Bank Customer Churn

Muhammad Mahmudur Rahman

EC Utbildning

Projekt i Data Science

202411

Abstract

Bank customer churn prediction is essential in the banking industry. This project employs machine learning and deep learning techniques using the Bank Customer Churn dataset from Kaggle. Logistic regression, SVM, XGBoost, and deep learning models were evaluated, with the best model deployed via Streamlit for interactive predictions. Significant predictors of churn were identified, aiding banks in proactive retention strategies.

Innehållsförteckning

1 Inledning

Kundavhopp är ett betydande problem inom många branscher, särskilt i banksektorn. När kunder slutar göra affärer med ett företag kan det leda till förlorade intäkter och ökade kostnader för att förvärva nya kunder. Detta fenomen gör det nödvändigt för företag att utveckla strategier för att förutsäga och förebygga kundavhopp. Genom att analysera kundbeteenden och identifiera potentiella avhoppare kan banker vidta åtgärder för att behålla dessa kunder och därigenom minska sina förluster.

Betydelsen av att kunna förutsäga kundavhopp har ökat med utvecklingen av avancerade analysmetoder, maskininlärning och djupinlärning. Dessa tekniker gör det möjligt att bearbeta stora mängder data och upptäcka mönster som inte är uppenbara vid en ytlig analys. Genom att använda maskininlärning och djupinlärning kan banker inte bara identifiera kunder som löper hög risk att lämna, utan även förstå de bakomliggande orsakerna till detta beteende.

Detta projekt syftar till att analysera ett omfattande dataset som innehåller information om bankkunder. Genom att förbehandla data och tillämpa olika maskininlärnings- och djupinlärningsmodeller kommer vi att försöka förutsäga vilka kunder som sannolikt kommer att lämna banken. De mest framgångsrika modellerna kommer att utvärderas och den bästa modellen kommer att implementeras i en interaktiv webbaserad applikation med hjälp av Streamlit.

Syftet med denna rapport är att utveckla en prediktiv modell för kundavhopp inom banksektorn. För att uppfylla detta syfte kommer följande frågeställningar att besvaras:

1. Vilka faktorer har störst inverkan på kundavhopp?
2. Hur presterar olika maskininlärnings- och djupinlärningsmodeller i att förutsäga kundavhopp?
3. Hur kan en interaktiv applikation användas för att tillämpa den prediktiva modellen i praktiken?

Genom att besvara dessa frågor hoppas vi kunna ge insikter och verktyg som kan hjälpa banker att förbättra sina strategier för kundbevarande och minska frekvensen av kundavhopp.

2 Teori

2.1 Explpratory Data Analysis (EDA)

Exploratory Data Analysis (EDA) är en viktig metod inom dataanalys som används för att undersöka och förstå datamängdens struktur och egenskaper innan mer avancerade analyser genomförs. EDA omfattar flera steg som inkluderar att sammanfatta datasetets huvudsakliga karakteristika, identifiera mönster och trender samt att visualisera data för att få insikter om dess underliggande struktur.

En av de första uppgifterna inom EDA är att granska datasetets grundläggande statistiska mått, såsom medelvärde, median, standardavvikelse, samt min- och maxvärden och kvartiler. Detta ger en övergripande bild av datafördelningen och hjälper till att identifiera extrema värden eller avvikelser.

Visualisering är en central del av EDA, där olika typer av grafer och diagram används för att upptäcka trender och mönster. Histogram och boxplot-diagram används för att visa fördelningen av enskilda variabler och för att identifiera potentiella outliers. Spridningsdiagram (scatter plots) kan avslöja relationer mellan två variabler, medan värmekartor (heatmaps) och korrelationsmatriser används för att visualisera samband mellan flera variabler.

En annan viktig del av EDA är hanteringen av saknade värden och felaktigheter i datasetet. Detta kan innebära att fylla i saknade värden, ta bort irrelevanta data eller korrigera inkonsekvenser. Att identifiera och förstå dessa problem tidigt är avgörande för att säkerställa kvaliteten och noggrannheten i den efterföljande analysen.

EDA kan även omfatta mer avancerade tekniker som klustring för att identifiera grupper eller segment inom data samt Principal Component Analysis (PCA) för att reducera datans dimensioner och identifiera de mest inflytelserika variablerna.

Sammanfattningsvis är EDA en iterativ och kreativ process som syftar till att utforska data på ett sätt som möjliggör upptäckt av oväntade mönster och relationer. Genom att kombinera statistiska tekniker och visuella verktyg ger EDA en solid grund för vidare analys och modellering, vilket är avgörande för att fatta välgrundade beslut baserade på data.[1, 2]

2.2 Dataförbehandling

Dataförbehandling är ett viktigt steg inom dataanalys och maskininlärning som syftar till att förbereda rådata för att modeller ska kunna tränas och byggas effektivt och tillförlitligt. Denna process involverar flera kritiska aktiviteter som är nödvändiga för att optimera modellernas prestanda och säkerställa att de insikter som erhålls är exakta och användbara.

En av de första aktiviteterna i dataförbehandling är att hantera saknade värden, vilket är avgörande eftersom dessa kan leda till förvrängda analyser och slutsatser. Det finns flera tekniker för att hantera saknade data, inklusive imputering, där de saknade värdena ersätts med exempelvis medelvärden, medianer eller andra relevanta statistiska mått.

En annan viktig del av dataförbehandlingen är kodning av kategoriska variabler. Eftersom många maskininlärningsalgoritmer kräver numeriska data, måste kategoriska data konverteras till numeriska format. Detta kan göras med metoder som one-hot encoding eller label encoding, beroende på vilken typ av data som hanteras och vilken modell som ska användas.

Slutligen är skalning av numeriska variabler en nödvändig process för att justera värdena så att de faller inom ett specifikt intervall. Detta är särskilt viktigt för algoritmer som är känsliga för skalan på data. Genom att standardisera eller normalisera numeriska variabler kan modeller tränas mer effektivt och ge mer precisa förutsägelser.

Sammanfattningsvis är noggrann dataförbehandling avgörande för att säkerställa tillförlitliga och korrekta insikter från modellerna, vilket i sin tur bidrar till framgångsrika projekt. Denna process inkluderar hantering av saknade värden, kodning av kategoriska variabler och skalning av numeriska variabler.[3, 4]

2.3 Hantering av obalanserad data

Obalanserad data uppstår när en klass i en dataset är betydligt mer representerad än de andra klasserna. Detta kan leda till att maskininlärningsmodeller blir partiska och presterar sämre på den underrepresenterade klassen. Att hantera obalanserad data är därför viktigt för att bygga robusta och rättvisa modeller.

