Churn-prediktion med maskininlärning

End-to-end pipeline för kundbeteendeanalys

Lence Majzovska

EC Utbildning

Projekt i Data Science

2025/10

Abstract

This report presents a machine learning approach for predicting customer retention risk using sales data. The goal is to design an end-to-end pipeline that transforms transactional data into actionable insights through preprocessing, feature engineering, model training, and evaluation. A logistic regression model is identified as the most effective baseline, achieving stable performance and interpretable results using RFM-derived customer features. The results are visualized in Power BI to demonstrate how to support business-oriented decision-making and risk segmentation.
Overall, the project demonstrates how a data-driven churn prediction framework can be implemented, evaluated, and prepared for future automation and integration with real-world data.

Innehållsförteckning

1 Inledning

I takt med att företag samlar in allt större mängder kunddata ökar behovet av att kunna analysera beteendemönster och förutse framtida affärsrisker. En central utmaning är att identifiera vilka kunder som riskerar att lämna.

Genom att kombinera historisk försäljningsdata med maskininlärning kan företag tidigt upptäcka förändringar i kundbeteenden och agera proaktivt. På så sätt kan risken för kundbortfall minska och kundlojaliteten stärkas över tid.

Detta projekt undersöker hur maskininlärning kan användas för att förutsäga kundbortfall genom att bygga en end-to-end pipeline som omfattar databehandling, feature engineering, modellträning, kalibrering och export till Power BI. Syftet är att visa hur en datadriven lösning kan implementeras, från inläsning och bearbetning av data till ett färdigt beslutsunderlag och hur prediktiv analys kan integreras i ett affärsinriktat BI-flöde.

Arbetet baseras på ett publikt e-handelsdataset som simulerar verklig orderhistorik. Även om resultaten inte är direkt generaliserbara visar projektet hur samma metodik och kodstruktur kan tillämpas på verklig data. Projektet fungerar därmed som proof-of-concept för hur prediktiv kundanalys kan implementeras i en pipeline som är förberedd för praktisk användning.

Frågeställningar:

1. Hur kan kunddata förberedas och användas för att bygga en prediktiv churn-modell?
2. Vilka maskininlärningsmetoder ger bäst balans mellan noggrannhet och tolkbarhet?
3. Hur kan resultaten visualiseras och användas i ett affärsinriktat beslutsstöd (BI)?

2 Teori

Detta kapitel presenterar teoretiska grunder inom prediktiv kundanalys: churn som analytiskt problem, segmentering via RFM, de använda modellerna samt utvärderingsmått. I rapporten används begreppet *churn* som synonym till kundbortfall.

2.1 Klassificeringsproblem

Inom maskininlärning innebär ett klassificeringsproblem att en modell tränas för att förutsäga vilken klass en observation tillhör utifrån dess egenskaper. Två huvudtyper är vanliga:

- **Binär klassificering:** två möjliga utfall, exempelvis kund stannar/lämnar.
- **Multiclass-klassificering:** fler än två utfall.

Churn betraktas som ett binärt problem där målet är att avgöra om en kund är aktiv (0) eller har lämnat (1). Kundbortfall sker sällan slumpmässigt utan påverkas av observerbara faktorer såsom köpfrekvens, engagemang och relationens längd. Ett vanligt kännetecken är att datan är obalanserad, det vill säga att andelen churnade kunder är betydligt lägre än andelen aktiva. Detta påverkar modellens träning och kräver särskilda utvärderingsmått som tar hänsyn till klassfördelningen.

2.2 RFM-analys och kundsegmentering

RFM (Recency, Frequency, Monetary) är en etablerad metod för att beskriva och segmentera kundbeteenden:

- **Recency:** tid sedan senaste köp - indikerar hur aktuell kundrelationen är.
- **Frequency:** antal köp under en given period - relaterat till lojalitet/engagemang.
- **Monetary:** total spendering - speglar ekonomiskt värde.

Genom att kombinera dessa tre dimensioner kan kunder delas in i grupper med liknande beteendemönster, till exempel lojala, växande eller inaktiva kunder. Inom prediktiv kundanalys används RFM som en teoretisk bas eftersom måtten fångar centrala aspekter av kundrelationens styrka och kan användas som insatsvariabler för att förutsäga sannolikheten för churn.

