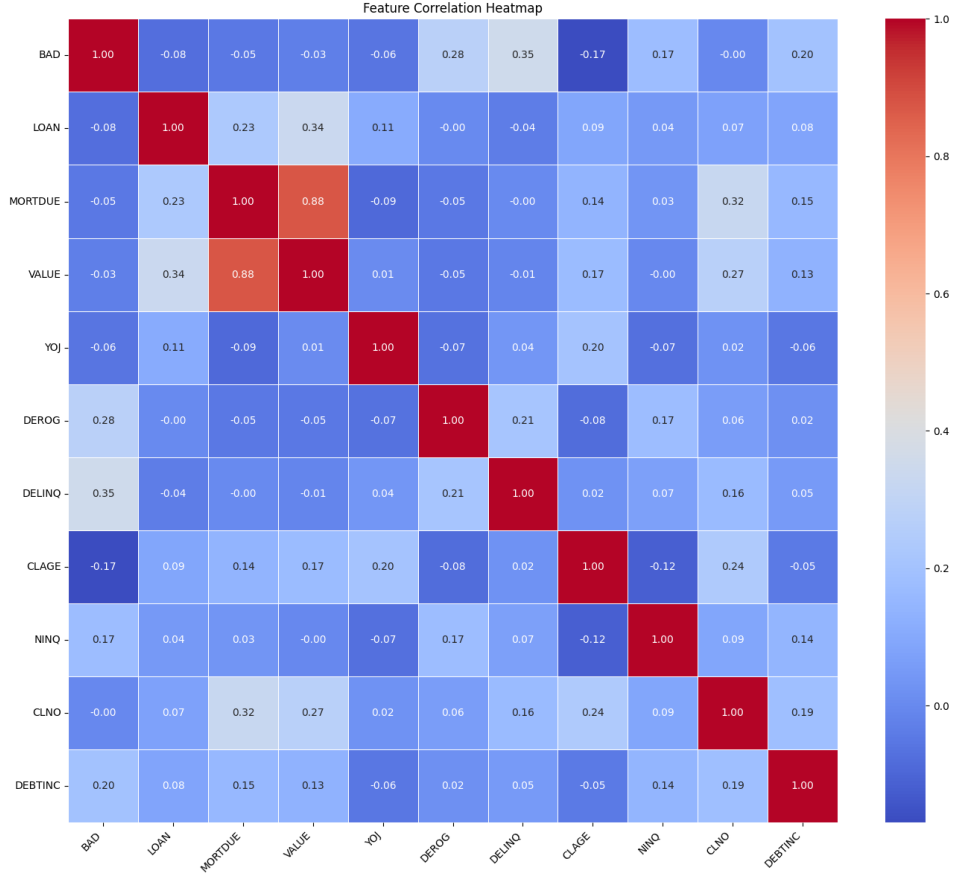


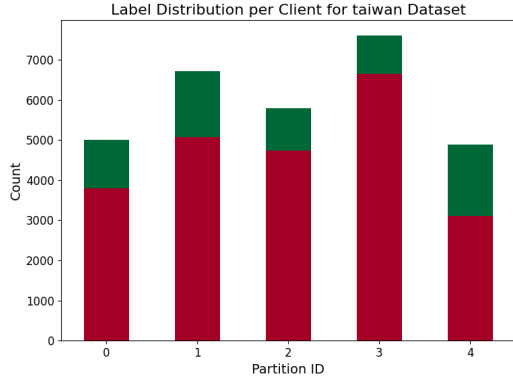
Figure 3.12 : Heatmap of features

heterogeneity is set via the parameter α (alpha), the number of clients is controlled by *num_partitions* and the target column information is controlled by *partition_by*. In our benchmarking experiments, we created a scenario with 5 clients and partitioned our datasets between them using *DirichletPartitioner*. In each experiment;

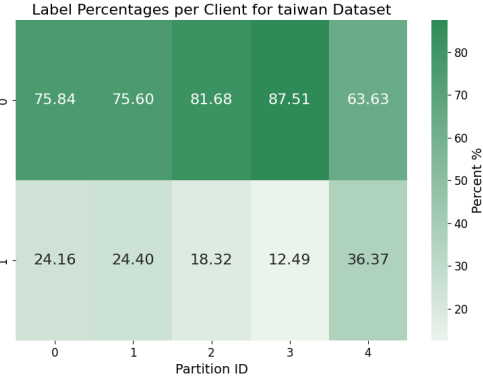
1. Entire central data was divided to 5 clients via *DirichletPartitioner* for the first time during the experiment.
2. The data for each client is split into train, test and validation datasets.

As an example, the label distribution of the dataset per clients and the label percentages of each class are shown in Figure 3.13. Figure 3.13a, 3.13c, and 3.13e show the frequencies of “default” and “non-default” classes on each client in the DCCC, HCDR and HMEQ datasets, respectively. Figure 3.13b, 3.13d, and 3.13f indicate the extent of the imbalance by presenting the percentage values of the

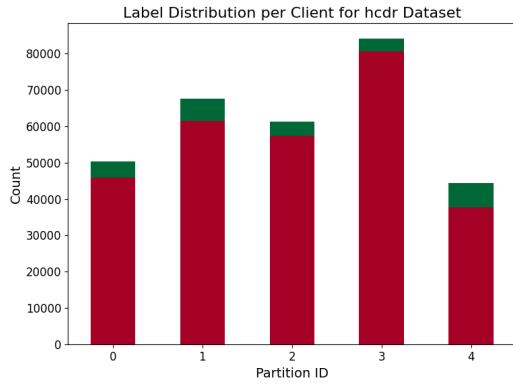
same distributions in heatmap format. This clearly illustrates how the different label rates on each client reflect real world non-IID scenarios and the extent of data heterogeneity in our FL experiments.



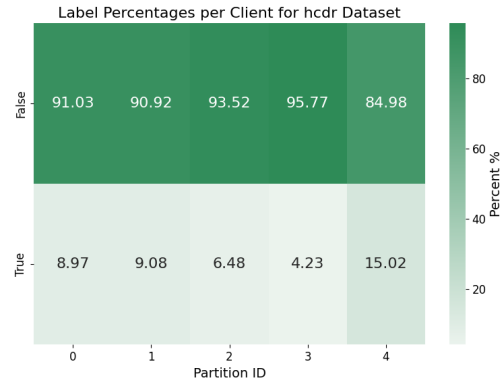
(a) Label distribution per client for DCCC Dataset.



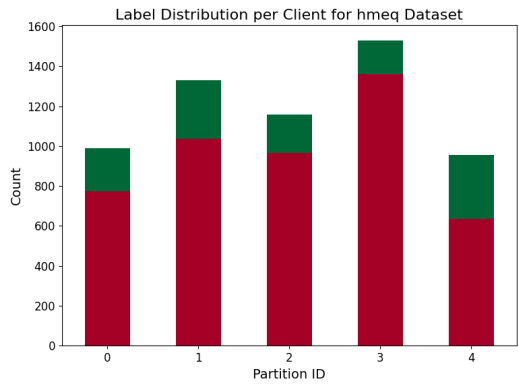
(b) Label percentages per client for DCCC Dataset.



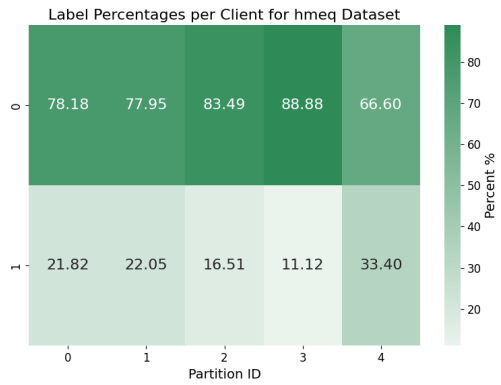
(c) Label distribution per client for HCDR Dataset.



(d) Label percentages per client for HCDR Dataset.



(e) Label distribution per client for HMEQ Dataset.



(f) Label percentages per client for HMEQ Dataset.

Figure 3.13 : Label distribution and percentage after using *DirichletPartitioner*. (Green as label 1, while red as label 0.)

3.5 Synthetic Noise Generation for Fairness Studies

The observability of fairness studies in FL requires unfair datasets as the first step. However, as seen in Table 4.1, HMEQ datasets do not contain any obvious social bias by nature, so it is not possible for us to use this dataset for our fairness experiments. Then, we were undecided in between DCCC and HCDR datasets for our fairness experiments. However, as we stated in Section 6, where we interpreted the tables values, that we preferred the DCCC dataset due to its prevalence in the literature and its fit to real world scenarios. We chose the gender bias (SEX) for this dataset as sensitive feature. However, in order to better observe the effect of fairness in the experiments, we decided to create a synthetic feature regarding the SEX feature of this dataset.

For this purpose, we used a synthetic noise that creates a gender specific correlation between a new attribute called “Interest” of each loan applicant and the target variable (default) [41]. The noise was generated by the “Interest” score drawn from Gaussian distributions with different variance values for male and female candidates as shown in Equation 3.1.

$$\text{Interest} \sim \begin{cases} \mathcal{N}(2 \times \text{Default}, 1), & \text{if Sex} = \text{Male}, \\ \mathcal{N}(2 \times \text{Default}, 2), & \text{if Sex} = \text{Female}. \end{cases} \quad (3.1)$$

Figure 3.14 shows the density curves after synthetically applying Equation 3.1 for Interest feature for male and female applicants. The “Payment Default” group shows higher Interest values than the “Payment on Time” group for both genders because defaulting individuals have higher synthetic interest scores. Finally, after integrating the Interest feature into the DCCC dataset, we run our fairness experiments with this structured data.

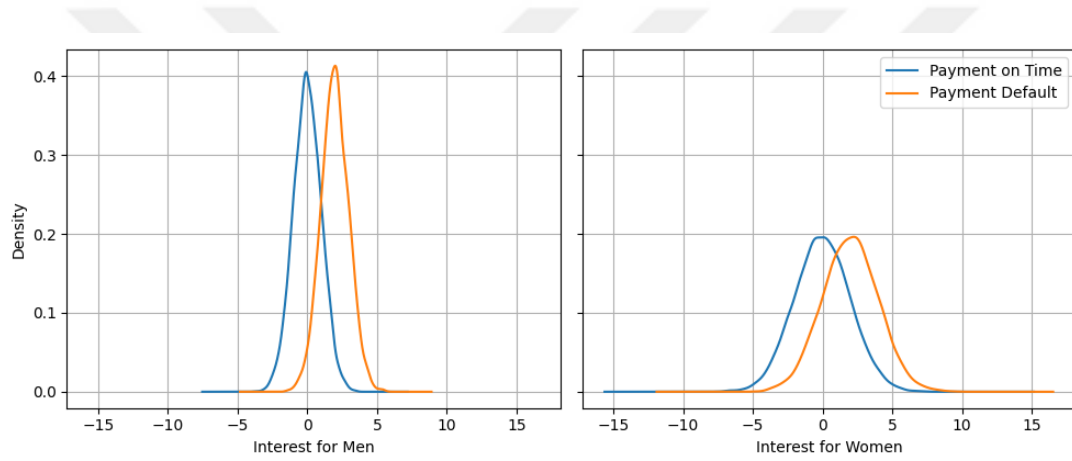


Figure 3.14 : Kernel Density Estimation distributions of the synthetic Interest attribute by gender and payment status.

4. METHODOLOGY

This section introduces the technology and methods we use so that you can better understand the experiments we will explore in detail later. In addition, we have shared all model parameters and project code in order to ensure the reproducibility of both experiments we conducted in our study. First, we explained the 5 ML algorithms we used in our experiments in Section 4.1. In addition, we mentioned data sampling methods in Section 4.2 and feature reduction methods in Section 4.3. Then, we talked about federated learning, which is the basis of our entire study, in Section 4.4 and explained FedAvg and FedF1 aggregation methods that we developed in Section 4.5. We explained the fairness issue, which is necessary for our second experiment in our study, in Section 4.6 and explained our metrics that we evaluated all our experiments in Section 4.7. At the end of the day, we shared information about our environments where we performed all our experiments in Section 4.8.

4.1 Machine Learning Algorithms

In our study, we have chosen different types of ML algorithms to be able to observe all effects. Accordingly, we used Logistic Regression (LR) (Section 4.1.1) as Linear Model, Multi Layer Perceptron (MLP) (Section 4.1.2) as Neural Network based model, Support Vector Machine (SVM) (Section 4.1.3) as Kernel based model, eXtreme Gradient Boosting (XGB) (Section 4.1.4) as Ensemble method, Random Forest (RF) (Section 4.1.5) as Tree based model.

4.1.1 Logistic Regression

The Logistic Regression (LR) [42] turns the output of a linear model into a probability using the sigmoid function. It is parametric since it employs many parameters to estimate a function from data, and discriminative as it directly

models the link between specified attributes (independent variables) and class labels (dependent variable) through class distinctions [43].

4.1.2 Multi Layer Perceptron

The Multi Layer Perceptron (MLP) [44] is a fully connected, feed-forward neural network that contains a minimum of one hidden layer. It exhibits strong performance in many classification tasks by means of its ability to model complex nonlinear relationships. However, to lower the risk of overfitting, it is very important to carefully tune the architecture and hyperparameter values.

4.1.3 Support Vector Machine

The Support Vector Machine (SVM) [45] is a parametric linear classification method that uses a hyperplane in the data dimension to divide two classes into two groups. The kernel trick moves data that can't be separated in a straight line to higher dimensions. The FL method depends on combining algorithm parameters based on the numbers that clients give. So, the number of factors from clients and their format must be the same. To do this, we used SVC with a linear kernel.

4.1.4 eXtreme Gradient Boosting

The XGBoost (XGB) [46] uses gradient boosting principles to construct multiple weak learners which are typically DTs that are stacked one after another to enhance and merge them. This method creates a strong prediction model by cutting down on mistakes. It is preferred for large data sets due to its high speed and regularization properties.

4.1.5 Random Forest

The Random Forest (RF) [47] ensemble method uses multiple DTs to generate predictions through majority voting. Parameters such as number of trees, depth and feature selection play a critical role to reduce the risk of overfitting. The

model outputs enable us to determine variable importance ratings which show the relative contribution of each variable.

4.2 Data Sampling Techniques

Since the datasets we used in our study are imbalance as shown in Table 3.1, we tested SMOTE (Section 4.2.1) and RUS (Section 4.2.2) methods in our experiments to show the effects of data sampling techniques.

4.2.1 Synthetic Minority Over-sampling

The Synthetic Minority Over-sampling (SMOTE) algorithm [48] overcomes class imbalance through minority class sample generation to prevent overfitting. The algorithm chooses a data point from the minority class. Then identifies the k nearest neighbors of the selected minority class data points. It determines the differences between the selected random neighbor and the original data points. A random value between 0 and 1 is used to multiply the calculated difference. The original point receives this calculated value to generate a synthetic sample. The process of creating new synthetic samples involves combining existing samples with newly generated samples through interpolation. SMOTE was used to isolate the target variable from the training data and create synthetic minority class samples for the training data. Please note that we applied SMOTE only to the training set, leaving test and validation sets unchanged to ensure unbiased model evaluation.

4.2.2 Random Under Sampling

The Random Under Sampling (RUS) method [49] addresses class imbalance by randomly removing majority class samples until the majority class matches the minority class size or reaches a specified ratio. The method works well with large datasets because it requires minimal computational power. The main drawback of RUS is its ability to discard essential information needed to define the model's decision boundary because it removes majority class data randomly. The removal of crucial data points from the majority class during the border

determination process can improve the process. The model’s performance may decrease because of its reduced dataset size. We applied RUS to each client’s data individually rather than to the entire dataset. After splitting the dataset with *DirichletPartitioner* algorithm, we used RUS on each client’s data to ensure balanced classes.

4.3 Feature Reduction Techniques

Since the datasets we use in our study have a wide variety of feature counts, we wanted to measure their impact on model performance by using Principal Component Analysis (PCA) (Section 4.3.1) and Select k-best (kBest) (Section 4.3.2) in our experiments.

4.3.1 Principal Component Analysis

The Principal Component Analysis (PCA) is a statistical technique reducing dimensionality by identifying the most significant directions of variation in data. The method first centers the data before computing the covariance matrix C and then chooses the k eigenvectors with the highest eigenvalues to create a new subspace. We used PCA on the train set data and performed the same transformation on both validation and test sets.

4.3.2 Select k-Best

The Select k-Best (kBest) algorithm identifies the most important k features present in a dataset. The algorithm determines feature importance through statistical tests and user-defined metrics to select the most relevant characteristics. We evaluated Categorical data using the *f_classif* technique and evaluated numerical data using the *chi2* method. This simplified training approach enhanced model performance and forecast accuracy.

4.4 Federated Learning

Federated Learning (FL) is a ML approach that allows training models that remain distributed across multiple clients, rather than aggregating data in a

central repository. The foundations of FL date back to 2015, when Konečný et al. [50] coined the term “federated optimization” to improve communication over distributed data on mobile devices. In a next step, in 2016, McMahan et al. [2] introduced the concept of FL, which iteratively improves model updates under real-world constraints (e.g., non-IID data, limited bandwidth of mobile devices, and user privacy issues). In this way, it became possible for private and large volumes of data, especially keyboard inputs, to contribute to local education without being sent to a central server, user privacy was preserved, and data traffic on the network was significantly reduced.

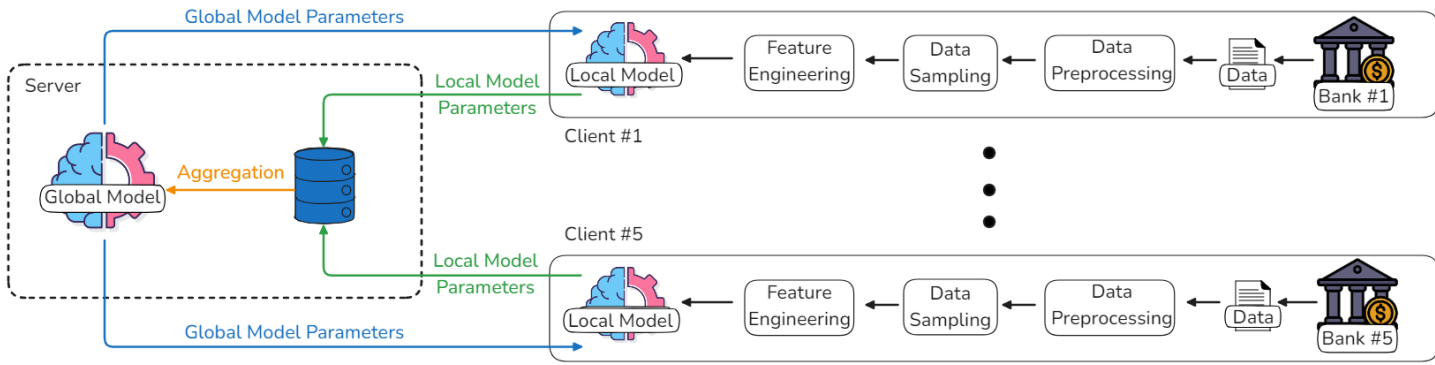


Figure 4.1 : Federated Learning system architecture

The process in FL consists of the following steps as shown in Figure 4.1:

1. Each client trains the current local model in its own environment with its own data.
2. Once the training is complete in local, the clients send only the model weights (parameters) to the central server without sharing the raw data (Shown in green line).
3. The server aggregates the weights from the different clients (Shown in orange line) and creates a new global model using the most common FedAvg algorithm that uses weighted average of the weights from the clients.
4. The updated global model is again sent to all clients (Shown in blue line).
5. Clients retrain the incoming global model with their local data, and steps 2-4 are repeated for the specified number of rounds.

This iterative structure indicates the so-called horizontal FL scenario, where each client has different instances with similar feature sets. In our work, we use this approach to perform both performance and fairness evaluations without violating the principle of keeping the data private.

4.5 Aggregation Techniques

The most distinguishing step in the FL process is the aggregation step, where the models are brought together (Figure 4.1 orange line). While all other steps, such as local training, parameter update and transmission, are similar to classical CML practice, the aggregation of model weights from clients into a single global model is a process unique to FL.

In this section, we first discuss Federated Averaging (FedAvg), the most widely used method in the FL world. Then, we will introduce the F1 Score Weighted Aggregation (FedF1) approach that we developed to balance performance in imbalance dataset scenario.