En vanlig metod för att hantera obalanserad data är SMOTE (Synthetic Minority Over-sampling Technique). SMOTE fungerar genom att syntetiskt skapa nya exempel för den underrepresenterade klassen genom att interpolera mellan befintliga exempel av den klassen. Genom att öka antalet exempel i minoritetsklassen hjälper SMOTE modellen att bättre lära sig representationer för alla klasser.

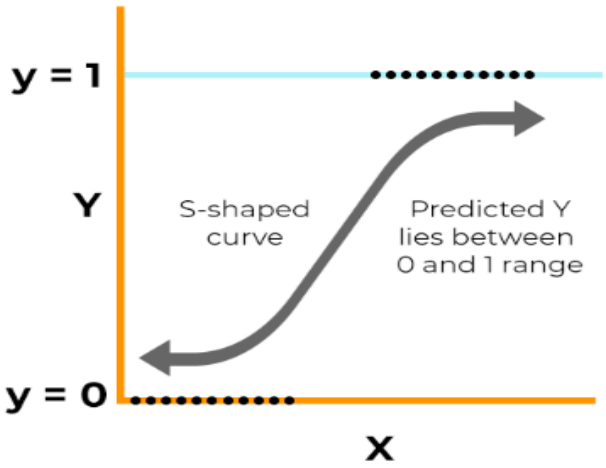
SMOTE-processen börjar med att välja ett exempel från minoritetsklassen och identifiera dess k närmaste grannar. Nya syntetiska exempel skapas sedan genom att interpolera mellan det valda exemplet och dess grannar. Denna teknik har visat sig vara effektiv för att förbättra modellens prestanda och säkerställa att den hanterar obalanserade dataset på ett mer rättvist och korrekt sätt.[5, 6]

2.4 Machine Learning

Vi tillämpade flera maskininlärningsalgoritmer för att jämföra deras prestanda vid förutsägelse av avhopp. Dessa inkluderar:

2.4.1 Logistic Regression

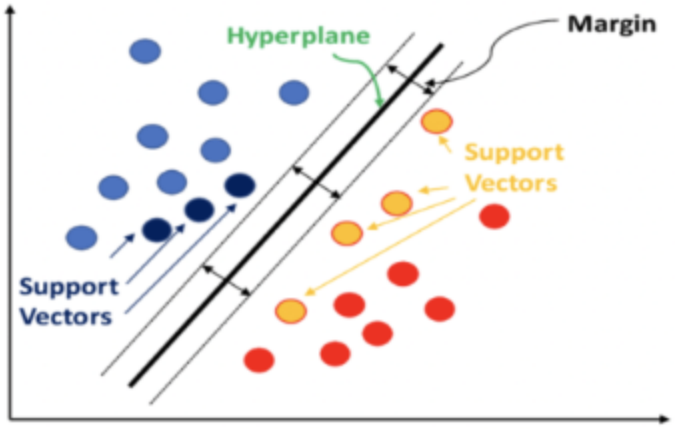
Logistic regression används för binär klassificering och hjälper till att förutspå sannolikheten för avhopp genom att uppskatta modellparametrarna med maximum likelihood-metoden. Denna metod är särskilt värdefull när man vill förstå sannolikheten för en binär utkomst, som om en kund kommer att avsluta sitt konto hos banken eller inte. Modellen använder en sigmoidfunktion för att omvandla en linjär kombination av ingångsvariabler till sannolikheter, vilket gör den lätt att tolka och implementera. Genom att identifiera de variabler som mest påverkar avhoppssannolikheten kan företag utveckla riktade strategier för att motverka dessa faktorer.[7]



Figur 1. Logistic Regression

2.4.2 Support Vector Machine (SVM)

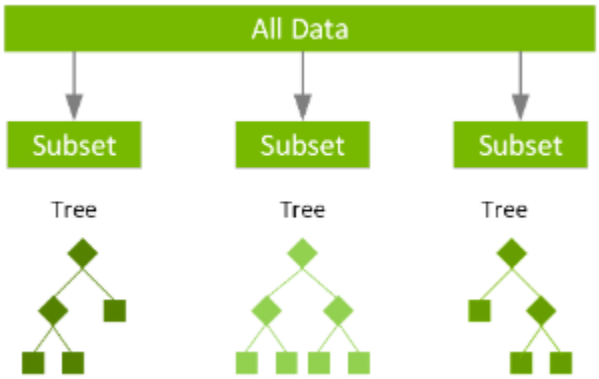
En Support Vector Machine (SVM) identifierar det optimala hyperplanet för att separera olika klasser, vilket gör att kunder kan klassificeras som sannolika att avvika eller inte. SVM är särskilt kraftfull eftersom den kan hantera både linjära och icke-linjära klassificeringsproblem genom att använda olika kärnfunktioner. Genom att maximera marginalen mellan klasserna levererar SVM robusta och exakta klassificeringsresultat. Den är särskilt effektiv när klasserna är klart åtskiljda, men den kan också justeras för att hantera mer komplexa, överlappande data genom att anpassa parametrarna som styr modellens flexibilitet och strafftermer.[8]



Figur 2. Support Vector Machine (SVM)

2.4.3 XGBoost

XGBoost är en kraftfull gradientboosting-modell som successivt bygger upp svaga inlärningsmodeller och kombinerar dem för att skapa en starkare och mer exakt modell. Den har blivit mycket populär tack vare sin höga prestanda och förmåga att effektivt hantera stora dataset. Genom att stegvis konstruera beslutsträd och justera fel från tidigare träd, optimerar XGBoost noggrant sina prediktioner. XGBoost erbjuder även stor flexibilitet genom att tillåta justering av olika hyperparametrar, som inlärningshastighet, antal estimatorer och trädens djup, för att ytterligare förbättra modellens prestanda.[9]