2.3 Maskininlärning och prediktiv modellering

Maskininlärning (ML) är en del av artificiell intelligens (AI) som fokuserar på att låta datorer lära sig mönster i data och fatta beslut baserat på tidigare observationer. I prediktiva analyser används oftast *supervised learning* (övervakad inlärning), där modeller tränas på historiska data med kända utfall för att kunna förutsäga framtida beteenden.

En central utmaning vid modellträning är *overfitting*, där modellen lär sig detaljer i träningsdatan som inte generaliserar till ny data. För att motverka detta används tekniker som *stratified cross-validation* och begränsning av modellkomplexitet. S*tratified cross-validation* delar upp datan i flera delar (*folds)* som turas om att användas för träning och test, samtidigt som klassfördelningen bevaras i varje delmängd. Detta ger en mer tillförlitlig uppskattning av modellens prestanda vid obalanserade datamängder.

2.4 Modellutvärdering

För att bedöma en modells prestanda används flera statistiska mått som beskriver både dess noggrannhet och förmåga att skilja mellan klasser. Vid obalanserade datamängder är det särskilt viktigt att komplettera traditionell noggrannhet med mått som tar hänsyn till felklassificeringar.

De centrala måtten är:

- Precision: andelen korrekta positiva prediktioner av alla som modellen klassificerat som positiva.
- Recall: andelen verkliga positiva observationer som modellen identifierat korrekt.
- F1-score: det harmoniska medelvärdet av precision och recall, ett mått på balansen mellan noggrannhet och känslighet.
- ROC-AUC: ett tröskeloberoende mått som beskriver modellens förmåga att skilja mellan klasser över alla möjliga beslutsgränser.

Dessa mått används i kombination för att jämföra modeller och välja den som bäst balanserar precision, recall och generell prestanda.

2.5 Kalibrering och tolkbarhet

För att en prediktiv modell ska vara användbar i praktiken krävs inte bara god prestanda utan även tillförlitliga sannolikheter och tolkbarhet. En modell kan ha hög noggrannhet men ändå vara dåligt kalibrerad, vilket innebär att sannolikheterna inte motsvarar verkliga utfall.

Kalibrering justerar predikterade sannolikheter så att de bättre speglar den faktiska risken. Två vanliga metoder är:

- Sigmoidkalibrering: baserad på logistisk regression, lämplig vid små datamängder.
- Isotonic regression: en icke-linjär metod som ger en flexibel anpassning och fungerar bättre vid större datamängder.

För att förstå varför en modell gör vissa prediktioner används metoder för modellförklarbarhet:

- SHAP (SHapley Additive exPlanations): visar hur varje variabel påverkar modellens förutsägelse för en enskild kund.
- Permutation Importance: mäter hur mycket modellens prestanda försämras när en variabel slumpas om och därmed förlorar sin informationskraft.

Tillsammans bidrar kalibrering och förklarbarhet till att göra modellerna transparenta, tillförlitliga och användbara i beslutsstöd, snarare än enbart tekniskt korrekta.

3 Metod

Detta kapitel beskriver hur projektet genomförs praktiskt och hur de teoretiska principerna har omsatts i en fungerande, reproducerbar pipeline för prediktiv kundanalys. Arbetet omfattar datainsamling, bearbetning, modellering, kalibrering, tolkning samt kvalitetssäkring genom loggning och testning. Målet har varit att utveckla en pipeline som inte bara fungerar som teknisk demonstration, utan som en grund för vidare tillämpning med verklig verksamhetsdata.

3.1 Datagrund och simulering

Projektet är ursprungligen utformat för att använda verklig order- och kunddata från affärssystem och CRM-plattformar. Eftersom dataleveransen försenades används i nuläget publik e-handelsdata från Kaggle, med motsvarande struktur och informationsinnehåll (kund-ID, orderdatum, kvantitet och pris). Detta möjliggör test och validering av hela pipeline-flödet på ett realistiskt sätt, även utan tillgång till faktisk produktionsdata.