4.5.1 Federated Averaging

Federated Averaging updates a global model by averaging model weights from clients (K) where each client's weight (k) depends on its data contribution to the server [2]. Each client trains the model locally and sends the updated weights to the server. The server aggregates those weights, weighted based on the amount of data used by each client during training. The model weights (w_{t+1}^k) are aggregated based on the number of data points (n_k) from each client, and the total number of data points (n) across all clients is divided by the sum to derive the weighted average, namely global model (w_{t+1}) (4.1). The updated global model is redistributed to the clients, after which the process proceeds to the next iteration. The distributed training process through FedAvg enables global model updates while maintaining client level data privacy.

$$w_{t+1} \leftarrow \sum_{k=1}^K \frac{n_k}{n} w_{t+1}^k \quad (4.1)$$

The Federated Averaging process (Figure 4.2) is briefly illustrated as a server-side flow in Algorithm 1 and a client update procedure in Algorithm 2. In Algorithm 1, the parameter C is used to determine the proportion of clients that will participate in the FL process for each round, and m is used to represent the actual number of clients corresponding to this proportion. The other parameters (e.g. B : minibatch size, E : number of local epochs, η : learning rate and n_k : number of samples for client k) are defined in detail in the previous sections and are used to control the training-aggregation cycle.

Algorithm 1 Federated Averaging	Algorithm 2 ClientUpdate Procedure
1: Server executes: 2: initialize w_0 3: for each round $t = 1, 2, \dots$ do 4: $m \leftarrow \max(C \cdot K, 1)$ 5: $S_t \leftarrow$ random set of m clients 6: for each client $k \in S_t$ in parallel 7: $w_{t+1}^k \leftarrow \text{ClientUpdate}(k, w_t)$ 8: end for 9: $m_t \leftarrow \sum_{k \in S_t} n_k$ 10: $w_{t+1} \leftarrow \sum_{k \in S_t} \frac{n_k}{m_t} w_{t+1}^k$ 11: end for	1: ClientUpdate (k, w): 2: $\mathcal{B} \leftarrow$ split local data P_k into batches of size B 3: for epoch $i = 1$ to E do 4: for each batch $b \in \mathcal{B}$ do 5: $w \leftarrow w - \eta \nabla \ell(w; b)$ 6: end for 7: end for 8: return w

Figure 4.2 : Federated Averaging Algorithm [2]

4.5.2 F1 Score Weighted Aggregation

F1 Score Weighted Aggregation was developed for this study to improve model performance. FedF1 aggregates model weights (w_{t+1}^k) from clients (K) based on their F1 scores ($f1_k$) (4.2) over all F1 scores ($F1_t$) and creates the global model (w_{t+1}). To enhance the overall model's reliability and stability, it assesses the performance of each client and gives more weight to models with higher F1 scores. FedF1 was utilized due to the significant impact of the imbalanced dataset on model performance. F1 score, as a weighted average of precision and recall (4.2), balances these metrics. As shown in Algorithm 3 and Algorithm 4, we assign higher weights to clients with more stable and superior F1 scores to improve the overall model.

$$w_{t+1} \leftarrow \sum_{k=1}^K \frac{f1_k}{F1_t} w_{t+1}^k \quad (4.2)$$

Algorithm 3 F1 Score Weighted Aggregation (FedF1)

```

1: Server executes:
2: initialize  $w_0$ 
3: for round  $t = 0, 1, 2, \dots$  do
4:    $S_t \leftarrow \{1, \dots, K\}$ 
5:   for each client  $k \in S_t$  in parallel
6:      $(w_{t+1}^k, f1_k) \leftarrow \text{ClientUpdate}(k, w_t)$ 
7:   end for
8:    $F1_t \leftarrow \sum_{k \in S_t} f1_k$ 
9:    $w_{t+1} \leftarrow \sum_{k \in S_t} \frac{f1_k}{F1_t} w_{t+1}^k$ 
10: end for

```

Algorithm 4 ClientUpdate Procedure

```

1: ClientUpdate( $k, w$ ):
2:  $D_{k_{\text{train}}}, D_{k_{\text{val}}} \leftarrow \text{ClientData}(k)$ 
3: for epoch  $i = 1$  to  $E$  do
4:   for each batch  $b \in D_{k_{\text{train}}}$  do
5:      $w \leftarrow w - \eta \nabla \ell(w; b)$ 
6:   end for
7: end for
8:  $f1 \leftarrow \text{CalcF1Score}(w, D_{k_{\text{val}}})$ 
9: return ( $w, f1$ )

```

Figure 4.3 : FedF1 server and client update procedures

4.6 Fairness in Machine Learning

ML systems directly affect human life and social structure through automated decision-making processes. These real life applications spans from recruitment processes to loan approvals, from forensic risk assessments to health diagnoses [51]. When algorithms are trained on biased data and employed in various applications, they may lead to discrimination against disadvantaged groups, worsen inequalities or creating new forms of discrimination. The literature focuses on two fairness approaches which address bias and fair decision-making through group fairness and individual fairness. Group fairness maintains equal outcomes between different demographic groups but individual fairness demands equal treatment for people who share similar characteristics.

Table 4.1 provides a summary of the three datasets mentioned in our study from a fairness perspective. Accordingly, the A value represents the sensitive attribute, $\Pr(A = 1)$ represents the advantageous group rate among the total samples, $Y = \oplus$ represents which label value is considered as a positive (favorable) outcome, and $\Pr(Y = \oplus)$ represents the positive outcome rate among all samples. According to the table, the HMEQ dataset is not suitable for fairness studies. In addition,

Table 4.1 : Summary of datasets used in the study in terms of fairness.

	DCCC	HCDR	HMEQ
# sample	30,000	307,511	5,960
Protected attributes	Sex, Education, Marriage	Sex, Marriage	–
Sensitive attribute (A)	Sex	Sex	–
$A = 1$	Male	Male	–
$\Pr(A = 1)$	0.40	0.34	–
Target variable	default (1=yes)	default (1=yes)	bad_loan (1=yes)
$Y = \oplus$	default = 1	default = 1	bad_loan = 1
$\Pr(Y = \oplus)$	0.22	0.08	0.19

since the DCCC dataset is used more frequently than HCDR in the literature, only the DCCC dataset was used in the fairness experiments in our study.

4.6.1 Group fairness

The goal of group fairness also known as statistical or demographic fairness is to guarantee that ML models generate consistent and comparable results between different demographic groups including race, gender and age [51]–[53]. The objective of this approach is to detect group-level biases through the comparison of basic statistics including positive decision rates and error rates generated by the classification algorithm for different groups. The metrics for detecting fairness and bias in the trained model are explained in detail in Section 4.7.2 and the protective measures to address bias and unfairness are described in Section 4.6.3.

4.6.2 Individual fairness

The main objective of individual fairness is to ensure that people with similar characteristics (e.g. people with similar credit histories or demographic profiles) get similar results from ML models. The principle of this method is that “similar individuals should get similar results” [54] and it ensures the consistency of model predictions by measuring the similarity between inputs (e.g. Mahalanobis distance or Lipschitz condition). Individual fairness metrics and calculation

methods are detailed in Section 4.7.2, while techniques to address these differences are described in Section 4.6.3.

4.6.3 Fairness mitigation techniques

The goal of pre-processing techniques is to reduce the disparities in training data before starting model training. Oversampling or undersampling [55] is used to make sure that minority and majority groups are equally represented when relabeling prior biased labels [56]. For example, changing the label of female candidates to positive in recruitment data. In addition, the Disparate Impact Removal (DIR) algorithm, which reduces the link between model attributes and proxy variables [57], and Fair Representation Learning [58], which learns fair representations independent of the protected attributes, are often used. The training data is made as fair as possible to prevent the model from reflecting group level discrimination.

In in-processing methods, fairness goals are directly incorporated into the optimization process of the model. The optimization process at each update step balances both accuracy and fairness metrics through the addition of fairness constraints or regularization terms to the loss function. For example, with the Prejudice Remover Regularizer, fairness penalty terms can be added to the model parameters [59]. In addition to adding fairness constraints to the loss function in CML, an example of an in-processing approach in the FL scenario is the weighting of model parameters from clients according to fairness criteria in the global aggregation phase. The FairFed approach detailed in Section 4.6.4 is an in-processing technique that optimizes the global model by aggregating clients updates in each round according to fairness metrics.

The output scores or classification thresholds of trained models undergo post-processing adjustments to achieve fairness between different groups. The application of distinct decision thresholds (threshold adjustment) to each group enables the achievement of Equalized Odds [60] or Equal Opportunity metrics; for example, in LR, the threshold for women is set at 0.6 and 0.5 for men to

balance the true positive rates. These methods offer the possibility to intervene only at the output level without retraining an existing model.

4.6.4 Fairness in federated learning: FairFed approach

Machine learning applications that use sensitive data have demographic information (such age and gender) that is hidden yet sensitive that leads models learn discriminating biases that lead to worse results for some groups. This is why fairness and ethics are of great importance in machine learning applications.

The FairFed [31] study modifies the traditional FedAvg strategy by incorporating fairness weights into the server-side model update process. The authors propose to adjust the global model update (aggregation) weights on the server side according to the fairness metrics, without disrupting the local debiasing that each client applies to its own data. The system enables each client to use any local debiasing method on their data independently regardless of other clients' methods. The local debiasing method enables clients to apply reweighting [32] or FairBatch [33] techniques to their datasets independently without needing central access to eliminate bias from their models.

The FairFed system implements fairness through group fairness which defines equality of positive prediction rates across groups based on the binary sensitive attribute ($A \in \{0,1\}$) that labels each data point (privileged group $A = 1$ and unprivileged group $A = 0$). The FairFed system uses a weighting approach that bases the fairness differences between global (F_{global} , equation 4.3) and local (F_k , equation 4.4) group biases. The weighting method enables clients with lower bias (more fair) to contribute model updates greater. The F_{global} value that will be sent to other clients remains fair otherwise biased client behavior cannot affect other clients. The formula notation used throughout the study is as follows: K denotes the total number of clients and D_k is the dataset of the k th client with size n_k . The symbol θ_k^t refers to the model parameters updated by client k in t th round, while ω_k^t is the aggregation weight that client k employs in that same round. The sensitive attribute A takes the value 1 for the privileged group and

0 for the non-privileged group, and $\hat{Y}(\theta, x)$ denotes the model's binary output, with $\hat{Y} = 1$ indicating a positive decision.

Global group fairness (in the context of EOD) is defined on the entire joint set ($D = \bigcup_k D_k$) as in Equation 4.3. The local fairness metric is calculated for each client k only on D_k as in Equation 4.4 and measures the imbalance between A=1 privileged and A=0 unprivileged groups.

$$F_{\text{global}} = \Pr(\hat{Y} = 1 \mid A = 0, Y = 1) - \Pr(\hat{Y} = 1 \mid A = 1, Y = 1) \quad (4.3)$$

$$F_k = \Pr(\hat{Y} = 1 \mid A = g, Y = 1, C = k) - \Pr(\hat{Y} = 1 \mid A = g, Y = 1, C = k). \quad (4.4)$$

The server updates the aggregation weights based on the difference between the global and local fairness metrics in each round. The method defined in Equation 4.5 performs a calculation that determines the effect of the clients on the overall model in each federated training round. In this formula, Δ_k^t stands for fairness difference. One situation in which the bias measure can't be calculated is when neither group has any positive class examples. For this particular situation, the accuracy metric is used instead of the fairness. The value of β in Equation 4.6 adjusts the relation between the model's accuracy and fairness. High values of β makes fairness stronger, but 0 makes that it is the same as the FedAvg.

$$\Delta_k^t = \begin{cases} |Acc_k^c - Acc_k^t|, & \text{if } F_k^t \text{ is undefined,} \\ |F_{\text{global}} - F_k^t|, & \text{otherwise,} \end{cases} \quad (4.5)$$

$$\bar{\omega}_k^t = \omega_k^{t-1} - \beta \left(\Delta_k - \frac{1}{K} \sum_{i=1}^K \Delta_i \right), \quad (4.6)$$

$$\omega_k^t = \frac{\bar{\omega}_k^t}{\sum_{i=1}^K \bar{\omega}_i^t}. \quad (4.7)$$

The server obtains the global EOD from the direct sharing of the local performances of the clients as in Equation 4.8.

$$\begin{aligned} F_{\text{global}} &= \Pr(\hat{Y} = 1 \mid A = 0, Y = 1) - \Pr(\hat{Y} = 1 \mid A = 1, Y = 1) \\ &= \sum_{k=1}^K \frac{n_k}{n} \left(\frac{\Pr(\hat{Y} = 1 \mid A = 0, Y = 1, C = k) \Pr(A = 0, Y = 1 \mid C = k)}{\Pr(Y = 1, A = 0)} \right. \\ &\quad \left. - \frac{\Pr(\hat{Y} = 1 \mid A = 1, Y = 1, C = k) \Pr(A = 1, Y = 1 \mid C = k)}{\Pr(Y = 1, A = 1)} \right). \end{aligned} \quad (4.8)$$

The general FairFed training steps are as follows:

1. Local Update: Each client k trains the global model θ^{t-1} with its own data D_k and generates θ_k^t .
2. Global Metric Calculation: The server collects the global fairness F_{global}^t and global accuracy Acc statistics from the clients with secure aggregation.
3. Weight Calculation: Clients generate Δ_k^t and then ω_k^t values (according to equation 4.6) from the received F_{global}^t , Acc and their own local fairness F_k^t .
4. Weighted Aggregation: Clients send $\omega_k^t \theta_k^t$ and ω_k^t values to the server with secure aggregation.
5. Model Update: The server calculates the new global model with the formula here: $\theta^t = \sum_{k=1}^K \omega_k^t \theta_k^t$ and broadcasts it to the clients.

With this mechanism, FairFed both preserves the flexibility of local debiasing and effectively improves global group fairness in heterogeneous client distributions

4.7 Evaluation Metrics

For two different experiments we conducted in our study, we will first present the metrics used in classical ML in Section 4.7.1, and then present the relevant metrics for the fairness experiment in Section 4.7.2.

4.7.1 Performance metrics for machine learning

We used various evaluation metrics to make the experimental results more meaningful. To evaluate the performance of different classification methods, we use recall, precision, F1 score and accuracy as shown in Table 4.3. Also, the Confusion Matrix required for the metrics is given in Table 4.2.

Table 4.2 : Confusion matrix definition.

	Predicted (1)	Predicted (0)
Actual (1)	True Positive (TP)	False Negative (FN)
Actual (0)	False Positive (FP)	True Negative (TN)

Table 4.3 demonstrates that accuracy represents the model’s correct classification rate across all examples. The equal weighting approach of this method produces deceptive results when the data distribution is unbalanced. Precision indicates the proportion of actual positive examples among all examples that the model predicts as positive. The Recall metric indicates the proportion of actual positive instances that the system correctly identifies. The F1 score calculates the harmonic mean between precision and recall to provide a balanced evaluation of these metrics. The F1 score method provides more significant results than accuracy because it considers both false positives and false negatives in unbalanced data sets.

Table 4.3 : Evaluation metrics and their formulas.

Evaluation Metrics	Formula
Recall (R)	$\frac{TP}{TP + FN}$
Precision (P)	$\frac{TP}{TP + FP}$
F1-Score (F1)	$\frac{2 \times \text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}}$
Accuracy (A)	$\frac{TP + TN}{TP + FP + TN + FN}$

4.7.2 Measuring fairness and bias

The following fairness metrics are commonly used in the literature. These metrics evaluate model decision fairness at the group level through absolute (SP, DI), difference (SPD, EOD, PP) and ratio (EO) based measures. The FairFed focuses on SPD, EOD and EO metrics while to maintain global and local fairness in the federated environment. The EOD based approach is used in this study since it is employed in FairFed. In the formulas below, $\hat{Y}$ represents the estimate, Y represents the label and A represents the sensitive attribute.

4.7.2.1 Statistical Parity

Statistical Parity (SP) aims to equalize the positive decision rates between groups. For example, in the bank loan model, if both male and female customers get 30% approval, it shows that SP is provided. The necessary condition for SP to be provided is shown below. Accordingly, the probability of the model giving approval is equal for both advantaged and disadvantaged groups.

$$P(\hat{Y} = 1 \mid A = 1) = P(\hat{Y} = 1 \mid A = 0) \quad (4.9)$$

4.7.2.2 Statistical Parity Difference

Statistical Parity Difference (SPD) is the quantitative difference form. For example, in a model where men get 35% approval and women get 25% approval, $SPD = 0.10$, which means a 10% point disadvantage to women.

$$SPD = P(\hat{Y} = 1 \mid A = 1) - P(\hat{Y} = 1 \mid A = 0) \quad (4.10)$$

4.7.2.3 Equal Opportunity Difference

Equal Opportunity Difference (EOD) measures the difference between the true positive rates. In the health diagnosis model, if 90% of sick men and 80% of sick women are correctly diagnosed, then $EOD = 0.10$; in this case, women experience more false negatives.

$$EOD = P(\hat{Y} = 1 \mid A = 1, Y = 1) - P(\hat{Y} = 1 \mid A = 0, Y = 1) \quad (4.11)$$

4.7.2.4 Equalized Odds

Equalized Odds (EO) balances both TPR and FPR (False Positive Rate). In other words, it ensures that both true and false positive rates are similar for the two groups in Equation 4.12. For example, when the positive/negative classification errors are equal for both male and female candidates in the recruitment model, EO is achieved.

$$EO = |TPR_1 - TPR_0| + |FPR_1 - FPR_0|,$$

where $TPR_a = P(\hat{Y} = 1 \mid A = a, Y = 1)$, $FPR_a = P(\hat{Y} = 1 \mid A = a, Y = 0)$.

(4.12)

4.7.2.5 Disparate Impact

Disparate Impact (DI) is a ratio-based measure, measuring the proportional relationship between positive decision rates. Known in the industry as the “80% Rule”: if $DI \geq 0.8$, the risk of discrimination is considered low. For example, if a model has 40% approval for women and 50% approval for men, $DI = 0.8$.

$$DI = \frac{P(\hat{Y} = 1 \mid A = 1)}{P(\hat{Y} = 1 \mid A = 0)} \quad (4.13)$$

4.7.2.6 Predictive Parity

Predictive Parity (PP) compares positive classification precision. In the credit score model, if 70% of approved women are actually good customers, while 75% of approved men are good customers, then $PP = -0.05$, indicating a slight negative bias in favor of women.

$$PP = P(Y = 1 \mid \hat{Y} = 1, A = 1) - P(Y = 1 \mid \hat{Y} = 1, A = 0) \quad (4.14)$$

4.8 Implementation Framework

The entire project was developed by using Python 3.10 with the Apple M2 MacBook Air machine. For the implementation of ML models in the experiments, scikit-learn [61] and XGBoost [62] were preferred as the essential libraries; SMOTE and RandomUnderSampler methods of the imbalanced-learn [63] package were used for data sampling steps. Feature engineering and dimensionality reduction were also performed with the scikit-learn infrastructure. Hugging Face [64] and PyTorch [65] libraries were also used for easy handling of the datasets in the experiments. The Flower [38] (flwr) framework was used to simulate FL experiments and the fairlearn [66] library was used for fairness oriented evaluations. All code, experiment configurations, and documentation are published in the GitHub repository at <https://github.com/Mstfakts/Federated-Learning-Comparative-Study>.

5. EXPERIMENTS

This section presents the experimental framework designed to empirically validate the methodologies discussed in the previous sections. Our study is organized into two distinct, complementary experiments, each of which addresses a critical aspect of the application of Federated Learning to financial credit risk assessment. The first experiment, detailed in Section 5.1, is a large-scale comparative study focused on model performance. Here, we systematically assess the predictive power of FL-based risk models relative to traditional centralized approaches. The second experiment, presented in Section 5.2, shifts the focus from raw performance to model fairness. This section describes in detail an application specifically designed to quantify and mitigate demographic bias in credit risk assessment in a FL environment.

5.1 Federated Learning Comparative Study For Credit Risk Assessment

This experiment demonstrates an FL-based risk prediction model which enables collaborative training of models without requiring data exchange to evaluate credit card default risk through debt and payment information. Our experiments utilized the DCCC [9], HCDR [1] and HMEQ [10] datasets which we described in Section 3 and data pass through preprocessing steps. We also employed sampling techniques, namely the SMOTE and RUS. To reduce the complexity and improve model performance, we used two feature reduction methods that are PCA and kBest feature selection. The research employed five ML algorithms which included LR, MLP, SVM, XGB and RF. The research examined FL effects through experiments using two separate aggregation methods.

5.1.1 System architecture

In this section, we provide a detailed explanation of the data splitting strategy that makes CML datasets usable in FL environments, which are critical for experiments to predict consumer payment behavior and identify default risks, the pipeline used in training and testing phases, and the overall system architecture required for the experiment.