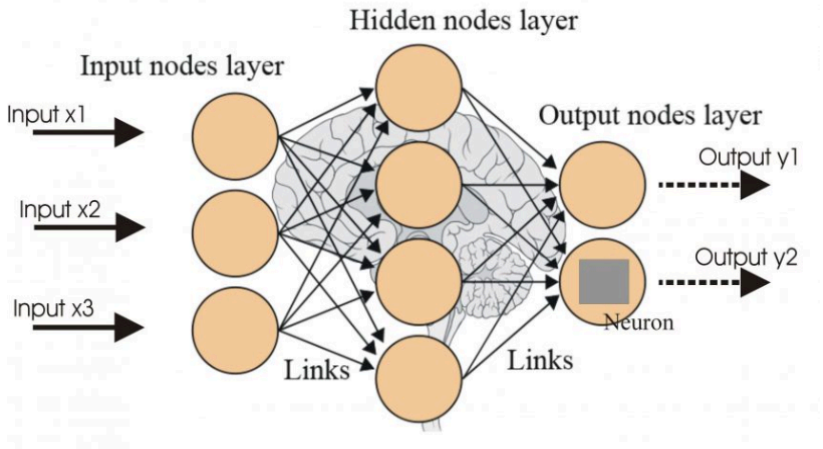


Figur 3. XGBoost-modell

2.5 Deep Learning

2.5.1 Artificial Neural Networks (ANNs)

Artificial Neural Networks (ANNs) är en typ av maskininlärningsmodell som är inspirerad av den mänskliga hjärnans struktur och funktion. Ett neuralt nätverk består av ett antal sammankopplade noder, eller "neuroner," som är organiserade i lager: ett indata lager, ett eller flera dolda lager, och ett utdata lager. Varje nod i ett lager är kopplad till noder i det nästa lagret genom vikter som justeras under träning. ANNs är kraftfulla verktyg för att upptäcka komplexa mönster och samband i data, vilket gör dem särskilt användbara för uppgifter som bild- och taligenkänning, samt för att förutsäga kundavhopp.[10]



Figur 4. Artificial Neural Networks (ANNs)

2.5.2 Träningskomponenter

För att träna ett neuralt nätverk krävs flera viktiga komponenter och steg:

- **Förlustfunktion**: Förlustfunktionen mäter hur bra nätverkets förutsägelser överensstämmer med de faktiska värdena. En vanlig förlustfunktion för binära klassificeringsproblem är binär korsentropi.
- **Optimerare**: En optimerare justerar vikterna i nätverket för att minimera förlusten. Adam är en populär optimerare som anpassar lärhastigheten för varje parameter.
- **Aktiveringsfunktioner**: Aktiveringsfunktioner introducerar icke-linjäritet i nätverket, vilket gör det möjligt att lära sig komplexa mönster. ReLU (Rectified Linear Unit) är en vanligt använd aktiveringsfunktion som endast aktiverar neuroner med positiva indata.
- **Tidigt stopp**: En teknik för att förhindra överanpassning genom att stoppa träningen när modellens prestanda på en valideringsuppsättning slutar förbättras.

Dessa komponenter och tekniker arbetar tillsammans för att säkerställa att det neurala nätverket tränas effektivt och generaliserar bra till ny, osedd data [11].

2.6 Modellutvärdering

När man bygger maskininlärningsmodeller är det avgörande att utvärdera deras prestanda för att säkerställa att de fungerar effektivt och tillförlitligt. Utvärderingsmetoder som acuuracy, precision, recall, F1 score och förvirringsmatris ger viktiga insikter i hur väl en modell presterar och kan hjälpa till att identifiera områden för förbättring.

2.6.1 Accuracy

Accuracy är en grundläggande mätning av en modells prestanda och definieras som andelen korrekta förutsägelser (både positiva och negativa) i förhållande till det totala antalet förutsägelser som modellen gör. Det är en enkel och intuitiv indikator, beräknad enligt formeln:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Figur 5. Accuracy

Här TP, TN, FP, FN är:

- TP (Sant positiva) representerar antalet korrekt klassificerade positiva exempel, såsom patienter med en sjukdom som verkligen har diagnosen.
- TN (Sant negativa) är antalet korrekt klassificerade negativa exempel, till exempel friska patienter som inte har sjukdomen.
- FP (Falskt positiva) är de negativa exempel som felaktigt klassificerades som positiva, vilket kan leda till onödig oro.
- FN (Falskt negativa) är de positiva exempel som modellen missade, vilket kan vara kritiskt i många tillämpningar, som att missa en diagnos.

En hög accuracy indikerar generellt att modellen fungerar väl, men det är viktigt att beakta balansen mellan de olika klasserna, särskilt i obalanserade dataset där en klass kan domineras av den andra.

2.6.2 Precision

Precision är ett mer specifikt mått som fokuserar på kvaliteten av de positiva förutsägelserna. Det mäter andelen verkliga positiva bland alla positiva förutsägelser som modellen gjort och beräknas med formeln:

$$\text{Precision} = \frac{TP}{TP + FP}$$

Figur 6. Precision

Här TP (Sant positiva) är antalet korrekt klassificerade positiva exempel och FP (Falskt positiva) är negativa exempel som felaktigt klassificerades som positiva. En hög precision innebär att modellen gör få falska positiva förutsägelser, vilket är viktigt i situationer där kostnaden för falska positiva är hög, som vid bedrägeridetektering eller sjukdomsdiagnoser.

2.6.3 Recall

Recall, eller sensitivitet, mäter en modells förmåga att korrekt identifiera positiva exempel. Det definieras som andelen verkliga positiva förutsägelser bland alla faktiska positiva fall och beräknas enligt formeln:

$$\text{Recall} = \frac{TP}{TP + FN}$$

Figur 7. Recall

Här TP (Sant positiva) är korrekt klassificerade positiva exempel och FN (Falskt negativa) är de positiva exempel som modellen missade. En hög återkallande indikerar att modellen effektivt fångar de flesta positiva fall, vilket är avgörande i kritiska tillämpningar, som medicinska diagnoser, där det är viktigt att upptäcka alla relevanta positiva exempel, även om det innebär fler falska positiva.

2.6.4 F1 Score

F1 Score är ett viktigt sammanfattande mått som kombinerar både precision och återkallande. Det är särskilt användbart när det är nödvändigt att hitta en balans mellan dessa två mått. F1 Score beräknas med formeln:

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Figur 8. F1 Score

Här precisionen representerar andelen korrekta positiva förutsägelser av alla positiva förutsägelser, och återkallande representerar andelen korrekta positiva förutsägelser av alla faktiska positiva fall. En hög F1 Score indikerar att modellen har en bra balans mellan precision och recall, vilket är viktigt i situationer där det är avgörande att minimera både falska positiva och falska negativa, som i medicinska tillämpningar eller bedrägeridetektering.

2.6.5 Confusion Matrix

Confusion Matrix är en kraftfull visuell representation av hur modellen presterar i klassificeringen av exempel. Den visar antalet korrekta och felaktiga förutsägelser uppdelade efter klass och hjälper till att identifiera specifika typer av fel.

	Prediktion Positiv	Prediktion Negativ
Verklig Positiv	TP	FN
Verklig Negativ	FP	TN

Figur 9. Confusion Matrix

Genom att granska Confusion Matrix kan man förstå om modellen har en tendens att felaktigt klassificera vissa exempel. Denna insikt kan leda till riktade förbättringar av modellen, som att justera algoritmen eller förbättra datakvaliteten.

2.7 GridSearchCV

För att maximera modellens prestanda användes GridSearchCV för att justera hyperparametrar. GridSearchCV utför en systematisk sökning för att hitta den bästa kombinationen av parametrar genom att iterativt utvärdera olika inställningar. Denna metod använder korsvalidering för varje kombination av parametrar, vilket säkerställer att modellen presterar optimalt. Genom att utvärdera olika parametrar med en vald prestandamått, som accuracy eller F1 Score, hjälper GridSearchCV till att hitta de inställningar som ger de mest pålitliga förutsägelserna. Detta är särskilt användbart när det finns många hyperparametrar att välja mellan, eftersom GridSearchCV automatiserar och effektiviserar processen att identifiera de bästa parametrarna för modellen [14].

3 Metod

3.1 Agil arbetsmetodik

Agil arbetsmetodik är en iterativ och flexibel strategi för projektledning som fokuserar på samarbete, anpassningsförmåga och kundcentrering. Genom att använda en agil metodik kan team snabbt reagera på förändringar och kontinuerligt förbättra sina arbetsprocesser. Detta tillvägagångssätt är särskilt effektivt inom mjukvaruutveckling och dataanalys, där krav och omständigheter kan förändras under projektets gång.

I vårt projekt, "Bank Customer Churn," tillämpade vi agil metodik för att säkerställa att vi kunde hantera de utmaningar och förändringar som uppstod under de fem veckor vi arbetade tillsammans. Genom att arbeta i korta iterationer kunde vi snabbt utveckla och utvärdera olika delar av vår lösning, vilket gjorde att vi kunde anpassa oss till nya insikter och feedback från vår kund.

En central del av vår arbetsprocess var regelbundna möten, där vi diskuterade framsteg, identifierade hinder och planerade nästa steg. Dessa möten skapade en öppen kommunikation inom teamet och säkerställde att vi alla var på samma sida, vilket i sin tur ökade vår effektivitet och produktivitet. Vidare involverade vi våra intressenter i utvärderingsprocessen, vilket gav oss möjlighet att få värdefull feedback som direkt påverkade vårt arbete. Denna kundcentrerade strategi hjälpte oss att skapa en lösning som verkligen mötte användarnas behov och förväntningar.

Genom att omfamna agil metodik kunde vi effektivt hantera risker och osäkerhet, vilket resulterade i en högkvalitativ lösning för att förutsäga kundavhopp. Denna erfarenhet har inte bara gett oss en djupare förståelse för agil arbetsmetodik utan har också stärkt vår förmåga att samarbeta och anpassa oss i framtida projekt.

3.2 Scrum

Scrum är en populär ramverk inom agil metodik som strukturerar arbete i korta, tidsbestämda cykler kallade sprinter. I vårt projekt, som pågick i fem veckor, delade vi upp arbetsuppgifterna i två-veckors sprinter, där vi satte upp specifika mål för varje cykel.

Under sprinterplaneringen identifierade vi de mest kritiska funktionerna som behövde utvecklas och bestämde prioriteringarna. Varje sprint inleddes med ett planeringsmöte där vi definierade arbetsuppgifterna och fördelade dem mellan oss.

Vi genomförde dagliga ståuppmöten för att följa upp framstegen, diskutera eventuella hinder och justera planerna vid behov. Vid slutet av varje sprint utvärderade vi vårt arbete genom en sprintgranskning, där vi demonstrerade de funktioner som vi hade slutfört.

Genom att tillämpa Scrum-metoden kunde vi effektivt hantera vårt arbete och anpassa oss till förändrade krav och insikter under projektets gång, vilket bidrog till en högre kvalitet och snabbare leveranser av resultat.

3.3 Datainsamling

Datasetet hämtades från Kaggle och innehöll kundinformation samt en etikett som angav avhopp.

```
# Ignore FutureWarnings only
warnings.filterwarnings("ignore", category=FutureWarning)

#Fetching data from kaggle
dataset_url = 'https://www.kaggle.com/datasets/gauravtopre/bank-customer-churn-dataset'

od.download(dataset_url)
```

Figur 10. Datainsamling

3.4 Datarepresentation

Vi började med att ladda datasetet med hjälp av Pandas-biblioteket och granskade dess dimensioner och de första posterna. Datasetet består av olika funktioner, inklusive:

Datasettet som användes för detta projekt inkluderar flera funktioner som beskriver bankkunderna:

- **customer_id**: Unik identifierare för varje kund.
- **credit_score**: Kundens kreditpoäng, som sträcker sig från 300 till 850. Kreditpoängen kategoriseras vidare i:
    - Dålig kredit: 300 till 579.
    - Rättvis kredit: 580 till 669.
    - God kredit: 670 till 739.
    - Mycket god kredit: 740 till 799.
    - Utmärkt kredit: 800 till 850.
- **country**: Kundens land (Frankrike, Tyskland, Spanien).
- **gender**: Kundens kön (Kvinna, Man).
- **age**: Kundens ålder.
- **tenure**: Antal år kunden har varit hos banken.
- **balance**: Kontobalansen för kunden.
- **products_number**: Antal produkter kunden har köpt.
- **credit_card**: Om kunden har ett kreditkort (1 = Ja, 0 = Nej).
- **active_member**: Om kunden är aktiv (1 = Ja, 0 = Nej).
- **estimated_salary**: Uppskattad årsinkomst för kunden.
- **churn**: Om kunden lämnade banken (1 = Ja, 0 = Nej) - mål för churn-prediktion.

```
#Load the dataset
data = pd.read_csv('bank-customer-churn-dataset/Bank Customer Churn Prediction.csv')
print(f'Dataframe dimensions: {data.shape}')
data_head = data.head(10)
print(tabulate(data_head, headers='keys', tablefmt='fancy_grid', showindex=False))
```

```
Dataframe dimensions: (10000, 12)
```