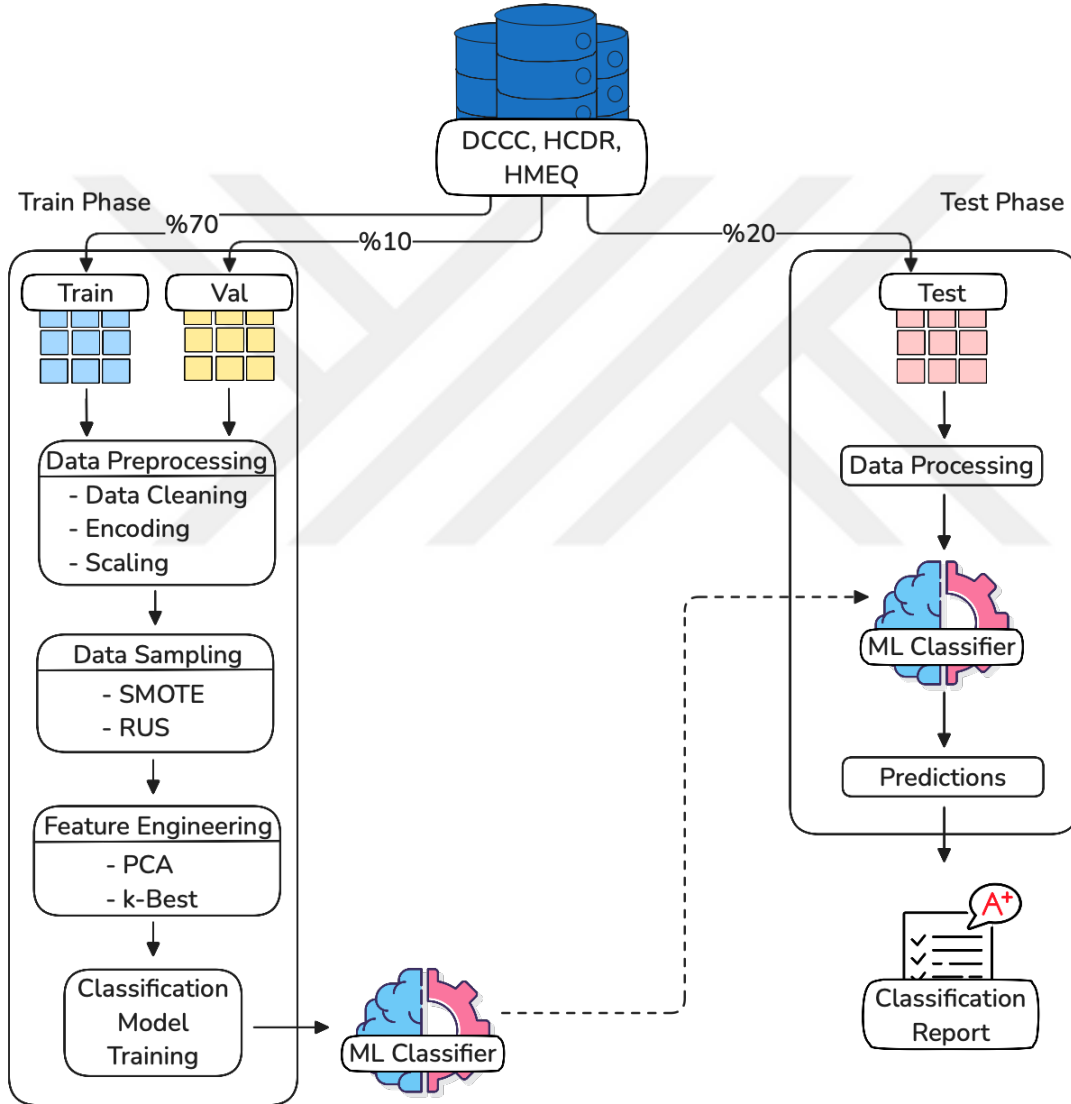


Figure 5.1 : Illustration of the proposed experimental setup.

Figure 5.1 illustrates the training and testing process for ML models. The dataset was split into training and testing subsets to prevent data leakage

and ensure a fair model evaluation. The datasets were cleaned by applying preprocessing and feature reduction methods to eliminate irrelevant features and decrease complexity. Five classifiers were trained with hyperparameter tuning and evaluated 10 times using client-specific test sets with different random seeds to ensure robustness. This study focused on predicting credit card defaults, specifically assessing the performance of Class-1 as shown in Table 4.1, which represents instances of default. The whole study was founded on a FL philosophy, as discussed in Section 4.4. This implies that the methods for data processing and all of the algorithms employed in Section 4.1 are implemented subsequent to the dataset’s distribution among the clients. Thus, the dataset and the operations associated with it are private to each client.

5.1.2 Train test configuration

To avoid data leaks and guarantee that model evaluation is impartial and reliable, we argue that test and validation sets must remain entirely separate at every stage of the study. One critical point here is that we only applied the data preprocessing, sampling, and feature reduction steps to the train set before moving on to the validation and test sets. It means that we also considered the scenario of not being able to access test data in real life. We did not apply these steps to the entire dataset at once.

Similarly, to ensure data privacy in accordance with FL, we use a client-specific data preprocessing approach when running FL-based models. In other words, we allocate data to each client, perform preprocessing only on their local data, and do not allow the global model to access or directly train on the clients’ data. The dataset was divided into three subsets: train (70%), validation (10%), and test (20%). To manage the randomization and ensure the experiments could be repeated, we used the *train_test_split* function with the *random_state* parameter.

We partitioned the dataset into clients with unique *random_states* to ensure independent data partitioning for each client. This strategy aims to replicate real-world FL scenarios as closely as possible [67]. To model real-world

data distributions, we used the Flower framework’s *DirichletPartitioner* component [38]. Each client used scaling and encoding techniques only on its own datasets. Additionally, we executed all steps specified in this study on a client-by-client basis.

5.2 Federated Learning Fairness In Finance: An Application To Credit Risk Assessment

In the second part of this study, we focused on quantifying and improving the level of fairness of model outputs among demographic groups on the DCCC dataset. In this section, the LR model with FairFed approach is used with the DCCC dataset consisting of 30,000 samples. The results will be presented in a way that reveals the performance-fairness trade-off. Thus, a comparative evaluation of methods that minimize discrimination against different demographic groups in real-world applications of credit scoring models will be made.

5.2.1 System architecture

The system architecture is shown in Figure 5.2. At this stage, a synthetic attribute called “Interest” was added to the DCCC dataset for fairness analysis as mentioned earlier (see Section 3.5 for details). In order to create a data distribution with different levels of unfairness for each client thanks to the synthetic attribute, the gender ratios (Table 5.1) are predetermined and a manual partitioning method is used instead of *DirichletPartitioner*. Each client dataset was split into 70% training, 20% testing and 10% validation, and the split portions were subjected to the data preprocessing steps described earlier. In order to guarantee the reproducibility of the experiments, the random partitioning process using the `train_test_split` method was retained in this section as in the first experiment.

The LR model, which has shown strong performance in benchmark results, was preferred for performance evaluations. The outputs of the model are analyzed based on the fairness metrics defined in Section 4.7.2.3. Furthermore, a new aggregation method called “FairFed” was added to the project code to integrate

Table 5.1 : Sex distribution rates in each client.

Client ID	SEX=1 (Male)	SEX=2 (Woman)
0	0.20	0.80
1	0.30	0.70
2	0.40	0.60
3	0.50	0.50
4	0.70	0.30

the approach in the FairFed paper. In this method, the EOD values obtained from the clients are weighted according to their distance from the global EOD value; the client closest to the global value is assigned the highest weight, while the farthest client is assigned the lowest weight. Thus, the inequity effects between different clients were quantified and optimized.

Our experiments on fairness were conducted in three different scenarios according to how the data was divided: Local, Centralized and Federated (Dervişoğlu, 2024 [68]). In the Local experiment, each client trained and tested its model with the CML method with only its own data, just like an independent bank; in this case, FL was not used. In the Centralized approach, the training data of the clients were combined in a single pool and a common model was trained, but the model was evaluated on each client’s own test and validation sets. In the Federated scenario, each client completed its training locally, holding its own data, in line with the FL protocol, and the results were obtained with its own test data. A summary of the approaches is given in Table 5.2. The aim here was to better observe the effects of different approaches. Accordingly, theoretically, since each client uses the entire train dataset in the Centralized approach, it should give lower EOD values than the Local approach. In addition, the Federated approach is expected to give as good results as the Centralized approach, although the clients do not have access to all the data, and it is expected to reach much lower EOD levels than the Local approach. In addition, as stated in Table 5.2, since the train_test_split process was performed randomly on a client basis, each approach was tested 100 times and the average was taken for the robustness of the experimental results.

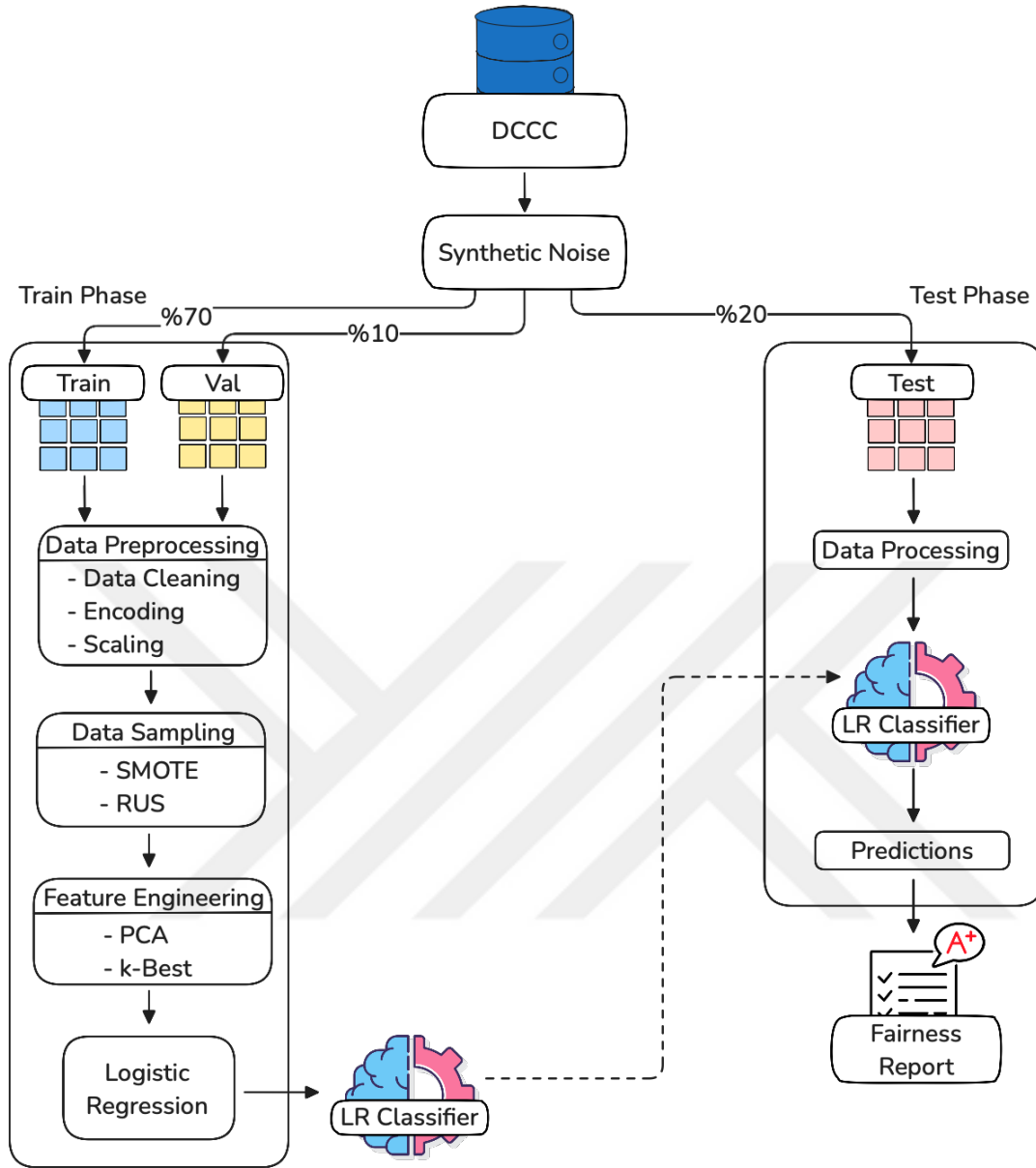


Figure 5.2 : Illustration of the proposed FL fairness experimental setup.

Table 5.2 : Different approaches for fairness experiments.

	Local Approach	Central Approach	FL Approach
Learning Paradigm	Central ML	Central ML	FL
Train set	Client based	Combined client	Client based
Test set	Client based	Client based	Client based
Validation set	Client based	Client based	Client based
Repeat number	100	100	100

6. RESULTS & DISCUSSION

In this section, we first show in the first experiment of this study that FL methods that provide data privacy requirements in the financial data offer similar accuracy levels without significant performance loss compared to CML approaches. In our comparisons on three different datasets, the classical FedAvg algorithm as well as the FedF1 aggregation method we developed sometimes achieved superior or at least equivalent performance results compared to FedAvg, especially under imbalanced data distribution conditions. In the second experiment, we focused on the risk of bias in financial models across demographic groups and found that some demographic groups may face favorable or unfavorable classification results at the end of the training process. To overcome this problem, we applied the FairFed approach and quantitatively assessed the improvement in fairness metrics by weighting model outputs on the basis of Equal Opportunity. Finally, we present the results of our experiment in this section.

In order to increase the readability of this section and focus on the main conclusions, visuals and comments summarizing the main trends and critical findings are presented in the upcoming sections. Detailed and comprehensive result tables containing all metrics for each experimental configuration are presented in the Appendices Section of the thesis for interested researchers and readers to conduct in-depth examination. Accordingly:

- For all results belonging to the DCCC dataset, Table A.1 in Appendix-A,
- For all results belonging to the HCDR dataset, Table B.1 in Appendix-B,
- For all results belonging to the HMEQ dataset, Table C.1 in Appendix-C can be referred.

All discussions and conclusions in this section are built on the main trends and statistical findings obtained from these detailed data presented in the appendices.

6.1 Federated Learning Comparative Experiment Results

In this section, the findings of the first experiment of the study are presented in detail. The main purpose of the analysis is to quantitatively compare the performance of the traditional CML approach and FL methods. This comparison also covers the performance of the FedAvg and FedF1 that we propose within this thesis. The results of the experiments conducted on three different financial data sets (DCCC, HCDR, and HMEQ) are examined in terms of metrics such as Precision, Recall, F1 Score and Accuracy. Also, the applicability of FL over CML and the potential superiority of the FedF1 method over standard approaches are examined.

The analyses of this section are presented under four subheadings. First, in the Section 6.1.1, the models that achieved the highest performance and the scores they achieved as a result of the comprehensive experiments conducted are examined through Table 6.1 and Figure 6.1. Then, in the Section 6.1.2, the performance of FL approaches is analyzed compared to the CML reference model using violin plots (Figure 6.2) showing the distribution of all experimental results. After, in the Section 6.1.3, the effectiveness of the proposed FedF1 method is evaluated through scatter plots (Figure 6.3) providing a one-to-one performance comparison with the standard FedAvg approach. Finally, in the Section 6.1.4, the effects of preprocessing techniques on model performance are examined with heat maps (Figure 6.4) showing the percentage change compared to the baseline model.

6.1.1 Overall performance review: Top models summary

In this section, the models that exhibit the highest performance for each experiment and the scores they achieved are examined as a result of the extensive experiments we conducted. This analysis is performed through the summary results presented in Table 6.1 and the explanatory heatmaps shown in Figure 6.1. The main goal is to reveal the general performance trends and prominent model architectures among different aggregation approaches, datasets, and metrics.

Table 6.1 : Best performance results and models by dataset

Dataset	Metric	Centralized (CML)		FL (FedAvg)		FL (FedF1)	
		Model	Score	Model	Score	Model	Score
DCCC	Precision	LR+kBest	0.73	LR+kBest	0.75	LR+kBest	0.71
	Recall	MLP+SMOTE	0.74	LR+RUS*	0.74	LR+RUS*	0.75
	F1	XGB+kBest+SMOTE*	0.53	RF+RUS*	0.52	RF+kBest+SMOTE*	0.52
	Accuracy	XGB+kBest*	0.82	MLP+kBest*	0.82	MLP+kBest*	0.82
HCDR	Precision	XGB	0.27	XGB+PCA	0.32	XGB+PCA	0.33
	Recall	XGB+PCA+RUS	0.99	XGB	0.88	RF+PCA+RUS	0.69
	F1	XGB	0.31	RF	0.32	XGB+kBest	0.89
	Accuracy	SVM+PCA+SMOTE*	0.92	XGB+PCA*	0.92	XGB+kBest*	0.92
HMEQ	Precision	RF	0.79	MLP	0.92	RF	0.88
	Recall	XGB+kBest+RUS	0.77	RF+RUS	0.88	RF+RUS	0.87
	F1	RF+kBest+RUS	0.77	RF+SMOTE	0.81	RF+SMOTE	0.82
	Accuracy	RF+SMOTE	0.95	RF+SMOTE	0.93	RF+SMOTE	0.93

* There are multiple models that achieve the same metric score.

When the results in Table 6.1 are examined, the first finding is that the FL (FedAvg and FedF1) approaches are highly competitive against the traditional CML approach. Despite offering a significant advantage such as protecting data privacy, it is seen that these approaches do not make a serious compromise in terms of performance and even achieve superior results in some scenarios.

The FedF1 approach has achieved remarkable results on multiple datasets. For example, it has left other methods (0.74) behind with its Recall score of 0.75 in the DCCC data set. This approach has also reached the highest result with a Precision value of 0.33 in the HCDR data set. Similarly, it has outperformed the centralized CML approach (0.77) with its performance of 0.82 in the F1-Score metric in the HMEQ data set. Our experiments showed that another FL method, FedAvg, also exhibited remarkable success. This method achieved the highest Precision value with 0.75 in the DCCC dataset. It reached both the highest Precision with 0.92 and the highest Recall with 0.88 in the HMEQ dataset. When the general results are evaluated, it is seen that FL approaches have a significant superiority over CML approaches. FL models (FedAvg and FedF1) obtained the best score in 7 of the 12 comparisons made on three datasets and four different metrics. The CML approach gave the most successful result only in 3 comparisons, while all methods received equal scores in the remaining 2 cases.

The performance, as expected, varies greatly depending on the nature of the datasets. The HMEQ dataset shows the most stable dataset profile with high and balanced scores (mostly in the range of 0.77-0.95) obtained by all approaches, while the HCDR dataset represents the most challenging scenario. The very low Precision (0.31) and misleadingly high Accuracy (0.92) scores in HCDR show how big a problem extreme class imbalance is in this dataset. Under these challenging conditions, the remarkable success of the FedF1 approach, especially in terms of F1-Score of 0.89, is an important finding that shows how effective this method can be compared to the standard FedAvg (0.32) and CML (0.31).

This proves that FL is a powerful alternative not only theoretically but also practically for areas with sensitive data such as financial risk modeling.

The heatmaps in Figure 6.1 provide a deeper analysis opportunity by converting the numerical results into visual patterns. The color intensities represent the magnitude of the scores, while the text in the cells indicates the best model configuration.

When the heatmaps are examined, it is seen that tree-based ensemble models such as RF and XGB are generally overwhelmingly dominant. The superiority of these models in capturing complex and nonlinear relationships in tabular data has enabled them to be among the best performing models in almost every experiment. In particular, the fact that the HMEQ dataset is almost completely dominated by RF-based models reveals the success of this algorithm in the relevant problem.

Another finding, which is at least as important as the model itself, is the critical role of the data preprocessing steps. The best performing models are rarely the "base" (no additions made) models. In three datasets, four different metrics and three different aggregations, the basic models achieved the best score in only 7 out of 36 experiments.

In particular, it is almost always possible to see +SMOTE or +RUS suffixes in the cells with the highest Recall and F1 scores. Data sampling was applied in half of the 36 experiments, 18 of them (50%), and remarkably, 14 of these 18 experiments

(78%) achieved the best Recall and F1 scores. This proves that combating class imbalance is indispensable for the success of catching default cases.

The experimental results show that feature engineering plays a critical role in improving the Accuracy and Precision metrics in particular. In fact, 11 of the 18 experiments (61%) where feature selection (+kBest) or extraction (+PCA) were applied achieved the highest Accuracy and Precision values. The most concrete examples of this effect are seen in the models such as LR+kBest and XGB+kBest being at the top of their categories. This confirms that selecting the most meaningful features directly increases both the overall accuracy of the model and its consistency in positive predictions (Precision).

These results reveal several key implications. First, FL has proven to be an alternative by offering competitive performance compared to CML. Second, there is no 'one best model' to achieve the best results. Instead, success depends on a smart combination of the characteristics of the dataset, the targeted metric, and data preprocessing techniques appropriate for this goal (sampling and feature selection, especially for class imbalance). Finally, the FedF1 aggregation method proposed in this study has been shown to have significant potential compared to standard methods, especially in challenging and imbalanced federative scenarios such as HCDR. In the following section, we will go into more detail and examine the performance comparisons of FedAvg and FedF1 under FL.