customer_id	credit_score	country	gender	age	tenure	balance	products_number	credit_card	active_member	estimated_salary	churn
15634602	619	France	Female	42	2	0	1	1	1	101349	1
15647311	608	Spain	Female	41	1	83807.9	1	0	1	112543	0
15619304	502	France	Female	42	8	159661	3	1	0	113932	1
15701354	699	France	Female	39	1	0	2	0	0	93826.6	0
15737888	850	Spain	Female	43	2	125511	1	1	1	79084.1	0
15574012	645	Spain	Male	44	8	113756	2	1	0	149757	1
15592531	822	France	Male	50	7	0	2	1	1	10062.8	0
15656148	376	Germany	Female	29	4	115047	4	1	0	119347	1
15792365	501	France	Male	44	4	142051	2	0	1	74940.5	0
15592389	684	France	Male	27	2	134604	1	1	1	71725.7	0

Figur 11. Datarepresentation

3.5 Databearbetning

Den initiala databearbetningen inkluderade att kontrollera för saknade värden och dubbletter inom datasetet. Vi identifierade förekomsten av saknade numeriska värden och rapporterade antalet dubbletter för att säkerställa dataintegritet.

```
# Check missing numeric values
missing_numeric = data.isnull().sum()
missing_df = pd.DataFrame(missing_numeric, columns=['Missing Values'])
print(tabulate(missing_df, headers='keys', tablefmt='fancy_grid'))
```

	Missing Values
customer_id	0
credit_score	0
country	0
gender	0
age	0
tenure	0
balance	0
products_number	0
credit_card	0
active_member	0
estimated_salary	0
churn	0

Figur 12. Databearbetning

3.6 Dataförbehandling

Den här steg inkluderade:

- Hantering av saknade värden och kodning av kategoriska variabler.
- Skalning av funktioner för att normalisera numeriska värden och minska bias.

Så här ser våra data ut efter förbehandling:

Complete Preprocessed Dataset for Churn Model:

	credit_score	age	tenure	balance	products_number	credit_card	active_member	estimated_salary	country_France	country_Germany	country_Spain	gender_Female	gender_Male
0	0	42	2	0	1	1	1	0	1	0	0	1	0
1	0	41	1	0	1	0	1	0	0	0	1	1	0
2	0	42	8	0	3	1	0	0	1	0	0	1	0
3	0	39	1	0	2	0	0	0	1	0	0	1	0
4	0	43	2	0	1	1	1	0	0	0	1	1	0

	churn
0	1
1	0
2	1
3	0
4	0

dtype: int64

Figur 16. Dataförbehandling

3.7 Hantering av klassobalans

SMOTE tillämpades för att åtgärda klassobalans, vilket gjorde att modellen bättre kunde särskilja mellan avhoppade och icke-avhoppade kunder.

```
XGBoost Validation Accuracy (with SMOTE): 0.81
XGBoost Test Accuracy (with SMOTE): 0.81
XGBoost Classification Report (Test, with SMOTE):
              precision    recall  f1-score   support

           0       0.89      0.88      0.88      1200
           1       0.53      0.55      0.54       300

    accuracy                           0.81      1500
   macro avg       0.71      0.71      0.71      1500
weighted avg       0.82      0.81      0.81      1500
```

Figur 17. Prognoser och utvärdering med SMOTE för XGBClassifier-modellen

```
SVM Validation Accuracy (SMOTE): 0.82
SVM Test Accuracy (SMOTE): 0.82
SVM Classification Report (Test (SMOTE):
              precision    recall  f1-score   support

           0       0.90      0.87      0.89      1200
           1       0.55      0.63      0.59       300

    accuracy                           0.82      1500
   macro avg       0.73      0.75      0.74      1500
weighted avg       0.83      0.82      0.83      1500
```

Figur 18. Prognoser och utvärdering med SMOTE för SVM-modellen

3.8 Datasplittring

Datasetet delades upp i tränings- och testset med en 70-30-fördelning för att möjliggöra modellvalidering på osedda data.

```
Original data: 10000 rows
Training data: 7000 rows
Training data (SMOTE): 11094 rows
Validation data: 1500 rows
Test data: 1500 rows
```

Figur 19. Datasplittring

3.9 Modellimplementering

Logistic Regression Model

Vi valde att inte använda SMOTE med Logistic Regression för att undvika överanpassning på grund av syntetiska data. Istället använde vi class_weight='balanced' för att hantera klassobalans.

- **Valideringsnoggrannhet**: 0,70
- **Testnoggrannhet**: 0,72

XGBoost Model

- **Utan SMOTE**: XGBoost uppnådde en accuracy på 84% men hade en låg recall på 45% för churnfall, vilket indikerade att den var bättre på att förutsäga icke-churn men missade många churn-kunder.
- **Med SMOTE**: Recall för churnfall förbättrades till 55%, men den övergripande accuracy minskade något till 81%, vilket resulterade i en mer balanserad modell men med fler falska positiva.

Support Vector Machine Model (SVM)

- **Utan SMOTE**: SVM uppnådde en testnoggrannhet på 79%, med en recall på 74% för churnfall (klass 1) men en låg precision på 49%, vilket indikerade många falska positiva.
- **Med SMOTE**: Testnoggrannheten förbättrades till 82%, med recall för churnfall på 63% och precision som förbättrades något till 55%.

Efter jämförelse fastställdes att SVM (med SMOTE) var den bäst presterande modellen baserat på noggrannhet, recall och precision.

3.10 Hyperparameter tuning

GridSearchCV användes för att finjustera hyperparametrarna för SVM-modellen. Trots justeringen förblev modellens prestanda nästan densamma, med en testnoggrannhet på 0,82, en liten nedgång i precision för klass 1 (från 0,55 till 0,54), och en liten förbättring i recall (från 0,63 till 0,65).

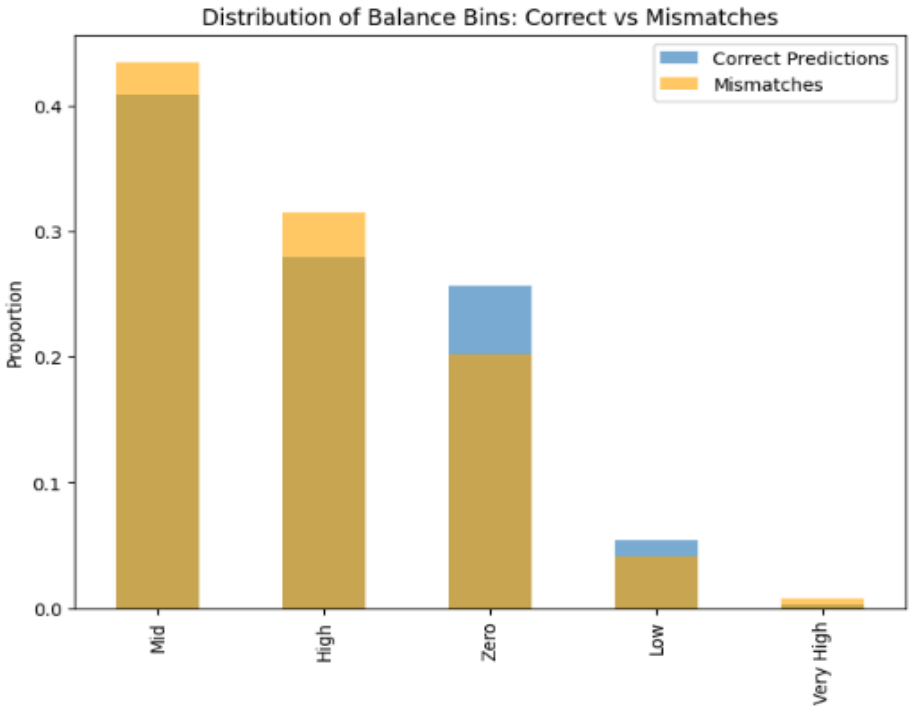
3.11 KS Test

KS-testresultaten visar signifikanta skillnader mellan korrekta förutsägelser och felaktigheter för vissa variabler:

- **Balance**: Mycket lågt p-värde (1.10e-08), vilket tyder på att modellen har problem med denna funktion.
- **Age**: P-värde på 4.40e-06, vilket indikerar inkonsekvenser i hanteringen av denna funktion.
- **Credit_score, estimated_salary, tenrue och land (Germany och Spain)**: Höga p-värden, vilket indikerar att dessa funktioner hanteras mer konsekvent av modellen.

3.12 Balance Bins

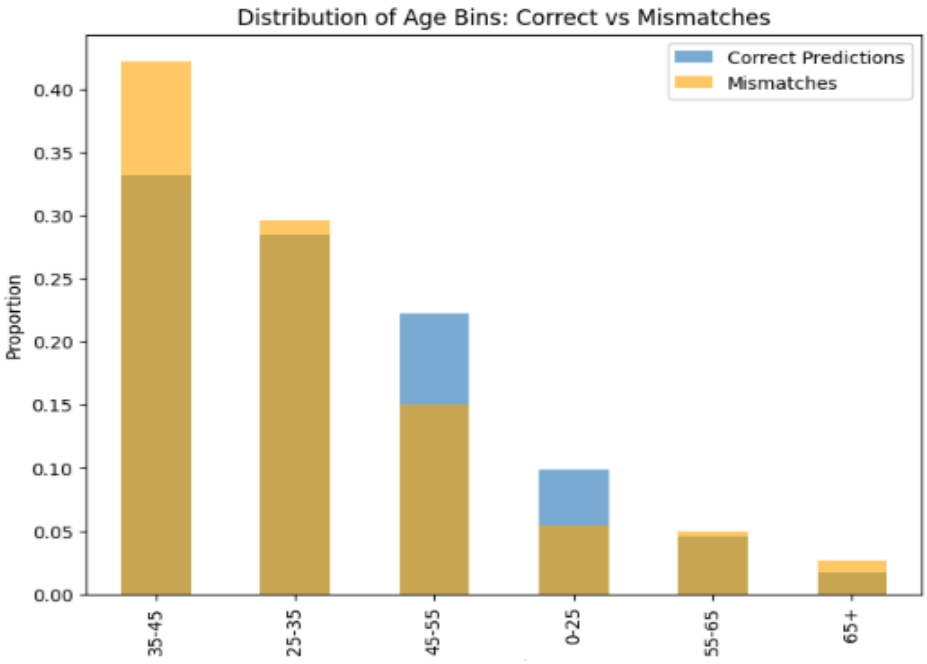
Modellen har störst problem med kunder i medel (0,2 - 0,5) och hög (0,5 - 0,8) balansintervall. För kunder med noll balans presterar modellen bättre, men det finns fortfarande fel. I låg balansgrupp (0 - 0,2) är felklassificeringsgraden högre än korrekta förutsägelser. I den mycket höga balansgruppen (över 0,8) är de flesta kunder felklassificerade, även om det finns färre exempel i denna grupp.



Figur 20. Distribution of Balace Bins

3.13 Age Bins

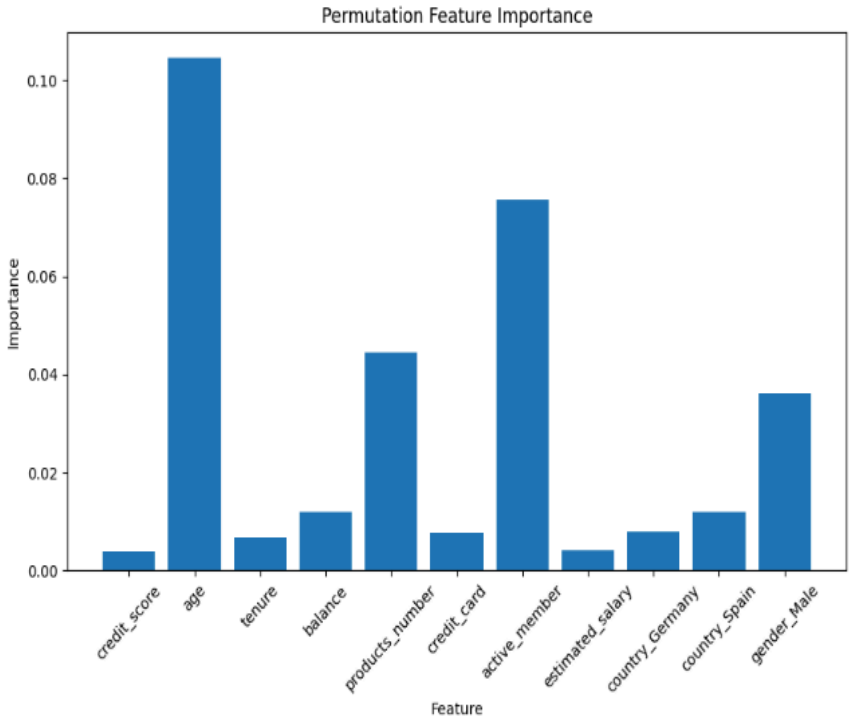
Modellen har störst problem med kunder i åldern 35-45 år. Betydande fel förekommer också i åldersgrupperna 25-35 och 45-55 år. Modellen presterar relativt bra för yngre (0-25 år) och äldre (55+ år) kunder.