6.1.2 Comparison of centralized and federated learning approaches

In this section, the performance of the FL approaches (FedAvg and FedF1) compared to the CML reference model is analyzed comparably. The main purpose of this analysis is to determine whether the data privacy and security advantages brought by FL create a trade-off in the model performance. The analysis is performed by taking into account the dynamics of each metric through violin graphs (See Figure 6.2) showing the distribution of all experimental results.

When the four graphs in Figure 6.2 are examined as a whole, the first and most important conclusion is that FL approaches exhibit a highly competitive

performance compared to the centralized model. It is seen that FL models trained in a distributed manner, without the need to collect data on a single server, do not significantly lag behind the CML model, which can access all the data simultaneously. This is a critical finding for the practical applicability of FL.



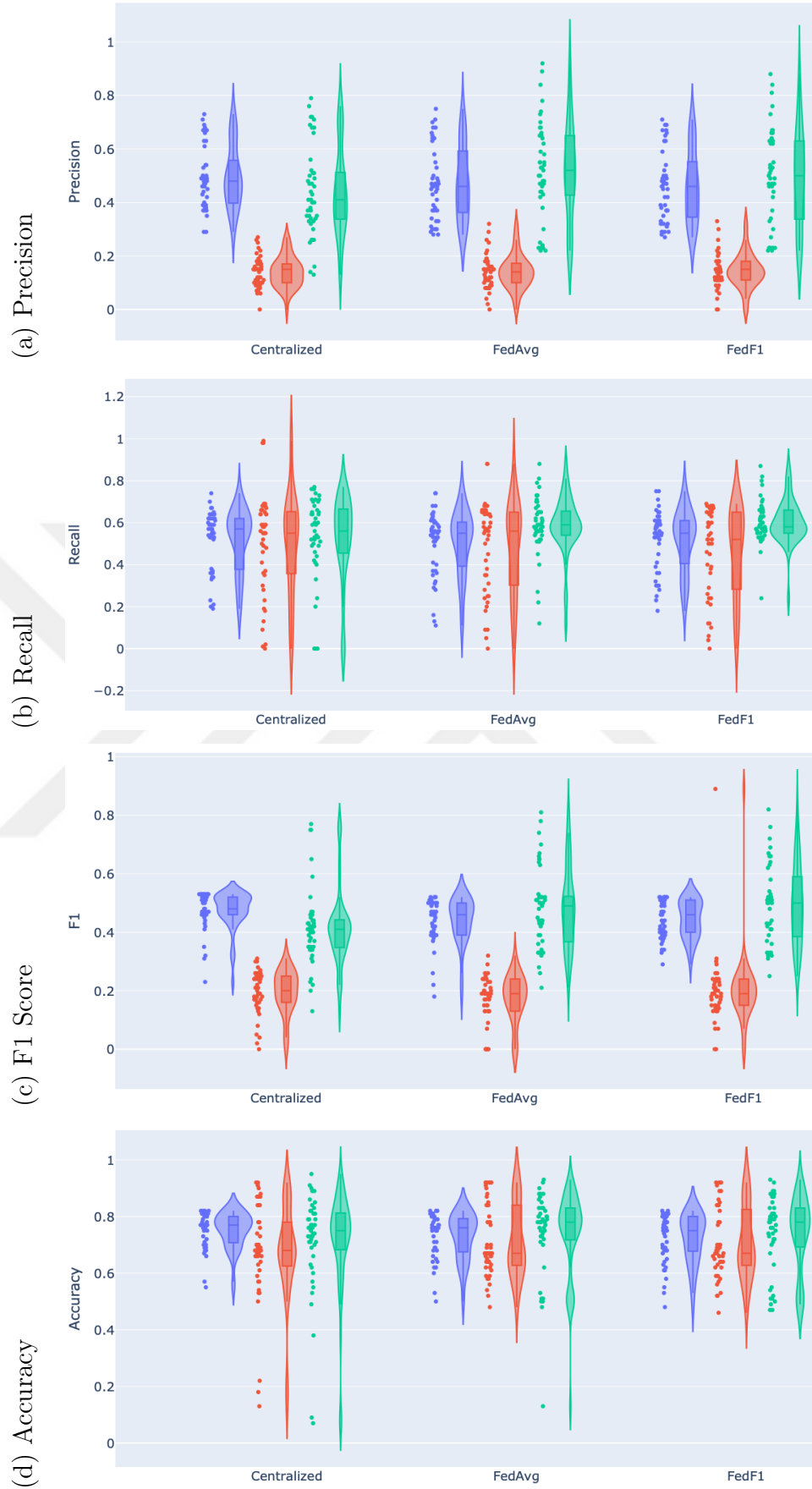


Figure 6.2 : Comparison of CML, FedAvg, and FedF1 on the DCCC (purple), HCDR (red), and HMEQ (green) datasets using metrics: (a) Precision, (b) Recall, (c) F1-Score, and (d) Accuracy.

When examined on a metric basis, it becomes clear that this competition contains some nuances. When looking at the Precision graph (Figure 6.2-a), it is seen that the median value of the CML violin is slightly higher than the FL approaches. This suggests that the central model can control the false positive rate more effectively by seeing the general distribution of the entire data set and thus make more "precise" predictions.

However, this situation is exactly the opposite for other metrics. Especially in Recall (Figure 6.2-b) and F1-Score (Figure 6.2-c) graphs, it is clearly observed that the median value of the FedF1 approach is higher than CML. This shows that FL, especially with a smart aggregation strategy like FedF1, can be more successful than the centralized approach in detecting the minority class (default cases) and establishing the Precision-Recall balance. This is a sign that FL is not only a privacy solution but also a model improvement tool with the potential to improve performance on certain tasks.

The Accuracy plot (Figure 6.2-d) shows that all three approaches have very similar distributions and median values, confirming that FL is as successful as the centralized model in preserving the overall accuracy rate.

When the dataset dynamics and performance are examined, the following results stand out:

The superiority of FL approaches over CML is most evident in the HMEQ dataset (Green Points). Especially in the F1-Score and Precision graphs, it is seen that the green points are concentrated in a higher position within the FL violins compared to the CML violin. In addition, the maximum and median values reached by the green points are higher than the maximum and median values reached by CML. This shows that in a well-structured problem, distributed learning can produce better results than centralized learning.

The HCDR (Red Dots) dataset is the most challenging dataset. The scores of all approaches (especially F1 and Precision) are concentrated in a low range. The performance difference between CML and FL is less obvious on this dataset because all methods struggle with the difficulties of the problem.

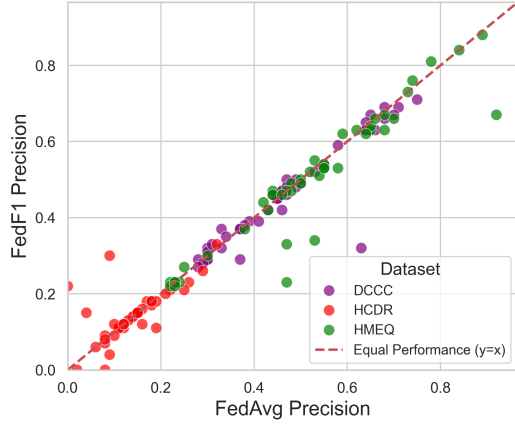
DCCC (Purple Dots) represents the scenario where the most balanced and competitive results are obtained among the three approaches. The performance distributions largely overlap, indicating that FL performs equivalently to CML on this standard benchmark dataset.

In conclusion, this comparative analysis shows that the key advantages of FL, such as privacy, come at an acceptable and often negligible performance cost. In fact, when a novel aggregation strategy such as FedF1 is used, FL has the potential to outperform centralized approaches on critical metrics such as F1 and Recall. These findings provide strong evidence that financial institutions can develop regulatory compliant and high performance credit risk models without centralizing their sensitive data.

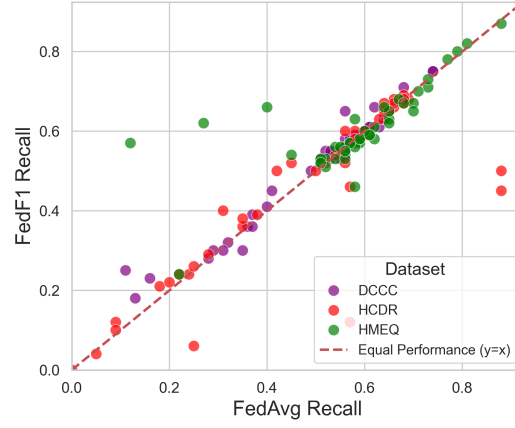
6.1.3 Analysis of aggregation methods: Comparison of FedAvg and FedF1

In this section, the effectiveness of the FedF1 method, which is proposed in the context of this study and provides a weighting mechanism based on model performance, is directly compared with the standard approach in the literature, FedAvg, within the FL framework. The main purpose of this comparison is to clearly demonstrate which aggregation method produces superior results under the same experimental conditions. The analysis is performed through scatter plots (Figure 6.3) that match the performance of each experimental configuration under the two methods one-to-one. In these plots, the Y -axis represents the FedF1 score, the X -axis represents the FedAvg score, and the $y = x$ reference line indicates equal performance. The points above this line indicate that FedF1 is more successful than FedAvg in that experiment.

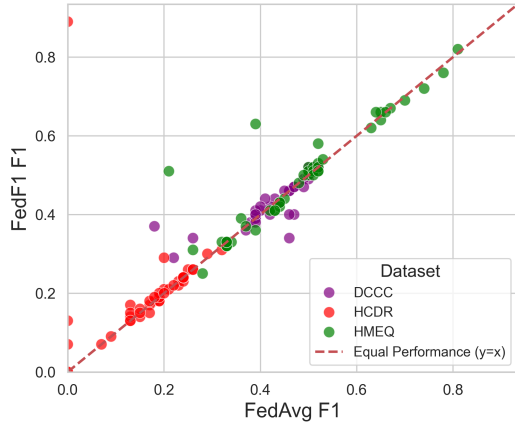
The strongest and most important findings of our study emerge in the F1-Score (Figure 6.3-c) graphs, which reflect the design purpose of the method, and the Recall (Figure 6.3-b) graphs, which directly affect this score.



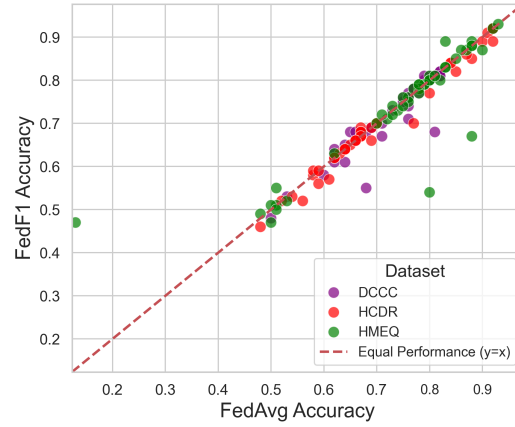
(a) Precision



(b) Recall



(c) F1 Score



(d) Accuracy

Figure 6.3 : Performance comparison of FedAvg and FedF1 methods according to metrics: (a) Precision, (b) Recall, (c) F1 and (d) Accuracy

The F1-Score plot visually demonstrates the performance difference between the two methods. A significant portion of the points in the plot lie on or exactly above the $y=x$ reference line, indicating that FedF1 is a competitive method. This visual finding is also confirmed numerically: FedF1 prevails in 47 out of 135 experiments, while FedAvg outperforms only in 37 experiments. This shows that the strategy of weighting client updates by F1 score positively affects the ability of the global model to balance Precision and Recall.

The main reason behind this success is clearly seen in the Recall chart. FedF1 is significantly better at catching defaults than FedAvg. This is also proven quantitatively; FedF1 outperforms FedAvg in 60 out of 135 experiments, while

FedAvg produces a better Recall score in only 41 experiments. By rewarding the contribution of expert clients who have a high F1 score in that round (and therefore usually have a high Recall value), FedF1 reduces the probability of 'missing a risky client' (False Negative), one of the most critical errors in the global model.

Interventions to increase the Recall value of a model can often lead to a trade off that results in a decrease in Precision. The Precision plot (Figure 6.3-a) confirms this. Numerical analysis shows that out of 135 experiments, FedAvg achieves a higher Precision score in 59, while FedF1 achieves a higher Precision score in 35. This suggests that FedAvg is generally more 'precise' in its positive predictions, but this is an expected trade off in the face of FedF1's large advantage in Recall. Although FedF1 lags behind in this metric, the fact that the points in the scatter plot are very close to the $y=x$ line suggests that this performance difference is not large, and that FedF1 achieves its large gain in Recall without a significant loss in Precision.

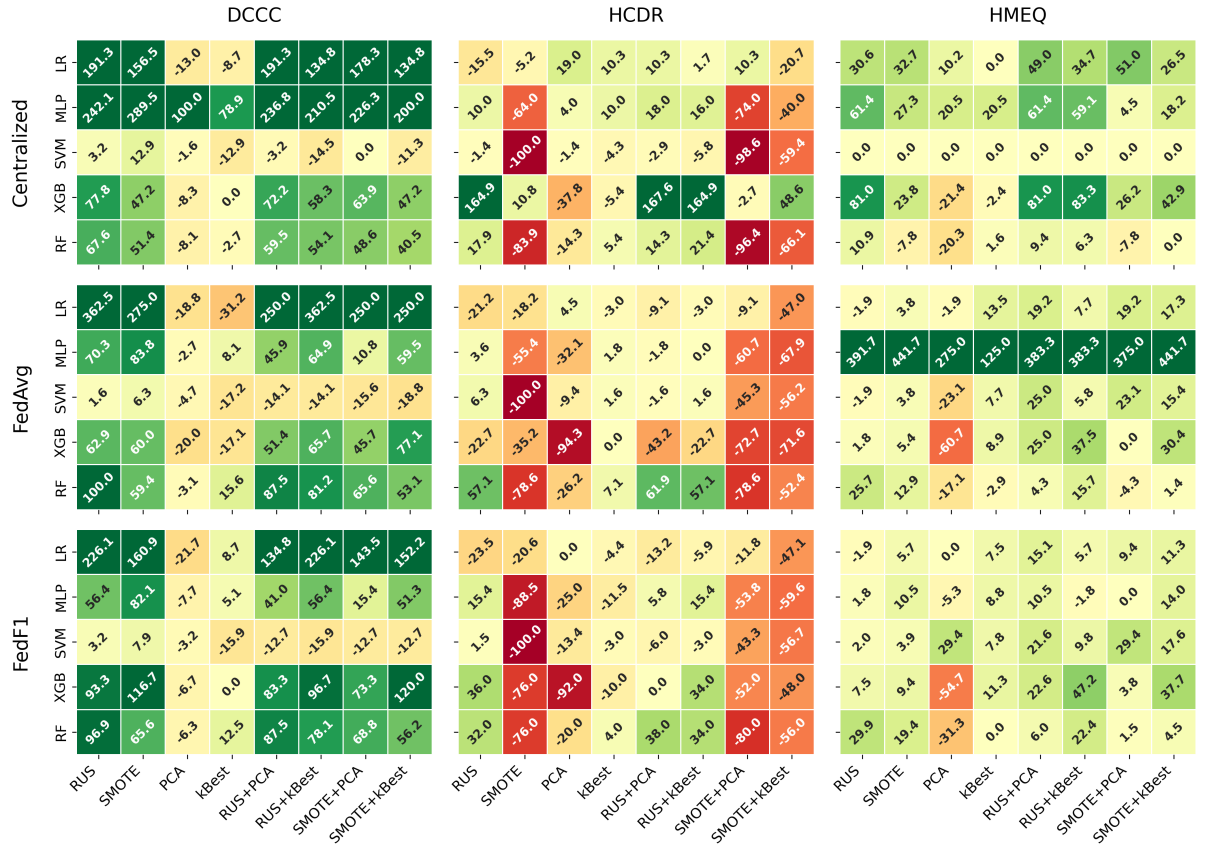
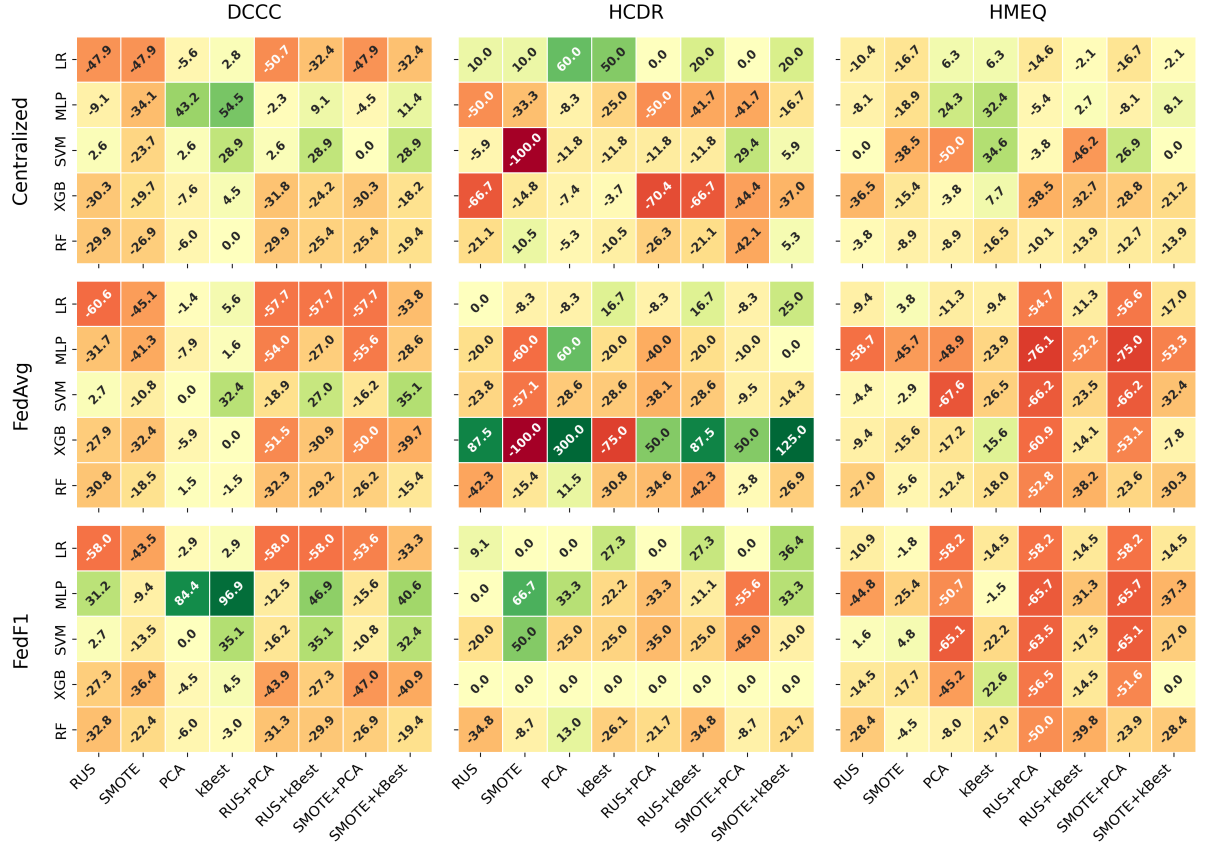
Similarly, the Accuracy plot (Figure 6.3-d) shows that both methods perform very closely in terms of overall accuracy. In 64 out of 135 experiments (47%), both methods achieve exactly the same accuracy score. In the remaining experiments, the superiority scores (FedAvg: 39, FedF1: 32) are very close to each other. This proves that FedF1 performs on par with the standard approach, especially on imbalanced datasets, while making a difference in more critical metrics that reflect the essence of the work.

The results clearly show that the FedF1 approach is superior, especially in problems with heterogeneous and imbalanced data, such as financial risk modeling. FedF1 consistently produces better results in the F1-Score and Recall metrics, while maintaining competitive performance in the Precision and Accuracy metrics, compared to the standard FedAvg method. These findings suggest that FedF1 is a more robust and superior aggregation strategy for real-world FL scenarios.

6.1.4 Impact of data preprocessing techniques on performance

The final success of ML models is dependent not only on the power of the chosen algorithm, but also on the quality of the preprocessing steps applied to the data. This section examines the systematic effects of four essential preprocessing techniques (Data Sampling (SMOTE, RUS) and Feature Reduction (PCA, kBest)) and their combinations on model performance. This analysis is performed using performance change heatmaps (Figure 6.4) that show the percentage change in the relevant metric for each technique compared to the baseline model without any preprocessing. In these maps, green tones represent an increase in performance, while red tones represent a decrease.

The most striking finding across all metrics and learning approaches is that the impact of preprocessing techniques is not uniform. None of the techniques appears to provide universally optimal results, with the impact largely dependent on aggregation methods, datasets, and ML algorithms used.



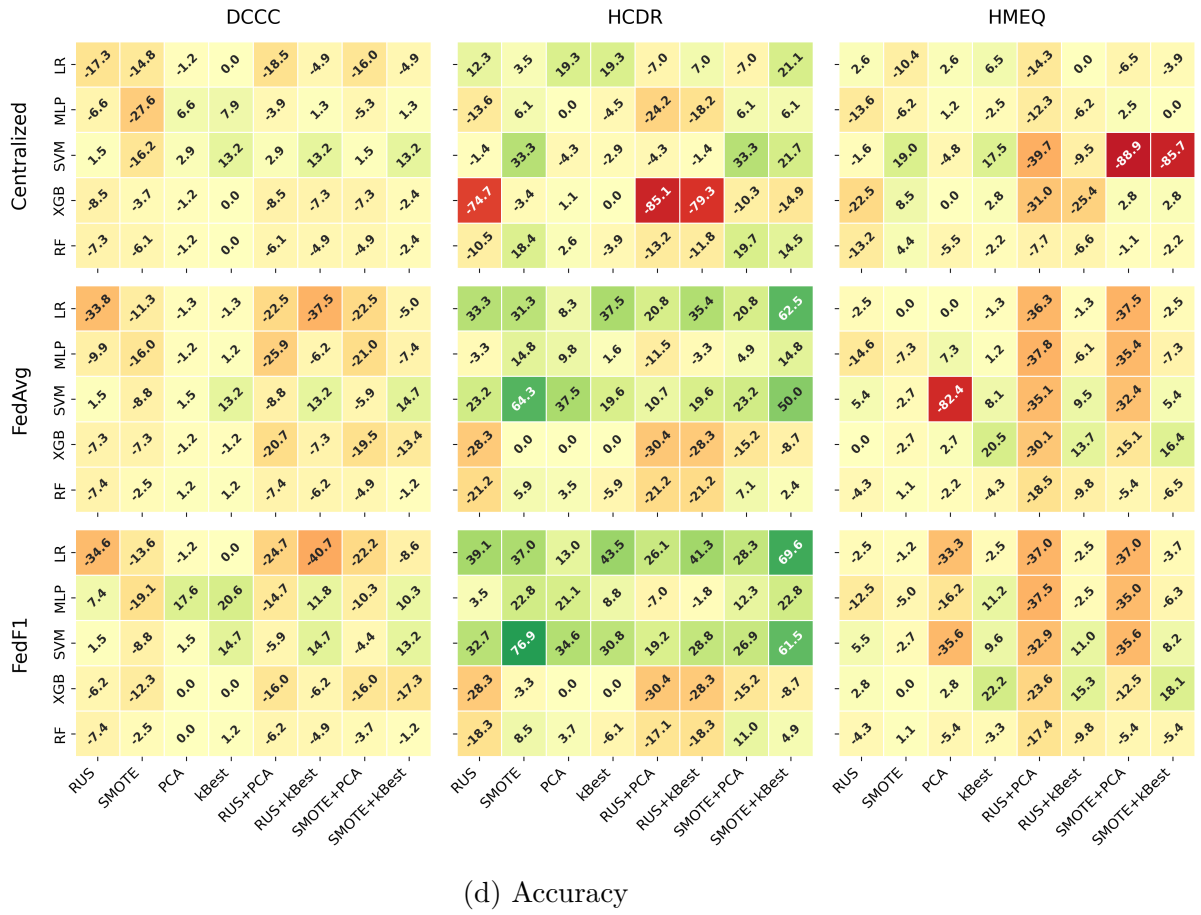
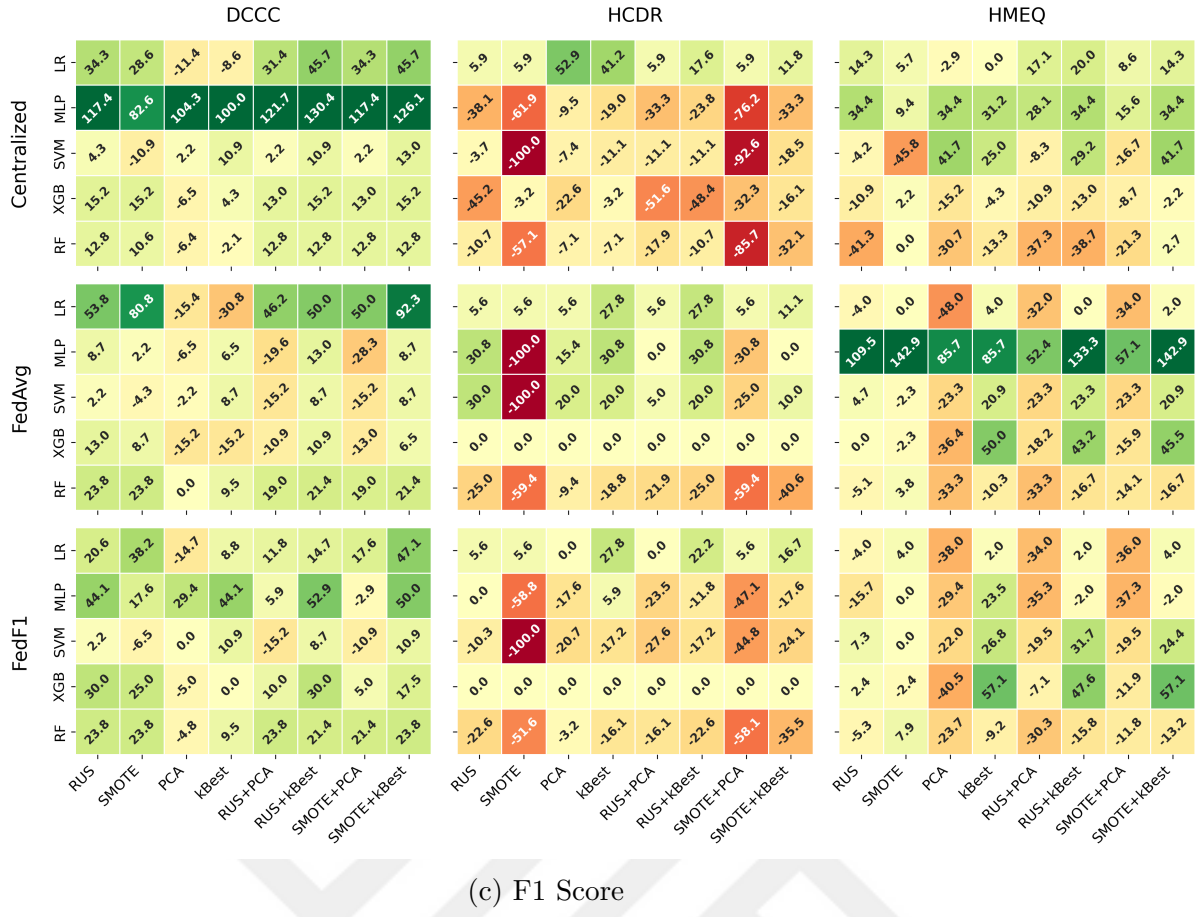


Figure 6.4 : Percentage impact of preprocessing techniques on metrics: (a) Precision, (b) Recall, (c) F1 and (d) Accuracy (Relative to the baseline model)

Recall: The Recall Heatmap (Figure 6.4-b) demonstrates the impact of data sampling techniques on the model’s ability to detect the minority class. Specifically, when oversampling techniques such as SMOTE and undersampling techniques such as RUS are applied, remarkable increases in the Recall metrics are observed in most scenarios.

This effect is most evident in scenarios where the initial Recall score of the base model is low, i.e., where the potential for improvement is high. For example, on the DCCC dataset, SMOTE application for the LR model under the FedAvg approach increased the Recall score by a massive +275.0%. Similarly, on the HMEQ dataset, the RUS technique for the MLP model under the same approach achieved a remarkable +391.7% success. This proves that combating class imbalance significantly reduces the False Negative rate, which is the most costly error for a financial institution.

However, this performance increase is directly related to the structure and difficulty level of the dataset used and is not always guaranteed. The most interesting example of this situation is seen in the HCDR dataset. Contrary to general expectations, the SMOTE technique and its derivatives (PCA+SMOTE, kBest+SMOTE) decreased the Recall score (red cells) in almost all models in the HCDR dataset and negatively affected the performance. For example, the SMOTE application for the RF model in the Centralized approach decreased the performance by -83.9%. The possible reason for this may be that the minority class in HCDR has a very complex and noisy structure. Since SMOTE creates synthetic new points among the existing minority examples, it may have produced invalid examples in this noisy structure that confused the model, disrupted the decision boundary, and made learning difficult.

On the other hand, it is important that the RUS technique consistently gives positive (green) results in the same difficult HCDR dataset. For example, RUS increased the Recall by +164.9% for the XGB model in the Centralized approach. Since RUS only reduces the majority class instead of generating new synthetic data, it forces the model to focus on the existing but small number of real minority

examples. This strategy is seen to be more successful in a complex and noisy dataset like HCDR.

In conclusion, this analysis shows that there is no "one and only" truth in the effectiveness of data sampling techniques. While both SMOTE and RUS were quite successful in more stable datasets like DCCC and HMEQ, RUS turned out to be a much more reliable and effective strategy compared to SMOTE in a difficult problem like HCDR. This finding revealed how carefully the specific characteristics of the problem and the dataset should be analyzed when choosing a preprocessing technique.

Precision: The Precision Heatmap (Figure 6.4-a) shows the impact of preprocessing steps on the Precision metric, which measures how many of the predictions the model labels as risky are actually correct. This analysis, when considered together with the Recall metric examined in the previous section, highlights one of the most fundamental trade-offs in machine learning: the Precision-Recall trade-off.

In stark contrast to the Recall analysis, it is clear that the RUS and SMOTE columns are overwhelmingly red in the heatmap. This confirms the general negative impact of data sampling techniques on Precision. When the model takes a more aggressive stance (i.e. increases Recall) to better capture the minority class, the risk of generating False Positives also increases. In other words, while the model labels more customers as risky, some of these labels actually hit non-risky customers.

The most striking examples of this effect are seen in the HMEQ dataset. For the MLP model under the FedAvg approach, the RUS application significantly reduced the Precision score by -58.7%. Similarly, the SMOTE technique negatively impacted the performance by -38.5% for the SVM model under the Centralized approach. These findings clearly show that the cost of the increase in Recall is a decrease in the accuracy of positive predictions.

F1 Score: The F1-Score Heatmap (Figure 6.4-c) shows the net result of the trade-off between the Precision and Recall metrics, showing the final impact of

the preprocessing techniques on the overall balanced performance of the model. Since the F1-Score is the harmonic mean of these two metrics, it is one of the most reliable indicators for understanding whether a technique really works.

As observed in the previous sections, data sampling techniques increase Recall while decreasing Precision. The F1-Score heatmap proves that the result of this equation is overwhelmingly positive. The RUS and SMOTE columns are green, especially in the DCCC and HMEQ datasets. This shows that the increases in the Recall metrics more than offset the decreases in the Precision metrics, resulting in a net performance gain.

The clearest examples that demonstrate this are: In HMEQ Dataset, RUS technique for MLP model under FedAvg approach provided an increase of +109.5% in F1-Score. This is mathematical proof that +391.7% gain in Recall dominates -58.7% loss in Precision in the same scenario. In DCCC Dataset, RUS technique for MLP model under Centralized approach provided a +117.4% improvement in F1-Score. This shows that even on a more balanced dataset like DCCC, sampling techniques have the potential to double the overall performance of the model.

On the other hand, the effect of feature reduction techniques on F1-Score is less clear. PCA columns are usually in red tones, especially in HCDR and HMEQ datasets. This shows that PCA can negatively affect both Precision and Recall by causing information loss, thus causing a net decrease in F1-Score.

kBest, on the other hand, has a more neutral profile. Although there are positive examples such as the kBest application increasing F1-Score by +100% for the Central MLP model on the DCCC dataset, it does not generally have a large and consistent effect.

In conclusion, this analysis on F1-Score highlights the strategic importance of preprocessing steps. A high-performance model is not one that maximizes only one metric, but one that achieves the balance between Precision and Recall. This study shows that data sampling techniques play a good role in achieving

this balance positively, while feature selection is a powerful that can be used to further fine-tune this balance.

Accuracy: The Accuracy Heatmap (Figure 6.4-d) shows the effect of preprocessing techniques on the Accuracy metric, which shows the proportion of correct predictions of the model among all predictions. Unlike the other metrics, the color distribution in this heatmap is paler and more changing. This shows that the effect of preprocessing steps on the overall accuracy is more limited. The main purpose of this analysis is to explain the reliability problem of the Accuracy metric in imbalanced datasets and the effects of preprocessing steps on this metric.

In DCCC and HMEQ datasets, the sampling application usually causes a slight decrease in Accuracy (red tones). Although this may seem like a performance loss, it is actually an indication that the model is learning more fairly. The basic model can achieve high accuracy by predicting only the majority class. When sampling forces the model to learn the minority class as well, it starts making more mistakes in the majority class, which decreases the overall accuracy. This is an acceptable price to pay for the large gains in Recall and F1-Score.

Conversely, on the most unbalanced dataset, HCDR, the same techniques generally lead to a significant increase in Accuracy (green tones). The reason for this paradox is that the base model almost never catches default cases (generating a lot of False Negatives). When sampling allows the model to correctly predict even a few of these missed cases, this gain can outweigh the newly generated False Positive errors, increasing the overall accuracy score (TP+TN).

The effect of feature reduction techniques is also remarkable. While kBest generally produces a more neutral effect, PCA can sometimes have devastating results. The most obvious example of this is seen for the SVM model under the Centralized approach on the HMEQ dataset. The application of SMOTE+PCA resulted in a decrease in Accuracy score of -88.9%.

Our comprehensive analysis proves that the effect of preprocessing techniques on model performance is not one-dimensional, but rather strongly depends on

the nature of the dataset, the methods used, and the targeted success metric. Data sampling methods SMOTE and RUS have demonstrated their effectiveness by providing improvements in Recall and F1-Score metrics, especially in datasets such as DCCC and HMEQ. This increase more than compensated for the decrease in Precision metric, clearly increasing the overall model success. On the other hand, the experiments in the HCDR dataset revealed that not every technique can be applied directly, and that noisy and complex data structures can render (i.e. SMOTE) ineffective. As a result, this analysis has shown that it is more accurate to focus on balanced metrics such as F1-Score rather than misleading metrics such as Accuracy in imbalanced data classification problems. It also shows that it is more accurate to choose the preprocessing strategy to be used by making a careful analysis of the specific difficulties of the dataset. After these experiments provide a comprehensive analysis of model performance, a fairness analysis is performed in FL in the next section.

6.2 Fairness in Federated Learning Experiment

Table 6.2 presents the Equal Opportunity Difference (EOD) scores of different fairness experiments; a fixed gender ratio distribution was applied to each client. Thus, different degrees of unfairness were applied to each client. EOD values are given in the Local Approach, Central Approach and Federated Learning (FL) columns, with lower scores indicating fairer performance.

Table 6.2 : EOD scores for different approaches in fairness experiments.

ID	# Sample	Sex Ratio (1:2)	EOD _{Local}	EOD _{Central}	EOD _{FL}	$EOD_{F_{global}}$
0	6000	20:80	0.1658	0.1592	0.0395	-
1	6000	30:70	0.1355	0.1458	0.0286	-
2	6000	40:60	0.1023	0.1043	0.0241	-
3	6000	50:50	0.1212	0.1176	0.0241	-
4	6000	70:30	0.1971	0.1393	0.0252	-
Mean	-	-	0.1444	0.1332	0.0283	0.0190
Std	-	-	0.0375	0.0221	0.0067	0.0000

As a result of the Local Approach, Client-0 (20:80) and Client-4 (70:30), which are at the extreme values of the gender distribution, received the highest EOD scores;

this situation reveals that unfairness became evident in these clients. On the other hand, Client-2 (40:60) and Client-3 (50:50), where the distribution is relatively balanced, obtained the lowest EOD scores and achieved fairer results compared to other clients. In addition, since each client uses completely independent data, the standard deviation of EOD scores between clients in the Local approach was the highest (0.0375). This wide variation paved the way for the average EOD value of the Local method (0.1444) to be the highest compared to the other approaches.

In the centralized approach experiment, the standard deviation of the EOD scores was minimal (0.0221) because all clients used a common training set. This scenario does not provide a method directly applicable to real-world problems; it only aims to intuitively demonstrate the difference in fairness between FL and CML. The average EOD value obtained in this experiment was 0.1332, which, as expected, maintained its position between Local and FL approaches on the test data they had never seen before.

In the FL approach, although the clients cannot access each other’s data, the variation between the EOD scores they obtain is quite low. The standard deviation observed in FL (0.0067) is significantly lower compared to the value of 0.0375 in the Local experiment, and this consistency shows the success of the FL approach. On the other hand, significant differences were recorded between the clients in the Local experiment. The average EOD score of the FL experiment showed an improvement of 80.5% and 78.8% compared to the Local and Central approaches, respectively. The value of F_{global} in the table represents the EOD score of the global model created on the server. The closeness of F_{global} to the FL average confirms FairFed’s expectation that each client will converge to the global score according to Equation 4.5 and reinforces the validity of the experiment.

When the experimental results were examined at the client level, a high consistency was again observed. According to the previously defined gender ratio distribution, Client-0, Client-1 and Client-4, which experienced the highest unfairness, recorded the highest EOD scores, respectively. For example, while Client-4 achieved the highest EOD value in both Local and Central conditions,

the highest score was produced by Client-1 in the FL scenario. In contrast, Client-2 and Client-3, which had a more balanced distribution, showed the most equitable results with the lowest EOD scores. Specifically, Client-2 achieved the lowest score in the Local approach and the second lowest value in the Central condition; while Client-3 achieved the lowest EOD score in the FL application.

Table 6.3 : Scores for FedAvg, FedF1 and fairness experiments with LR

Metric	FL (FedAvg)	FL (FedF1)	Local	Central	FL (FairFed)
Precision	0.71	0.71	0.74	0.74	0.73
Recall	0.16	0.26	0.49	0.48	0.48
F1	0.26	0.37	0.59	0.58	0.58
Accuracy	0.80	0.81	0.84	0.84	0.84

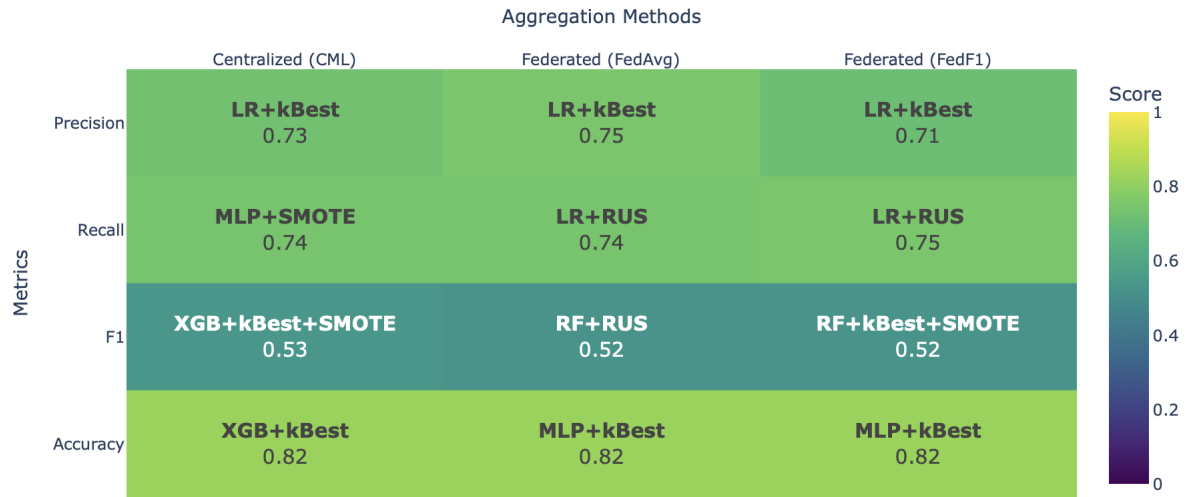
Table 6.3 summarizes the performance of the models created in the fairness experiments based on ML metrics. The “Metric” column shows the basic metrics such as Precision, Recall, F1 and Accuracy; the "FL (FedAvg)" column shows the FL results obtained with FedAvg in the ML Benchmark experiment and "FL (FedF1)" shows results obtained with FedF1; and the other columns show the performance of the Local, Central and FairFed approaches. The aim of this experiment is to quantitatively examine the trade-off between fairness and success.