Figur 21. Distribution of Age Bins

3.14 Permutation Feature Importance

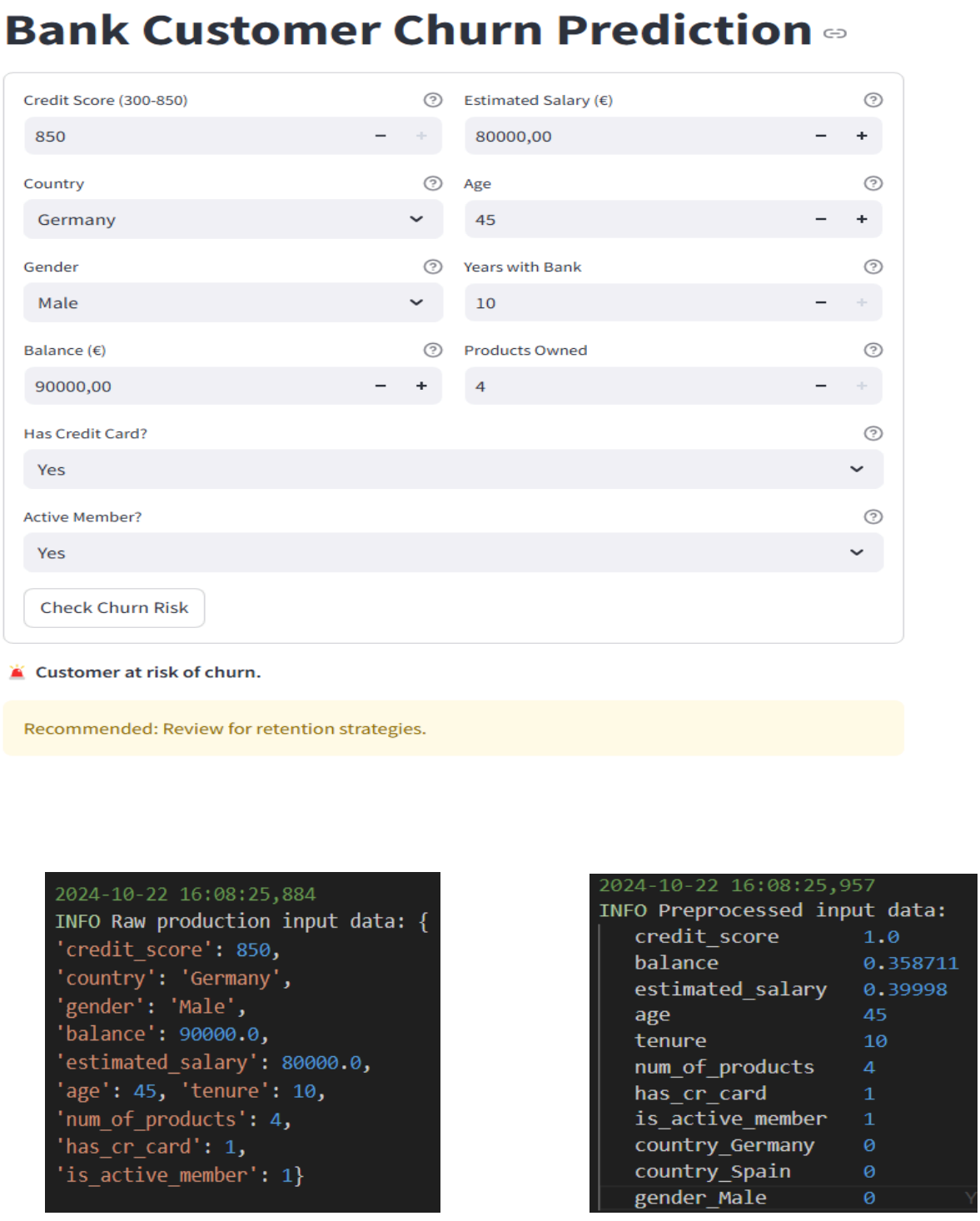
I det här projektet användes permutation feature importance för att förstå påverkan av varje funktion på modellens förutsägelser. Denna metod innebär att värdena för varje funktion i testuppsättningen blandas, och minskningen av modellens prestanda, mätt i noggrannhet, observeras. En större minskning indikerar en viktigare funktion.



Figur 22. permutation feature importance

3.15 Streamlit

Den slutliga modellen implementerades med Streamlit, vilket ger ett användarvänligt webbgränssnitt för churn-prediktion. Detta gränssnitt gör det möjligt för intressenter att ange kunddata och få omedelbara förutsägelser. Genom att använda Streamlit kan användarna enkelt interagera med modellen och se resultatet av sina inmatningar i realtid, vilket underlättar beslutsfattande och strategiska insikter för att hantera kundretention effektivt.



Figur 23. Implementering med Streamlit

4 Resultat och Diskussion

Resultaten från vårt projekt visar tydliga skillnader i prestanda mellan de olika modellerna för att förutsäga customer churn. Den Logistic Regression Model, som vi implementerade med class_weight='balanced', uppnådde en valideringsnoggrannhet på 70 % och en testnoggrannhet på 72 %. Även om detta är en acceptabel grundlinje blev det tydligt att mer avancerade modeller skulle kunna erbjuda bättre resultat.

XGBoost Model utan SMOTE uppvisade den högsta accuracy bland maskininlärningsmodellerna, med en accuracy på 84 %. Trots detta resulterade den i en låg recall på 45 % för churnfall, vilket indikerar att modellen hade svårigheter att korrekt identifiera kunder som var benägna att lämna. Genom att implementera SMOTE förbättrades recall till 55 %, men den övergripande accuracy minskade något till 81 %, vilket gav en mer balanserad modell med fler falska positiva.