Looking at the table, while the average Recall value of the FedAvg-based benchmark model on Class-1 is only 16%, in the fairness-focused Local, Central and FairFed approaches, Recall increased to 48–49% and F1 score increased from 0.26 to approximately 0.58–0.59. On the other hand, Precision (0.74) remained the same, while Accuracy (0.84) increased by 4%. In addition, in the first experiment in our study, FedF1 gave better results than FedAvg for this ML Model. Again, we see that FedF1 has better results in all metrics in this table. However, it can be followed behind other approaches. Here, the recall value increased from 0.26 to 0.48-0.49. In addition, the F1 value increased from 0.37 to 0.58-0.59.

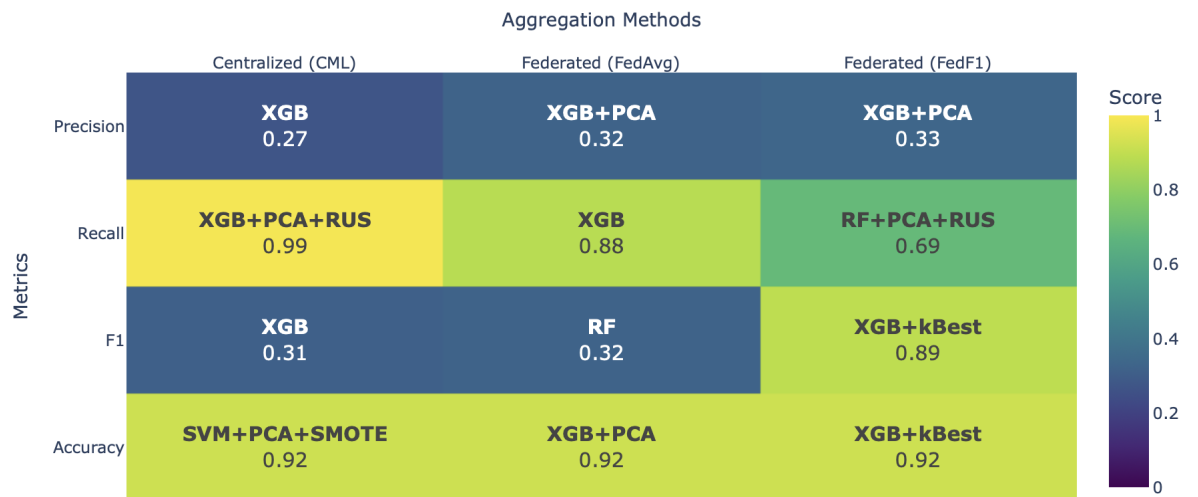
In FairFed, the contribution of each client is weighted according to the EOD score it achieves; clients with low EOD, i.e., clients with almost equal distribution of

positive and negative classes, receive higher weight. Thus, the global model learns more signals from clients with relatively good class balance, which directly increases the recognition of the minority (disadvantaged) class (i.e., the recall value). Thanks to FairFed’s weighting strategy, the model finds more true positives (TP) and, in parallel, the number of small false positives (FP) increases. Therefore, when the TP and FP increase rates remain close to each other, the $TP/(TP+FP)$ ratio (i.e., precision) remains similar. For Recall, the model starts to catch the positives it previously missed. This increases the number of TPs, and as the $TP/(TP+FN)$ ratio increases, a significant increase in recall is observed. In terms of F1, since this metric is already based on Recall and Precision, an increase in either one directly increases F1.

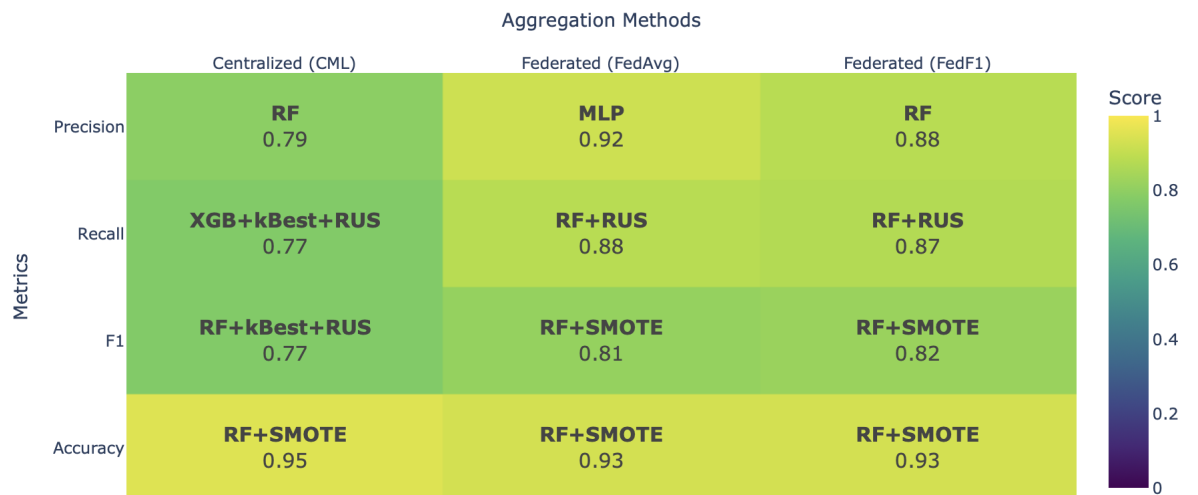




(a) Best Performance Results for DCCC



(b) Best Performance Results for HCDR



(c) Best Performance Results for HMEQ

Figure 6.1 : Heatmap of Best Performance Results and Models According to Dataset



7. CONCLUSION

Federated learning is an effective approach that allows model training on distributed data while preserving data privacy. This method offers significant advantages in dealing with data ownership and regulatory restrictions. For this reason, its use has become increasingly widespread in various areas such as healthcare, finance and IoT.

This thesis explores ways to boost both model performance and fairness in federated learning (FL) for financial data, without ever sharing sensitive records. It focuses on how different FL aggregation methods impact standard evaluation metrics (like accuracy and F1) and fairness measures (such as Equal Opportunity Difference). To make sure the findings are reliable, experiments are run on three well-known financial datasets.

In the thesis, comprehensive performance comparisons were made using common methods such as LR, RF, XGB, MLP and SVM. In the data preprocessing step, sampling techniques such as SMOTE, RUS and dimensionality reduction methods such as PCA and kBest were applied. In the first experiment in the thesis, model performance was analyzed as CML and FL on DCCC, HCDR and HMEQ data sets, and in the second experiment, fairness optimization strategies and their effects on model performances under non-IID distributions on the DCCC data set were compared.

According to the outputs of the first experiment, precision, recall, f1 and accuracy metrics results were exhibited with FL and CML methods. In addition, each experiment was repeated 10 times and the consistency of the results was ensured. Accordingly, the FedF1 method we proposed gave better or equally good results than FedAvg in most scenarios. In general, FL methods provided very low performance loss while preserving data privacy compared to CML.

In the second experiment results, the FairFed method was compared with the Local, Central, and FL methods under unbalanced data distribution conditions and it was seen that it provided consistent improvements on the EOD basis. In addition, the model performance either increased or was not affected at all based on ML success metrics. Thus, we showed that FairFed increased the model performance in addition to ensuring fairness. In this section, we ensured the robustness of the results by repeating the experiments we conducted 100 times.

The developed codebase has been published on GitHub and the full reproducibility of all experiments has been ensured.

For future work, there are several possible directions to extend our current research. First, the scalability of the FL approach should be evaluated on larger and more heterogeneous datasets representing different types of financial institutions. Additionally, customer-based performance should be analyzed to examine how well it performs with limited datasets. Another direction is to integrate other data sources such as real-time financial transaction data to improve the model performance for detecting anomalies and predicting financial defaults. Furthermore, to increase the robustness of this study, future experiments should be conducted using various publicly available datasets.

In conclusion, this thesis study has made concrete contributions to the field by demonstrating the applicability of the FL approach in the field of finance and the impact of fairness oriented developments and has established a solid foundation for further research.

REFERENCES

- [1] **HomeCreditGroup** (2018). *Home Credit Default Risk*, <https://www.kaggle.com/c/home-credit-default-risk/overview>.
- [2] **McMahan, H.B., Moore, E., Ramage, D. and y Arcas, B.A.** (2016). Federated Learning of Deep Networks using Model Averaging, *CoRR*, *abs/1602.05629*, <http://arxiv.org/abs/1602.05629>, 1602.05629.
- [3] **Chatterjee, P., Das, D. and Rawat, D.B.** (2024). Securing Financial Services with Federated Learning and Blockchain, chapter Secure and Smart Cyber-Physical Systems, CRC Press eBooks, p.178–207, <https://doi.org/10.1201/9781003376712-9>.
- [4] **McMahan, H.B., Moore, E., Ramage, D., Hampson, S. and Arcas, B.A.Y.** (2016). Communication-Efficient Learning of Deep Networks from Decentralized Data, *arXiv (Cornell University)*, <https://arxiv.org/abs/1602.05629>.
- [5] **Varmedja, D., Karanovic, M., Sladojevic, S., Arsenovic, M. and Anderla, A.** (2019). Credit Card Fraud Detection - Machine Learning methods, *2019 INFOTEH*, 1–5, <https://doi.org/10.1109/infoteh.2019.8717766>.
- [6] **Aurna, N.F., Hossain, M.D., Taenaka, Y. and Kadobayashi, Y.** (2023). Federated Learning-Based Credit Card Fraud Detection: Performance Analysis with Sampling Methods and Deep Learning Algorithms, *2023 IEEE CSR*, 180–186, <https://doi.org/10.1109/csr57506.2023.10224978>.
- [7] **Awosika, T., Shukla, R.M. and Pranggono, B.** (2024). Transparency and Privacy: The role of explainable AI and federated learning in financial fraud Detection, *IEEE Access*, *12*, 64551–64560, <https://doi.org/10.1109/access.2024.3394528>.
- [8] **Chatterjee, P., Das, D. and Rawat, D.B.** (2023). Federated Learning Empowered Recommendation Model for Financial Consumer services, *IEEE Transactions on Consumer Electronics*, *70*(1), 2508–2516, <https://doi.org/10.1109/tce.2023.3339702>.
- [9] **Yeh, I.C. and Lien, C.H.** (2008). The comparisons of data mining techniques for the predictive accuracy of probability of default of credit

card clients, *Expert Systems With Applications*, 36(2), 2473–2480, <https://doi.org/10.1016/j.eswa.2007.12.020>.

- [10] **VALLALA, A.** (2018). *HMEQ Data*, <https://www.kaggle.com/datasets/ajay1735/hmeq-data/data>.
- [11] **Saxena, A., Agarwal, A., Pandey, B.K. and Pandey, D.** (2024). Examination of the criticality of customer segmentation using unsupervised learning methods, *Circular Economy and Sustainability*, 4(2), 1447–1460, <https://doi.org/10.1007/s43615-023-00336-4>.
- [12] **Jiangshan, W., Tingting, D., Shuochen, B., Wenqing, B. and Jue, X.** (2024). Research on the application of data analysis in predicting financial risk, *Financial Engineering and Risk Management*, 7(5), <https://doi.org/10.23977/ferm.2024.070424>.
- [13] **Machine Learning Group ULB, A.** (2018). *Credit Card Fraud Detection*, <https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud>.
- [14] **Bakumenko, A. and Elragal, A.** (2022). Detecting anomalies in financial data using machine learning algorithms, *Systems*, 10(5), 130, <https://doi.org/10.3390/systems10050130>.
- [15] **Ramakrishnan, J., Shaabani, E., Li, C. and Sustik, M.A.** (2019). Anomaly Detection for an E-commerce Pricing System, *The 25th ACM SIGKDD*, <https://doi.org/10.1145/3292500.3330748>.
- [16] **Guo, L., Song, R., Wu, J., Xu, Z. and Zhao, F.** (2024). Integrating a Machine Learning-Driven Fraud Detection System Based on a Risk Management Framework, *6th ICCDS*, <https://doi.org/10.20944/preprints202406.1756.v1>.
- [17] **Qiu, Y. and Wang, J.** (2024). A Machine Learning Approach to Credit Card Customer Segmentation for Economic Stability, *Proceedings of the 4th International Conference on Economic Management and Big Data Applications, ICEMBDA 2023*, <https://doi.org/10.4108/eai.27-10-2023.2342007>.
- [18] **Wahab, F., Khan, I. and Sabada, S.** (2024). Credit card default prediction using ML and DL techniques, *Internet of Things and Cyber-Physical Systems*, 4, 293–306, <https://doi.org/10.1016/j.iotcps.2024.09.001>.
- [19] **Zhou, Y., Chi, G., Liu, J., Xiong, J. and Wang, B.** (2022). Default discrimination of credit card: Feature combination selection based on improved FDAF-score, *Expert Systems With Applications*, 206, 117829, <https://doi.org/10.1016/j.eswa.2022.117829>.

- [20] **Antunes, R.S., Da Costa, C.A., Küderle, A., Yari, I.A. and Eskofier, B.** (2022). Federated Learning for Healthcare: Systematic Review and architecture Proposal, *ACM Transactions on Intelligent Systems and Technology*, 13(4), 1–23, <https://doi.org/10.1145/3501813>.
- [21] **Ülver, B., Yurtoğlu, R.A., Dervişoğlu, H., Halepmollası, R. and Haklıdır, M.** (2023). Federated Learning in Predicting Heart Disease, *2023 31st SIU*, 1–4, <https://doi.org/10.1109/siu59756.2023.10223935>.
- [22] **Nguyen, D.C., Ding, M., Pathirana, P.N., Seneviratne, A., Li, J. and Poor, H.V.** (2021). Federated Learning for Internet of Things: A Comprehensive survey, *IEEE Communications Surveys & Tutorials*, 23(3), 1622–1658, <https://doi.org/10.1109/comst.2021.3075439>.
- [23] **Yang, Z., Chen, M., Saad, W., Hong, C.S. and Shikh-Bahaei, M.** (2020). Energy efficient federated learning over wireless communication networks, *IEEE Transactions on Wireless Communications*, 20(3), 1935–1949, <https://doi.org/10.1109/twc.2020.3037554>.
- [24] **Saputra, Y.M., Hoang, D.T., Nguyen, D.N., Dutkiewicz, E., Mueck, M.D. and Srikanteswara, S.** (2019). Energy Demand Prediction with Federated Learning for Electric Vehicle Networks, *2015 IEEE Global Communications Conference (GLOBECOM)*, <https://doi.org/10.1109/globecom38437.2019.9013587>.
- [25] **Goutier, M., Diebel, C., Adam, M. and Benlian, A.** (2024). Federated Learning for credit risk assessment, *Proceedings of the ... Annual Hawaii International Conference on System Sciences/Proceedings of the Annual Hawaii International Conference on System Sciences*, <https://doi.org/10.24251/hicss.2023.048>.
- [26] **Imteaj, A. and Amini, M.H.** (2022). Leveraging asynchronous federated learning to predict customers financial distress, *Intelligent Systems With Applications*, 14, 200064, <https://doi.org/10.1016/j.iswa.2022.200064>.
- [27] **Yan and Guihua** (2023). Research and Practice of Financial Credit Risk Management Based on Federated Learning., *Engineering Letters*, 2023, Vol 31, Issue 1, P271, 31, 271, https://www.engineeringletters.com/issues_v31/issue_1/EL_31_1_28.pdf.
- [28] **Singh, A.K., Blanco-Justicia, A., Domingo-Ferrer, J., Sánchez, D. and Rebollo-Monedero, D.** (2020). Fair Detection of Poisoning Attacks in Federated Learning, *2020 IEEE 32nd International Conference on Tools with Artificial Intelligence (ICTAI)*, pp.224–229.

- [29] **Becker, B. and Kohavi, R.** (1996). *Adult*, UCI Machine Learning Repository, DOI: <https://doi.org/10.24432/C5XW20>.
- [30] **Thu, H.G.T., Doan, T.V. and Quy, T.L.** (2024). *An experimental study on fairness-aware machine learning for credit scoring problem*, <https://arxiv.org/abs/2412.20298>, 2412.20298.
- [31] **Ezzeldin, Y.H., Yan, S., He, C., Ferrara, E. and Avestimehr, S.** (2022). *FairFed: Enabling Group Fairness in Federated Learning*, <https://arxiv.org/abs/2110.00857>, 2110.00857.
- [32] **Abay, A., Zhou, Y., Baracaldo, N., Rajamoni, S., Chuba, E. and Ludwig, H.** (2020). *Mitigating Bias in Federated Learning*, <https://arxiv.org/abs/2012.02447>, 2012.02447.
- [33] **Roh, Y., Lee, K., Whang, S.E. and Suh, C.** (2021). *FairBatch: Batch Selection for Model Fairness*, <https://arxiv.org/abs/2012.01696>, 2012.01696.
- [34] **Wasif, D., Chen, D., Madabushi, S., Alluru, N., Moore, T.J. and Cho, J.H.** (2025). *Empirical Analysis of Privacy-Fairness-Accuracy Trade-offs in Federated Learning: A Step Towards Responsible AI*, <https://arxiv.org/abs/2503.16233>, 2503.16233.
- [35] **Yeh, I.C.** (2016). *UCI ML Repository Default of Credit Card Clients*, <https://archive.ics.uci.edu/dataset/350/default+of+credit+card+clients>.
- [36] **HomeCredit** (2024). *Home Credit Welcome Page*, <https://www.homecredit.net/about-us.aspx/>.
- [37] **contributors, W.** (2024). *Home credit*, https://en.wikipedia.org/wiki/Home_Credit.
- [38] **Flower** (2025). *Flower: A friendly federated AI framework*, <https://flower.ai/>.
- [39] **FlowerDatasets** (2025). *DirichletPartitioner*, https://flower.ai/docs/datasets/ref-api/flwr_datasets.partitioner.DirichletPartitioner.html.
- [40] **Yurochkin, M., Agarwal, M., Ghosh, S., Greenewald, K., Hoang, T.N. and Khazaeni, Y.** (2019). *Bayesian Nonparametric Federated Learning of Neural Networks*, <https://arxiv.org/abs/1905.12022>, 1905.12022.
- [41] **Fairlearn** (2025). *Ernst & Young (EY) case study*, https://fairlearn.org/main/auto_examples/plot_credit_loan_decisions.html.

- [42] **He, D., Du, R., Zhu, S., Zhang, M., Liang, K. and Chan, S.** (2022). Secure Logistic Regression for Vertical Federated Learning, *IEEE Internet Computing*, 26(2), 61–68.
- [43] **KayChansiri** (2025). *GitHub - KayChansiri/Demo_logistic_regression_ML*, https://github.com/KayChansiri/Demo_logistic_regression_ML.
- [44] **Pathak, A. and Pandey, A.** (2025). Federated Learning for Kidney Disease Prediction using Combined LSTM and MLP Models, *2025 IEEE 14th International Conference on Communication Systems and Network Technologies (CSNT)*, pp.196–201.
- [45] **Wang, M., Bodonhelyi, A., Bozkir, E. and Kasneci, E.** (2024). TurboSVM-FL: Boosting Federated Learning through SVM Aggregation for Lazy Clients, *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(14), 15546–15554, <https://ojs.aaai.org/index.php/AAAI/article/view/29481>.
- [46] **Yang, M., Song, L., Xu, J., Li, C. and Tan, G.** (2019). *The Tradeoff Between Privacy and Accuracy in Anomaly Detection Using Federated XGBoost*, <https://arxiv.org/abs/1907.07157>, 1907.07157.
- [47] **Markovic, T., Leon, M., Buffoni, D. and Punnekkat, S.** (2022). Random Forest Based on Federated Learning for Intrusion Detection, *I. Maglogiannis, L. Iliadis, J. Macintyre and P. Cortez, editors, Artificial Intelligence Applications and Innovations*, Springer International Publishing, Cham, pp.132–144.
- [48] **Shingi, G.** (2020). A federated learning based approach for loan defaults prediction, *2020 International Conference on Data Mining Workshops (ICDMW)*, pp.362–368.
- [49] **Salam, M.A., Fouad, K.M., Elbably, D.L. and Elsayed, S.M.** (2024). Federated learning model for credit card fraud detection with data balancing techniques, *Neural Computing and Applications*, 36(11), 6231–6256, <https://doi.org/10.1007/s00521-023-09410-2>.
- [50] **Konečný, J., McMahan, B. and Ramage, D.** (2015). Federated Optimization: Distributed Optimization Beyond the Datacenter, *CoRR*, *abs/1511.03575*, <http://arxiv.org/abs/1511.03575>, 1511.03575.
- [51] **Caton, S. and Haas, C.** (2023). Fairness in Machine Learning: a survey, *ACM Computing Surveys*, 56(7), 1–38, <https://doi.org/10.1145/3616865>.

- [52] **Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K. and Galstyan, A.** (2021). A Survey on Bias and Fairness in Machine Learning, *ACM Comput. Surv.*, 54(6), <https://doi.org/10.1145/3457607>.
- [53] **Pessach, D. and Shmueli, E.** (2022). A Review on Fairness in Machine Learning, *ACM Comput. Surv.*, 55(3), <https://doi.org/10.1145/3494672>.
- [54] **Dwork, C., Hardt, M., Pitassi, T., Reingold, O. and Zemel, R.** (2011). *Fairness Through Awareness*, <https://arxiv.org/abs/1104.3913>, 1104.3913.
- [55] **Salazar, T., Santos, M.S., Araújo, H. and Abreu, P.H.** (2021). FAWOS: Fairness-Aware Oversampling Algorithm Based on Distributions of Sensitive Attributes, *IEEE Access*, 9, 81370–81379.
- [56] **Kamiran, F. and Calders, T.** (2012). Data preprocessing techniques for classification without discrimination, *Knowl. Inf. Syst.*, 33(1), 1–33, <https://doi.org/10.1007/s10115-011-0463-8>.
- [57] **Feldman, M., Friedler, S., Moeller, J., Scheidegger, C. and Venkatasubramanian, S.** (2015). *Certifying and removing disparate impact*, <https://arxiv.org/abs/1412.3756>, 1412.3756.
- [58] **Pouget, A., Jovanović, N., Vero, M., Staab, R. and Vechev, M.** (2024). *Back to the Drawing Board for Fair Representation Learning*, <https://arxiv.org/abs/2405.18161>, 2405.18161.
- [59] **Kamishima, T., Akaho, S., Asoh, H. and Sakuma, J.**, (2012). Fairness-Aware Classifier with Prejudice Remover Regularizer, **P.A. Flach, T. De Bie and N. Cristianini**, editors, Machine Learning and Knowledge Discovery in Databases, volume 7524 of *Lecture Notes in Computer Science*, chapter 3, Springer, Berlin, Heidelberg, pp.35–50, https://doi.org/10.1007/978-3-642-33486-3_3.
- [60] **Awasthi, P., Kleindessner, M. and Morgenstern, J.** (2020). *Equalized odds postprocessing under imperfect group information*, <https://arxiv.org/abs/1906.03284>, 1906.03284.
- [61] **scikit learn** (2025). *scikit-learn Machine Learning in Python*, <https://scikit-learn.org/>.
- [62] **xgboost** (2025). *xgboost Installation Guide*, https://xgboost.readthedocs.io/en/release_3.0.0/install.html.
- [63] **ImbalancedLearn** (2024). *imbalanced-learn documentation*, <https://imbalanced-learn.org/stable/>.
- [64] **HuggingFace** (2025). *Hugging Face*, <https://huggingface.co/>.

- [65] **PyTorch** (2025). *PyTorch Get Started Webpage*, <https://pytorch.org/>.
- [66] **Fairlearn** (2025). *Credit Loan Decisions*, <https://fairlearn.org/>.
- [67] **Zhu, H., Xu, J., Liu, S. and Jin, Y.** (2021). Federated learning on non-IID data: A survey, *Neurocomputing*, 465, 371–390, <https://doi.org/10.1016/j.neucom.2021.07.098>.
- [68] **Dervişoğlu** (2024). *Mitigating bias in machine learning models: Federated Learning Approaches*, Yıldız Teknik Üniversitesi, https://tez.yok.gov.tr/UlusalTezMerkezi/TezGoster?key=KMB79M3N7zK1UR2WYeRgQhgkEp-W8K_yswdPjMqnYieF7rZQrVh4tYFinNLlnZ_e.
- [69] <https://flower.ai/docs/framework/tutorial-quickstart-xgboost.html>.





APPENDICES

APPENDIX A : Model Performance on DCCC Dataset

APPENDIX B : Model Performance on HCDR Dataset

APPENDIX C : Model Performance on HMEQ Dataset

APPENDIX D : Application Architecture and Coding Details





Appendix A: Model Performance on DCCC Dataset

In this section, a detailed raw result table is presented that supports the analysis and discussions made on the DCCC dataset. Accordingly, Table-A.1 contains the performance metrics of all experimental configurations performed on the relevant dataset. All analyses made in Section-6 are based on the results in this table.



Table A.1 : Comparative experimental performance on the DCCC dataset

Model Name	Central ML				FL with FedAvg				FL with FedF1			
	P.	R.	F.	Ac.	P.	R.	F.	Ac.	P.	R.	F.	Ac.
LR	0.71	0.23	0.35	0.81	0.71	0.16	0.26	0.80	0.69	0.23	0.34	0.81
XGB	0.66	0.36	0.46	0.82	0.68	0.35	0.46	0.82	0.66	0.30	0.40	0.81
MLP	0.44	0.19	0.23	0.76	0.63	0.37	0.46	0.81	0.32	0.39	0.34	0.68
SVM	0.38	0.62	0.46	0.68	0.37	0.64	0.46	0.68	0.37	0.63	0.46	0.68
RF	0.67	0.37	0.47	0.82	0.65	0.32	0.42	0.81	0.67	0.32	0.42	0.81
LR+PCA	0.67	0.20	0.31	0.80	0.70	0.13	0.22	0.79	0.67	0.18	0.29	0.80
XGB+PCA	0.61	0.33	0.43	0.81	0.64	0.28	0.39	0.81	0.63	0.28	0.38	0.81
MLP+PCA	0.63	0.38	0.47	0.81	0.58	0.36	0.43	0.80	0.59	0.36	0.44	0.80
SVM+PCA	0.39	0.61	0.47	0.70	0.37	0.61	0.45	0.69	0.37	0.61	0.46	0.69
RF+PCA	0.63	0.34	0.44	0.81	0.66	0.31	0.42	0.82	0.63	0.30	0.40	0.81
LR+kBest	0.73	0.21	0.32	0.81	0.75	0.11	0.18	0.79	0.71	0.25	0.37	0.81
XGB+kBest	0.69	0.36	0.48	0.82	0.68	0.29	0.39	0.81	0.69	0.30	0.40	0.81
MLP+kBest	0.68	0.34	0.46	0.82	0.64	0.40	0.49	0.82	0.63	0.41	0.49	0.82
SVM+kBest	0.49	0.54	0.51	0.77	0.49	0.53	0.50	0.77	0.50	0.53	0.51	0.78
RF+kBest	0.67	0.36	0.46	0.82	0.64	0.37	0.46	0.82	0.65	0.36	0.46	0.82
LR+SMOTE	0.37	0.59	0.45	0.69	0.39	0.60	0.47	0.71	0.39	0.60	0.47	0.70
XGB+SMOTE	0.53	0.53	0.53	0.79	0.46	0.56	0.50	0.76	0.42	0.65	0.50	0.71
MLP+SMOTE	0.29	0.74	0.42	0.55	0.37	0.68	0.47	0.68	0.29	0.71	0.40	0.55
SVM+SMOTE	0.29	0.70	0.41	0.57	0.33	0.68	0.44	0.62	0.32	0.68	0.43	0.62
RF+SMOTE	0.49	0.56	0.52	0.77	0.53	0.51	0.52	0.79	0.52	0.53	0.52	0.79
LR+PCA+SMOTE	0.37	0.64	0.47	0.68	0.30	0.56	0.39	0.62	0.32	0.56	0.40	0.63
XGB+PCA+SMOTE	0.46	0.59	0.52	0.76	0.34	0.51	0.40	0.66	0.35	0.52	0.42	0.68
MLP+PCA+SMOTE	0.42	0.62	0.50	0.72	0.28	0.41	0.33	0.64	0.27	0.45	0.33	0.61
SVM+PCA+SMOTE	0.38	0.62	0.47	0.69	0.31	0.54	0.39	0.64	0.33	0.55	0.41	0.64
RF+PCA+SMOTE	0.50	0.55	0.53	0.78	0.48	0.53	0.50	0.77	0.49	0.54	0.51	0.78
LR+kBest+SMOTE	0.48	0.54	0.51	0.77	0.47	0.56	0.50	0.76	0.46	0.58	0.50	0.74
XGB+kBest+SMOTE	0.54	0.53	0.53	0.80	0.41	0.62	0.49	0.71	0.39	0.66	0.47	0.67
MLP+kBest+SMOTE	0.49	0.57	0.52	0.77	0.45	0.59	0.50	0.75	0.45	0.59	0.51	0.75
SVM+kBest+SMOTE	0.49	0.55	0.52	0.77	0.50	0.52	0.50	0.78	0.49	0.55	0.51	0.77
RF+kBest+SMOTE	0.54	0.52	0.53	0.80	0.55	0.49	0.51	0.80	0.54	0.50	0.52	0.80
LR+RUS	0.37	0.67	0.47	0.67	0.28	0.74	0.40	0.53	0.29	0.75	0.41	0.53
XGB+RUS	0.46	0.64	0.53	0.75	0.49	0.57	0.52	0.76	0.48	0.58	0.52	0.76
MLP+RUS	0.40	0.65	0.50	0.71	0.43	0.63	0.50	0.73	0.42	0.61	0.49	0.73
SVM+RUS	0.39	0.64	0.48	0.69	0.38	0.65	0.47	0.69	0.38	0.65	0.47	0.69
RF+RUS	0.47	0.62	0.53	0.76	0.45	0.64	0.52	0.75	0.45	0.63	0.52	0.75
LR+PCA+RUS	0.35	0.67	0.46	0.66	0.30	0.56	0.38	0.62	0.29	0.54	0.38	0.61
XGB+PCA+RUS	0.45	0.62	0.52	0.75	0.33	0.53	0.41	0.65	0.37	0.55	0.44	0.68
MLP+PCA+RUS	0.43	0.64	0.51	0.73	0.29	0.54	0.37	0.60	0.28	0.55	0.36	0.58
SVM+PCA+RUS	0.39	0.60	0.47	0.70	0.30	0.55	0.39	0.62	0.31	0.55	0.39	0.64
RF+PCA+RUS	0.47	0.59	0.53	0.77	0.44	0.60	0.50	0.75	0.46	0.60	0.52	0.76
LR+kBest+RUS	0.48	0.54	0.51	0.77	0.30	0.74	0.39	0.50	0.29	0.75	0.39	0.48
XGB+kBest+RUS	0.50	0.57	0.53	0.76	0.47	0.58	0.51	0.76	0.48	0.59	0.52	0.76
MLP+kBest+RUS	0.48	0.59	0.53	0.77	0.46	0.61	0.52	0.76	0.47	0.61	0.52	0.76
SVM+kBest+RUS	0.49	0.53	0.51	0.77	0.47	0.55	0.50	0.77	0.50	0.53	0.50	0.78
RF+kBest+RUS	0.50	0.57	0.53	0.78	0.46	0.58	0.51	0.78	0.47	0.57	0.51	0.77

¹ Top scores per metric (out of 45 models) are shown in **bold**; if there are multiple top scores, they are broken by Ac., F., P., R.

Appendix B: Model Performance on HCDR Dataset

In this section, a detailed raw result table is presented that supports the analysis and discussions made on the HCDR dataset. Accordingly, Table-B.1 contains the performance metrics of all experimental configurations performed on the relevant dataset. All analyses made in Section-6 are based on the results in this table.



Table B.1 : Comparative experimental performance on the HCDR dataset

Model Name	Central ML				FL with FedAvg				FL with FedF1			
	P.	R.	F.	Ac.	P.	R.	F.	Ac.	P.	R.	F.	Ac.
LR	0.10	0.58	0.17	0.57	0.12	0.66	0.18	0.48	0.11	0.68	0.18	0.46
XGB	0.27	0.37	0.31	0.87	0.08	0.88	0.00	0.92	0.00	0.50	0.00	0.92
MLP	0.12	0.50	0.21	0.66	0.10	0.56	0.13	0.61	0.09	0.52	0.17	0.57
SVM	0.17	0.69	0.27	0.69	0.21	0.64	0.20	0.56	0.20	0.67	0.29	0.52
RF	0.19	0.56	0.28	0.76	0.26	0.42	0.32	0.85	0.23	0.50	0.31	0.82
LR+PCA	0.16	0.69	0.26	0.68	0.11	0.69	0.19	0.52	0.11	0.68	0.18	0.52
XGB+PCA	0.25	0.23	0.24	0.88	0.32	0.05	0.07	0.92	0.33	0.04	0.07	0.92
MLP+PCA	0.11	0.52	0.19	0.66	0.16	0.38	0.15	0.67	0.12	0.39	0.14	0.69
SVM+PCA	0.15	0.68	0.25	0.66	0.15	0.58	0.24	0.77	0.15	0.58	0.23	0.70
RF+PCA	0.18	0.48	0.26	0.78	0.29	0.31	0.29	0.88	0.26	0.40	0.30	0.85
LR+kBest	0.15	0.64	0.24	0.68	0.14	0.64	0.23	0.66	0.14	0.65	0.23	0.66
XGB+kBest	0.26	0.35	0.30	0.87	0.02	0.88	0.00	0.92	0.00	0.45	0.89	0.92
MLP+kBest	0.09	0.55	0.17	0.63	0.08	0.57	0.17	0.62	0.07	0.46	0.18	0.62
SVM+kBest	0.15	0.66	0.24	0.67	0.15	0.65	0.24	0.67	0.15	0.65	0.24	0.68
RF+kBest	0.17	0.59	0.26	0.73	0.18	0.45	0.26	0.80	0.17	0.52	0.26	0.77
LR+SMOTE	0.11	0.55	0.18	0.59	0.11	0.54	0.19	0.63	0.11	0.54	0.19	0.63
XGB+SMOTE	0.23	0.41	0.30	0.84	0.00	0.57	0.00	0.92	0.22	0.12	0.13	0.89
MLP+SMOTE	0.08	0.18	0.08	0.70	0.04	0.25	0.00	0.70	0.15	0.06	0.07	0.70
SVM+SMOTE	0.00	0.00	0.00	0.92	0.09	0.00	0.00	0.92	0.30	0.00	0.00	0.92
RF+SMOTE	0.21	0.09	0.12	0.90	0.22	0.09	0.13	0.90	0.21	0.12	0.15	0.89
LR+PCA+SMOTE	0.10	0.64	0.18	0.53	0.11	0.60	0.19	0.58	0.11	0.60	0.19	0.59
XGB+PCA+SMOTE	0.15	0.36	0.21	0.78	0.12	0.24	0.15	0.78	0.12	0.24	0.15	0.78
MLP+PCA+SMOTE	0.07	0.13	0.05	0.70	0.09	0.22	0.09	0.64	0.04	0.24	0.09	0.64
SVM+PCA+SMOTE	0.22	0.01	0.02	0.92	0.19	0.35	0.15	0.69	0.11	0.38	0.16	0.66
RF+PCA+SMOTE	0.11	0.02	0.04	0.91	0.25	0.09	0.13	0.91	0.21	0.10	0.13	0.91
LR+kBest+SMOTE	0.12	0.46	0.19	0.69	0.15	0.35	0.20	0.78	0.15	0.36	0.21	0.78
XGB+kBest+SMOTE	0.17	0.55	0.26	0.74	0.18	0.25	0.20	0.84	0.18	0.26	0.20	0.84
MLP+kBest+SMOTE	0.10	0.30	0.14	0.70	0.10	0.18	0.13	0.70	0.12	0.21	0.14	0.70
SVM+kBest+SMOTE	0.18	0.28	0.22	0.84	0.18	0.28	0.22	0.84	0.18	0.29	0.22	0.84
RF+kBest+SMOTE	0.20	0.19	0.19	0.87	0.19	0.20	0.19	0.87	0.18	0.22	0.20	0.86
LR+RUS	0.11	0.49	0.18	0.64	0.12	0.52	0.19	0.64	0.12	0.52	0.19	0.64
XGB+RUS	0.09	0.98	0.17	0.22	0.15	0.68	0.24	0.66	0.15	0.68	0.24	0.66
MLP+RUS	0.06	0.55	0.13	0.57	0.08	0.58	0.17	0.59	0.09	0.60	0.17	0.59
SVM+RUS	0.16	0.68	0.26	0.68	0.16	0.68	0.26	0.69	0.16	0.68	0.26	0.69
RF+RUS	0.15	0.66	0.25	0.68	0.15	0.66	0.24	0.67	0.15	0.66	0.24	0.67
LR+PCA+RUS	0.10	0.64	0.18	0.53	0.11	0.60	0.19	0.58	0.11	0.59	0.18	0.58
XGB+PCA+RUS	0.08	0.99	0.15	0.13	0.12	0.50	0.18	0.64	0.12	0.50	0.19	0.64
MLP+PCA+RUS	0.06	0.59	0.14	0.50	0.06	0.55	0.13	0.54	0.06	0.55	0.13	0.53
SVM+PCA+RUS	0.15	0.67	0.24	0.66	0.13	0.63	0.21	0.62	0.13	0.63	0.21	0.62
RF+PCA+RUS	0.14	0.64	0.23	0.66	0.17	0.68	0.25	0.67	0.18	0.69	0.26	0.68
LR+kBest+RUS	0.12	0.59	0.20	0.61	0.14	0.64	0.23	0.65	0.14	0.64	0.22	0.65
XGB+kBest+RUS	0.09	0.98	0.16	0.18	0.15	0.68	0.24	0.66	0.15	0.67	0.24	0.66
MLP+kBest+RUS	0.07	0.58	0.16	0.54	0.08	0.56	0.17	0.59	0.08	0.60	0.15	0.56
SVM+kBest+RUS	0.15	0.65	0.24	0.68	0.15	0.65	0.24	0.67	0.15	0.65	0.24	0.67
RF+kBest+RUS	0.15	0.68	0.25	0.67	0.15	0.66	0.24	0.67	0.15	0.67	0.24	0.67

¹ Top scores per metric (out of 45 models) are shown in **bold**; if there are multiple top scores, they are broken by Ac., F., P., R.

Appendix C: Model Performance on HMEQ Dataset

In this section, a detailed raw result table is presented that supports the analysis and discussions made on the HMEQ dataset. Accordingly, Table-C.1 contains the performance metrics of all experimental configurations performed on the relevant dataset. All analyses made in Section-6 are based on the results in this table.