Support Vector Machine (SVM) visade sig vara en stark kandidat; utan SMOTE nådde den en testnoggrannhet på 79 % och en recall på 74 % för churnfall. Efter att ha använt SMOTE ökade testnoggrannheten till 82 %, vilket visade att SVM med SMOTE var den bäst presterande modellen utifrån accuracy, recall och precision.

När vi implementerade en djupinlärningsmodell uppnådde den en testnoggrannhet på 78,2 %, med både precision och recall på 78 %. F1 Score och AUC-värdet på 0,783 bekräftar modellens förmåga att särskilja mellan churn och icke-churn. Trots att djupinlärningsmodellen inte överträffade XGBoost i accuracy, erbjöd den en konkurrenskraftig och balanserad prestation.

Analysen av funktionsvikterna visade att faktorer som balans, ålder, anställningstid och antalet produkter som innehas var avgörande för att förutsäga churn. Modellens svårigheter att identifiera kunder med medel- och högbalans samt i åldersgruppen 35-45 år indikerar att dessa variabler kräver ytterligare granskning och potentiella justeringar.

5 Slutsatser

Detta projekt demonstrerar den framgångsrika tillämpningen av både maskininlärning och djupinlärning för att förutsäga kundchurn inom banksektorn. Genom att implementera algoritmer som Logistic Regression, XGBoost, Support Vector Machine (SVM) och en djupinlärningsmodell har vi identifierat nyckelfaktorer som påverkar kundretention, inklusive kontobalans, ålder, anställningstid och antalet produkter som innehas.

Genom att genomföra KS Test, följt av analyser av Balance Bins och Age Bins, har vi fått insikter om hur olika faktorer påverkar churn-beteendet. Dessa analyser visar på variationer i kundernas beteende och hjälper till att förstå vilka grupper som är mer benägna att lämna. Genom att till sist utföra Permutation Feature Importance har vi ytterligare förstått varje funktions påverkan på modellens förutsägelser, vilket ger djupare insikter i vilka faktorer som är mest kritiska för att förutsäga churn.

Implementeringen av dessa modeller via Streamlit ger banker värdefulla insikter om kundbeteende, vilket möjliggör proaktivt engagemang med kunder som löper risk att lämna. Framtida förbättringar kan innefatta att utforska mer avancerade modeller, såsom ensemble-tekniker, samt säkerställa regelbundna uppdateringar med nya data för att upprätthålla och förbättra noggrannheten.

Sammanfattningsvis belyser detta projekt potentialen för maskininlärning och djupinlärning att förbättra kundretention och stärka relationerna inom banksektorn genom att erbjuda insikter och verktyg för att effektivt hantera riskabla kundrelationer.

6 Självutvärdering

1. Utmaningar du haft under arbetet samt hur du hanterat dem?

Under arbetet med projektet "Bank Customer Churn" stötte jag på utmaningar relaterade till klassobalans i datasetet, vilket påverkade accuracy och recall av våra modeller. För att hantera detta implementerade jag SMOTE (Synthetic Minority Over-sampling Technique) för att syntetiskt öka antalet exempel på churn-kunder, vilket bidrog till en bättre balans i datamängden och förbättrade modellernas förmåga att identifiera kunder med hög risk för avhopp. Jag hade också svårigheter med hyperparameteroptimering, men genom att använda GridSearchCV kunde jag justera parametrarna för Support Vector Machine (SVM) och uppnå en liten förbättring i modellens prestanda.

2. Vilket betyg du anser att du skall ha och varför?

Jag anser att jag bör få betyg VG, eftersom jag har visat förmåga att tillämpa olika maskininlärningstekniker och djupinlärning för att analysera och förutsäga customer churn. Jag har också genomfört en noggrann utvärdering av modellerna och implementerat en användarvänlig lösning med Streamlit, vilket visar på en god förståelse för både teknik och affärsrelevans.

3. Något du vill lyfta fram till lärare?

Efter att ha genomfört detta projekt har jag lärt mig mycket, och min kunskap inom Data Science har ökat avsevärt. Jag vill också rikta ett stort tack till den bästa läraren, Antonio Prgomet, för Projekt i Data Science-kursen. Hans vägledning och stöd har varit ovärderliga under hela processen.

Källförteckning

1. Waller, W., & Fawcett, T. (2013). "Data Science for Business: What You Need to Know about Data Mining and Data-Analytic Thinking." Hämtad från O'Reilly Media (https://www.oreilly.com/).
2. GeeksforGeeks. (2021). "Exploratory Data Analysis in Python." Hämtad från GeeksforGeeks (https://www.geeksforgeeks.org/).
3. Brownlee, J. (2017). "Data Preparation for Machine Learning." Hämtad från Machine Learning Mastery (https://machinelearningmastery.com/).
4. scikit-learn. (n.d.). "Data Preprocessing." Hämtad från scikit-learn (https://scikit-learn.org/stable/).
5. Chawla, N. V., et al. (2002). "SMOTE: Synthetic Minority Over-sampling Technique." Hämtad från SpringerLink (https://link.springer.com/).
6. Towards Data Science. (2020). "Handling Imbalanced Data: SMOTE Technique." Hämtad från Towards Data Science (https://towardsdatascience.com/).
7. Brownlee, J. (2016). "Logistic Regression for Machine Learning." Hämtad från Machine Learning Mastery (https://machinelearningmastery.com/).
8. Brownlee, J. (2017). "Support Vector Machines (SVM) for Machine Learning." Hämtad från Machine Learning Mastery (https://machinelearningmastery.com/).
9. Chen, T., & Guestrin, C. (2016). "XGBoost: A Scalable Tree Boosting System." Hämtad från arXiv (https://arxiv.org/).
10. Goodfellow, I., Bengio, Y., & Courville, A. (2016). "Deep Learning." Hämtad från Deep Learning Book (https://www.deeplearningbook.org/).
11. Brownlee, J. (2019). "A Gentle Introduction to ReLU and Other Activation Functions." Hämtad från Machine Learning Mastery (https://machinelearningmastery.com/).
12. scikit-learn. (n.d.). "Metrics and scoring: quantifying the quality of predictions." Hämtad från scikit-learn (https://scikit-learn.org/stable/).
13. Brownlee, J. (2018). "A Gentle Introduction to ROC Curves and AUC." Hämtad från Machine Learning Mastery (https://machinelearningmastery.com/).
14. scikit-learn. (n.d.). "GridSearchCV." Hämtad från scikit-learn (https://scikit-learn.org/stable/).