Table C.1 : Comparative experimental performance on the HMEQ dataset

Model Name	Central ML				FL with FedAvg				FL with FedF1			
	P.	R.	F.	Ac.	P.	R.	F.	Ac.	P.	R.	F.	Ac.
LR	0.48	0.49	0.35	0.77	0.53	0.52	0.50	0.80	0.55	0.53	0.50	0.81
XGB	0.52	0.42	0.46	0.71	0.64	0.56	0.44	0.73	0.62	0.53	0.42	0.72
MLP	0.37	0.44	0.32	0.81	0.92	0.12	0.21	0.82	0.67	0.57	0.51	0.80
SVM	0.26	0.00	0.24	0.63	0.68	0.52	0.43	0.74	0.63	0.51	0.41	0.73
RF	0.79	0.64	0.75	0.91	0.89	0.70	0.78	0.92	0.88	0.67	0.76	0.92
LR+PCA	0.51	0.54	0.34	0.79	0.47	0.51	0.26	0.80	0.23	0.53	0.31	0.54
XGB+PCA	0.50	0.33	0.39	0.71	0.53	0.22	0.28	0.75	0.34	0.24	0.25	0.74
MLP+PCA	0.46	0.53	0.43	0.82	0.47	0.45	0.39	0.88	0.33	0.54	0.36	0.67
SVM+PCA	0.13	0.00	0.34	0.60	0.22	0.40	0.33	0.13	0.22	0.66	0.32	0.47
RF+PCA	0.72	0.51	0.52	0.86	0.78	0.58	0.52	0.90	0.81	0.46	0.58	0.87
LR+kBest	0.51	0.49	0.35	0.82	0.48	0.59	0.52	0.79	0.47	0.57	0.51	0.79
XGB+kBest	0.56	0.41	0.44	0.73	0.74	0.61	0.66	0.88	0.76	0.59	0.66	0.88
MLP+kBest	0.49	0.53	0.42	0.79	0.70	0.27	0.39	0.83	0.66	0.62	0.63	0.89
SVM+kBest	0.35	0.50	0.30	0.74	0.50	0.56	0.52	0.80	0.49	0.55	0.52	0.80
RF+kBest	0.66	0.65	0.65	0.89	0.73	0.68	0.70	0.88	0.73	0.67	0.69	0.89
LR+SMOTE	0.40	0.65	0.37	0.69	0.55	0.54	0.50	0.80	0.54	0.56	0.52	0.80
XGB+SMOTE	0.44	0.52	0.47	0.77	0.54	0.59	0.43	0.71	0.51	0.58	0.41	0.72
MLP+SMOTE	0.30	0.56	0.35	0.76	0.50	0.65	0.51	0.76	0.50	0.63	0.51	0.76
SVM+SMOTE	0.16	0.20	0.13	0.75	0.66	0.54	0.42	0.72	0.66	0.53	0.41	0.71
RF+SMOTE	0.72	0.59	0.75	0.95	0.84	0.79	0.81	0.93	0.84	0.80	0.82	0.93
LR+PCA+SMOTE	0.40	0.74	0.38	0.72	0.23	0.62	0.33	0.50	0.23	0.58	0.32	0.51
XGB+PCA+SMOTE	0.37	0.53	0.42	0.73	0.30	0.56	0.37	0.62	0.30	0.55	0.37	0.63
MLP+PCA+SMOTE	0.34	0.46	0.37	0.83	0.23	0.57	0.33	0.53	0.23	0.57	0.32	0.52
SVM+PCA+SMOTE	0.33	0.24	0.20	0.07	0.23	0.64	0.33	0.50	0.22	0.66	0.33	0.47
RF+PCA+SMOTE	0.69	0.59	0.59	0.90	0.68	0.67	0.67	0.87	0.67	0.68	0.67	0.87
LR+kBest+SMOTE	0.47	0.62	0.40	0.74	0.44	0.61	0.51	0.78	0.47	0.59	0.52	0.78
XGB+kBest+SMOTE	0.41	0.60	0.45	0.73	0.59	0.73	0.64	0.85	0.62	0.73	0.66	0.85
MLP+kBest+SMOTE	0.40	0.52	0.43	0.81	0.43	0.65	0.51	0.76	0.42	0.65	0.50	0.75
SVM+kBest+SMOTE	0.26	0.40	0.34	0.09	0.46	0.60	0.52	0.78	0.46	0.60	0.51	0.79
RF+kBest+SMOTE	0.68	0.64	0.77	0.89	0.62	0.71	0.65	0.86	0.63	0.70	0.66	0.87
LR+RUS	0.43	0.64	0.40	0.79	0.48	0.51	0.48	0.78	0.49	0.52	0.48	0.79
XGB+RUS	0.33	0.76	0.41	0.55	0.58	0.57	0.44	0.73	0.53	0.57	0.43	0.74
MLP+RUS	0.34	0.71	0.43	0.70	0.38	0.59	0.44	0.70	0.37	0.58	0.43	0.70
SVM+RUS	0.26	0.00	0.23	0.62	0.65	0.51	0.45	0.78	0.64	0.52	0.44	0.77
RF+RUS	0.76	0.71	0.44	0.79	0.65	0.88	0.74	0.88	0.63	0.87	0.72	0.88
LR+PCA+RUS	0.41	0.73	0.41	0.66	0.24	0.62	0.34	0.51	0.23	0.61	0.33	0.51
XGB+PCA+RUS	0.32	0.76	0.41	0.49	0.25	0.70	0.36	0.51	0.27	0.65	0.39	0.55
MLP+PCA+RUS	0.35	0.71	0.41	0.71	0.22	0.58	0.32	0.51	0.23	0.63	0.33	0.50
SVM+PCA+RUS	0.25	0.00	0.22	0.38	0.23	0.65	0.33	0.48	0.23	0.62	0.33	0.49
RF+PCA+RUS	0.71	0.70	0.47	0.84	0.42	0.73	0.52	0.75	0.44	0.71	0.53	0.76
LR+kBest+RUS	0.47	0.66	0.42	0.77	0.47	0.56	0.50	0.79	0.47	0.56	0.51	0.79
XGB+kBest+RUS	0.35	0.77	0.40	0.53	0.55	0.77	0.63	0.83	0.53	0.78	0.62	0.83
MLP+kBest+RUS	0.38	0.70	0.43	0.76	0.44	0.58	0.49	0.77	0.46	0.56	0.50	0.78
SVM+kBest+RUS	0.14	0.60	0.31	0.57	0.52	0.55	0.53	0.81	0.52	0.56	0.54	0.81
RF+kBest+RUS	0.68	0.68	0.46	0.85	0.55	0.81	0.65	0.83	0.53	0.82	0.64	0.83

¹ Top scores per metric (out of 45 models) are shown in **bold**; if there are multiple top scores, they are broken by Ac., F., P., R.

Appendix D: Application Architecture and Coding Details

This section presents the application architecture and coding details behind the Comparative Federated Learning and Federated Learning Fairness experiments conducted within the scope of this thesis. The aim is to explain the technical infrastructure of the study and to ensure reproducibility. All project code is available via the Git repository at <https://github.com/Mstfakts/Federated-Learning-Comparative-Study>

D.1 Project Structure and Dependencies

Our codebase is designed to be modular and extensible. This structure allows for easy management of different experiments (benchmarks, fairness, etc.) and components (datasets, models, aggregation strategies). The main directory structure is as follows:

```
Federated-Learning-Comparative-Study
├── configs .....YAML based experiment settings
├── data ..... Data prep and analysis notebooks
├── experiments .....Experiment runners for ML and fairness
├── models ..... Model factory definition
├── results .....Scripts for visualization and outputs
├── src .....Core modules for centralized and federated ML
│   ├── centralized .....Central ML: training and fairness
│   │   ├── fairness_experiments .....Centralized fairness analysis
│   │   └── federated .....Federated learning pipeline
│   │       ├── aggregations .....Aggregation strategies: FedAvg, FedF1
│   │       ├── clients .....Client side logic and setup
│   │       └── server .....Server side orchestration
└── utils ..... Helper tools for experiments and reports
```

D.2 Development of Custom Aggregation Strategies

To evaluate the impact of fairness-focused and performance-focused federation learning, we implemented specialized aggregation strategies such as FedF1 and FairFed. These strategies go beyond standard FedAvg by incorporating client-specific metrics or fairness constraints into the aggregation logic. This section details our implementation of FedAvg (Section D.2.1), FedF1 (Section D.2.2), and FairFed (Section D.2.3) using the Flower framework.

D.2.1 Implementing the FedAvg strategy

Federated Averaging (FedAvg) was used as a baseline for FL. This strategy combines model updates with a weighted average proportional to the number of data samples used by each client in local training.

This strategy was used in the project by directly calling the `flwr.server.strategy.FedAvg` class from the Flower library. When the `experiments/ml_pipeline_experiments.py` script reads the `strategy:FedAvg` setting from the `configs/federated.yaml` file, it activates this standard strategy. Parameter aggregation was performed in the `custom_aggregate` function of the `src/federated/aggregations/fedavg.py` file.

It reflects the mathematical formulation in Equation 4.1 presented in Section 4.5.1 exactly. This process begins with processing the list of results arriving at the server; from the results of each client, the updated model parameters after local training (w_{t+1}^k in the formula) and the number of samples used in training (n_k) are extracted. Simultaneously, the sample counts from all clients are summed to calculate the total number of samples (n), which forms the denominator of the weighted average. Then, weighting is performed by multiplying each client's model parameters by its sample count ($w_{t+1}^k \times n_k$). In the final stage, these weighted parameters are summed layer-by-layer (the $\sum$ corresponds to the formula using the `reduce(np.add, ...)` operation), and the resulting sum is divided by the overall number of samples (n). This sequence of operations creates new global model parameters (w_{t+1}) that reflect each client's contribution scaled according to the data size and prepares them for the next federated learning round.

```

1 def custom_aggregate(results: List[Tuple[NDArrays, int]]) → NDArrays:
2     # Step 1:
3     # Extract client parameters (w_k) and sample counts (n_k)
4     weights_results = [
5         (fit_res[1].parameters, fit_res[1].num_examples)
6         for fit_res in results
7     ]
8
9     # Step 2:
10    # Calculate the total number of samples (n)
11    num_examples_total = \
12        sum(num_examples for _, num_examples in weights_results)
13
14    # Step 3:
15    # Multiply each client's parameters
16    # by its number of samples (w_k * n_k)
17    weighted_weights = [
18        [layer * num_examples for layer in weights]
19        for weights, num_examples in weights_results
20    ]
21
22    # Step 4:
23    # Sum the weighted parameters and divide
24    # by the total number of samples
25    weights_prime: NDArrays = [
26        reduce(np.add, layer_updates) / num_examples_total
27        for layer_updates in zip(*weighted_weights)
28    ]
29    return weights_prime

```

D.2.2 Implementing the FedF1 strategy

The F1 Score Weighted Aggregation (FedF1) strategy, developed within this thesis to improve performance on imbalanced datasets, weights clients not based on their data size but rather on their F1 score obtained with the validation dataset in that round. This approach is formulated in Equation 4.2, presented in Section 4.5.2 of the thesis. The FedF1 strategy is implemented within the project's `src/federated/aggregations/fedf1.py` file, using a class that inherits from the FedAvg strategy and customizes the `custom_aggregate` method. This structure allows for reusing FedAvg's core mechanisms while only modifying the weight calculation logic. The `custom_aggregate` function, which demonstrates the basic aggregation logic of the FedF1 strategy, is shown below.

```
1 def custom_aggregate(results: List[Tuple[NDArrays, int]]) → NDArrays:
2
3     # Specifies the name of the metric to be used for weighting
4     parametre1 = 'l_f1-score'
5
6     # Step 1:
7     # Extract parameters (w_k) and metrics of each client
8     weights_results = [
9         (parameters_to_ndarrays(fit_res.parameters), fit_res.metrics)
10        for i, (_, fit_res) in enumerate(results)
11    ]
12
13    # Step 2:
14    # Calculate the total F1 score (F1_t)
15    num_param_total1 = \
16        sum(dict_[parametre1] for (_, dict_) in weights_results)
17
18    # Step 3:
19    # Multiply each client's parameters by the F1 score (w_k * f1_k)
20    weighted_weights = [
21        [layer * metric[parametre1] for layer in weights]
22        for weights, metric in weights_results
23    ]
24
25    # Step 4:
26    # Add the weighted parameters and divide by the total F1 score
27    weights_prime: NDArrays = [
28        reduce(np.add, layer_updates) / num_param_total1
29        for layer_updates in zip(*weighted_weights)
30    ]
31
32    return weights_prime
```

D.2.3 Implementing the FairFed strategy

In the thesis's fairness focused experiments, we implemented the FairFed approach, which aims to optimize group fairness on the server side. This strategy

is based on the mathematical formulation detailed in Section 4.6.4 of the thesis and presented in Equations 4.5, and 4.6. FairFed’s core principle is to dynamically adjust each client’s model update weight based on the difference between its local fairness metric (EOD) and the global fairness metric on the server. This way, clients that are closer to the global target, more fair, contribute to the model’s generalization.

Implementing the FairFed strategy requires specialized aggregation logic that processes client metrics, similar to FedF1. Unlike FedAvg and FedF1, this logic weights the fairness difference (Δt_k) calculated on the server in each round, rather than the F1 score or the number of samples.

```

1 def custom_aggregate(results: List[Tuple[NDArrays, int]]) → NDArrays:
2
3     # Step 1:
4     # Total number of samples and collection of client data
5     weights_results = []
6     total_examples = 0
7     client_edos = {}
8     for _, fit_res in results:
9         ndarrays = parameters_to_ndarrays(fit_res.parameters)
10        n = fit_res.num_examples
11
12        i=0
13        while i < 5:
14            key = f"equal.opportunity.difference_{i}"
15            if key in fit_res.metrics:
16                eod = fit_res.metrics[key]
17                if i not in client_edos.keys():
18                    client_edos[i] = eod
19                break
20            i += 1
21        weights_results.append((ndarrays, n, eod))
22        total_examples += n
23
24    # Step 2:
25    # Global EOD (weighted average)
26    eods = [eod for _, n, eod in weights_results]
27    nums = [n for _, n, _ in weights_results]
28    eod_global = \
29        sum(e * n for e, n in zip(eods, nums)) / total_examples
30    print("G_eod: ", eod_global)
31
32    # Step 3:
33    # Basis weights
34    base_weights = [n / total_examples for n in nums]
35
36    # Step 4:
37    # Exponential reweighting
38    raw_weights = [
39        b * math.exp(-self.fairness_budget * abs(eod_i - eod_global))
40        for b, eod_i in zip(base_weights, eods)
41    ]
42    Z = sum(raw_weights)
43    norm_weights = [w / Z for w in raw_weights]

```

```

44
45 # Step 5:
46 # Layer-by-layer weighted average
47 num_layers = len(weights_results[0][0])
48 aggregated: NDArrays = []
49 for layer_idx in range(num_layers):
50     layer_updates = [
51         client_ndarrays[layer_idx] * w
52         for (client_ndarrays, _, _) , w \
53             in zip(weights_results, norm_weights)
54     ]
55     aggregated.append(reduce(np.add, layer_updates))
56 return aggregated

```

Implementing the FairFed strategy requires fairness-focused computation on the server side. The function's operation can be divided into five main steps. First, the results from the clients are processed; each client's model parameters, sample count (n_k), and locally calculated EOD score are extracted. In the second step, a global EOD metric is calculated by weighting the local EOD scores from all clients by the number of client data samples. This is a measure of the current fairness of the entire system. In the third step, as in the standard FedAvg strategy, each client's baseline weights (n_k/n) are generated based on their data size. In the fourth step, these baseline weights are adjusted according to the fairness objective. Each client's baseline weight is multiplied by an exponential function (`math.exp(...)`) that contains the absolute difference between the client's local EOD and the global EOD. This "exponential reweighting" mechanism, controlled by the `fairness_budget` hyperparameter, reduces the weight of clients whose fairness metric deviates from the global target (less fairness) while increasing the weight of clients who produce fairer outcomes. These raw weights are then normalized so that their sum is 1. In the final step, each client's model parameters are multiplied by these newly calculated, fairness-normalized weights, and the weighted parameters from all clients are summed layer-by-layer to create the new global model.

D.3 Integration with Different Algorithms and Model Parameters

Flower framework allowed for the integration of various machine learning algorithms into the federated learning system. The `models/factory.py` script acts as a central factory, responsible for creating an instance of the selected model based on the configuration files. The federated client, defined in `src/federated/clients/`, then handles the extraction and updating of the model's learnable parameters. The integration approach differed based on the nature of the algorithms used.

D.3.1 Natively compatible models

These models are inherently suitable for federated learning via parameter averaging because their intelligence is stored in numerical arrays (tensors or vectors) that can be mathematically averaged in a meaningful way.

- **Logistic Regression (LR) and Linear SVM:** For these models, implemented using Scikit-learn, the learnable parameters consist of the model's coefficients (`coef_`) and its intercept (`intercept_`). The federated client extracts these two NumPy arrays, sends them to the server for aggregation, and receives the averaged versions back. These new averaged parameters are then set on a new, clean model instance on the client side to be used in the next round of local training.
- **Multi-Layer Perceptron (MLP):** The MLP model, implemented in PyTorch, has its parameters stored in its `state_dict`. This dictionary contains the weight and bias tensors for each layer of the neural network. The client logic extracts these tensors, converts them into a list of NumPy arrays, and sends them for aggregation. The server aggregates these arrays element-wise, and the resulting global parameters are sent back to the clients, who then load them back into their local model using the `load_state_dict` method.

This direct parameter extraction and averaging mechanism makes these models perfectly compatible with the FedAvg, FedF1, and FairFed strategies implemented in this thesis.

D.3.2 Tree based models

Unlike models defined by numerical weightings, such as LR or MLP, tree-based ensemble models, such as RF and XGB, pose a particular challenge for FL. The parameters of these models are structural information such as split points, features used, and tree depths. Mathematically averaging these structures does not produce a meaningful result.

To overcome this problem, our project implemented an advanced federation scheme, fundamentally different from parameter averaging and known in the literature as Federated Bagging or Model Fusion. This approach is implemented with the special FedXgbBagging strategy [69], located in the project's `src/federated/aggregations/fedxgbbagging_fedavg_fedf1.py` file.

The basic logic of the strategy is detailed in the following code blocks (Check Flower's "Tree-based Bagging Aggregation" documentation [69] for more detail) and explanations.

```

1 def aggregate(
2     bst_prev_org: Optional[bytes],
3     bst_curr_org: bytes,
4     client_weights: int = 1
5 ) → bytes:
6     """Conduct bagging aggregation for given trees."""
7     if not bst_prev_org:
8         bst_curr = json.loads(bytearray(bst_curr_org))
9         for idx, tree_curr in enumerate( \
10             bst_curr['learner']['gradient_booster']['model']['trees']):
11             base_weights = [
12                 bw * client_weights
13                 for bw in tree_curr['base_weights']
14             ]
15             bst_curr['learner']['gradient_booster'] \
16                 ['model']['trees'][idx] = base_weights
17         return bst_curr_org
18
19     tree_num_prev, _ = _get_tree_nums(bst_prev_org)
20     _, paral_tree_num_curr = _get_tree_nums(bst_curr_org)
21
22     bst_prev = json.loads(bytearray(bst_prev_org))
23     bst_curr = json.loads(bytearray(bst_curr_org))
24
25     learner = bst_prev["learner"]
26     learner["gradient_booster"]["model"]["gbtree_model_param"][
27         "num_trees"
28     ] = str(tree_num_prev + paral_tree_num_curr)
29     iteration_indptr = learner["gradient_booster"]["model"][
30         "iteration_indptr"
31     ]
32     learner["gradient_booster"]["model"]["iteration_indptr"].append(
33         iteration_indptr[-1] + paral_tree_num_curr
34     )
35
36     # Aggregate new trees
37     trees_curr = \
38     bst_curr["learner"]["gradient_booster"]["model"]["trees"]
39     for tree_count in range(paral_tree_num_curr):
40         base_weights = [
41             bw * client_weights
42             for bw in trees_curr[tree_count]['base_weights']
43         ]
44         trees_curr[tree_count]['base_weights'] = base_weights
45
46         trees_curr[tree_count]["id"] = tree_num_prev + tree_count
47         learner = bst_prev["learner"]
48         learner["gradient_booster"]["model"]["trees"].append(
49             trees_curr[tree_count]
50         )
51         learner["gradient_booster"]["model"]["tree_info"].append(0)
52
53     bst_prev_bytes = bytes(json.dumps(bst_prev), "utf-8")
54     return bst_prev_bytes

```

The operation of the above code can be divided into the following steps:

1. Initial Case (First Client Model): The function starts by checking whether the parameter `bst_prev_org` (previous global model) is empty. If it is empty, it means the first client of the round has been encountered. In this case, the model from the client (`bst_curr_org`) is directly taken as the basis of the global model. However, before taking it, the `base_weights` at the leaves of each tree within the model are weighted by multiplying them by the `client_weights` (number of samples or F1 score) to reflect that client's contribution. This process makes the first client's model, with its weights adjusted, the first version of the global model.
2. Append to Existing Model (Other Clients): If the server already has a global model, the function performs a more complex append operation:
 - Preparing Metadata: First, metadata such as tree counts of both the existing global model (`bst_prev`) and the new model (`bst_curr`) from the client are read.
 - Global Model Update: The total number of trees in the global model on the server (`num_trees`) is incremented by the number of newly added trees. Similarly, the `iteration_indptr` list, which allows XGBoost to track which iteration each tree belongs to, is updated with the number of new trees. This ensures a consistent structure for the global model.
 - Merging and Weighting New Trees: The function processes each tree within the client model individually. Two basic operations are performed for each new tree:
 - (a) Just like in the initial case, the `base_weights` values at the leaves of the tree are weighted by multiplying them by the `client_weights`.
 - (b) A new, unique identification number (`id`) is assigned to the tree to avoid conflicts with other trees in the global model.
 - Adding to Model: The weighted and re-identified tree prepared in this way is added to the tree list (`bst_prev["learner"]["..."]["trees"]`) of the global model.

Finally, the JSON structure of the global model, thus enriched and updated with all new trees, is converted back to byte format and stored on the server, waiting for the next client's contribution or to complete the round. This mechanism implements Federated Bagging by combining trees from different clients into a single, more robust ensemble model, rather than averaging parameters.

CURRICULUM VITAE

Mustafa AKTAŞ

EDUCATION:

- **B.Sc.:** 2021, Istanbul Medipol University, School of Engineering and Natural Sciences, Computer Science & Engineering Department

PROFESSIONAL EXPERIENCE AND REWARDS:

- 2021-Present : Data Scientist | TUBITAK - BILGEM, Cloud Computing And Big Data Research Laboratory (B3LAB)

PUBLICATIONS, PRESENTATIONS AND PATENTS ON THE THESIS:

- **Aktaş, M.**, Halepmollası, R., & Töreyn, B. U. (2024, December). Enhancing Credit Risk Assessment with Federated Learning through A Comparative Study. In *Proceedings of the 8th EAI International Conference on Robotic Sensor Networks*.

OTHER PUBLICATIONS, PRESENTATIONS AND PATENTS:

- **Aktaş, M.**, & Ateş, H. F. (2021). Small Object Detection and Tracking from Aerial Imagery. In *Proceedings of the 6th International Conference on Computer Science and Engineering (UBMK)* (pp. 688–693).