Den centrala inläsningsmodulen `load_orders()` är byggd för att först försöka hämta data via API-anrop och därefter falla tillbaka på lokala filer (CSV/XLSX). Funktionen hanterar autentisering, paginering, felhantering och konverterar inkommande JSON-strukturer till ett internt schema med kolumnerna *customer_id, order_date* och *sales_amount.*

Pipeline-designen är medvetet generisk och kan köras oförändrad oavsett datakälla. När de verkliga API-källorna kopplas in behöver endast datainhämtningen bytas ut, medan efterföljande steg såsom feature-beräkning, modellträning och export till Power BI redan är färdigställda. Detta arbetssätt visar hur simulerad data kan användas systematiskt för att testa och kvalitetssäkra ett färdigt analysflöde innan riktiga datakällor ansluts.

3.2 Datahantering

Efter inläsning av data genomfördes en omfattande rensning och standardisering för att säkerställa datakvalitet, enhetlig struktur och spårbarhet i analysflödet. Arbetet inleddes med att identifiera saknade värden, dubbletter och orimliga observationer. Därefter skapades nya variabler som beskriver kundbeteenden över tid.

De centrala måtten hämtas från RFM-ramverket (Recency, Frequency, Monetary) och beräknas individuellt för varje kund.

Utöver dessa skapades kompletterande beteendevariabler såsom:

- köpaktivitet de senaste 90 dagarna
- säsongsmönster per kvartal (Q1–Q4)
- genomsnittligt ordervärde (AOV) över livstid och de senaste 90 dagarna

En kund definierades som churnad om inga köp genomförts under de senaste 90 dagarna. Tröskelvärdet valdes för att på ett balanserat sätt skilja mellan tillfällig inaktivitet och faktisk churn. För att undvika dataläckage exkluderas variabler som direkt relaterar till den period som används för att definiera churn, såsom indikatorer på nyligen genomförda köp.

Detta säkerställer att modellen endast baseras på information som hade varit känd vid prediktionstillfället. Sammantaget skapar detta en konsekvent och tolkbar kundmatris som är redo för vidare analys och modellering. Genom att kombinera beteendemått och affärslogik läggs grunden för en modell som både är statistiskt hållbar och affärsmässigt relevant.

3.3 Modellval och träning

Tre modeller jämförs baserat på olika nivåer av komplexitet och tolkbarhet. Data delas upp i 75 % träningsdata och 25 % testdata. För att minska slumpmässiga effekter används *stratified 5-fold cross-validation*, där modellerna tränas på olika delmängder av datan. Detta ger en mer tillförlitlig uppskattning av modellens generaliseringsförmåga. Det separata testsetet används endast i slutet för att utvärdera den modell som presterar bäst.

3.3.1 Logistic Regression

Logistic Regression är en linjär modell som uppskattar sannolikheten för churn. Den är enkel, snabb och lätt att tolka, vilket gör den lämplig som baslinjemodell för jämförelse mot mer komplexa metoder. Modellen implementeras i en pipeline tillsammans med *StandardScaler* för att normalisera indata och förbättra stabiliteten.

3.3.2 Random Forest

Random Forest är en ensemblemetod som kombinerar många beslutsträd för att minska variation och överanpassning. Modellen kan fånga icke-linjära samband och interaktioner mellan variabler, vilket gör den väl lämpad för mer komplexa kundmönster.

3.3.3 XGBoost

XGBoost är en gradientförstärkt ensemblemetod som bygger beslutsträd sekventiellt och korrigerar tidigare fel. Modellen används för att undersöka om mer avancerad boosting kan förbättra prediktionsprecisionen.

3.3.4 Modelljämförelse

Logistic Regression fungerar som en linjär referensmodell med hög förklarbarhet, medan Random Forest och XGBoost utvärderar om icke-linjära samband kan förbättra prediktionen.

Modellernas prestanda jämförs med hjälp av *F1-score, ROC-AUC*, *precision-recall* och *Precision@K*. Den modell som uppnår bäst balans mellan dessa mått väljs för vidare kalibrering och tolkning.

Samtliga modeller tränas med explicit hantering av klassobalans: `class_weight` för *Logistic Regression* och *Random Forest* samt `scale_pos_weight` för *XGBoost*. Detta säkerställer att modellerna inte favoriserar majoritetsklassen.

3.4 Kalibrering

Efter modelljämförelsen kalibrerades den bästa modellen med hjälp av *CalibratedClassifierCV* för att säkerställa att de predikterade sannolikheterna motsvarade faktiska risknivåer. Metoden justerar sannolikhetsfördelningen med antingen *sigmoid*- eller *isotonic regression*, beroende på datamängdens storlek och modellens egenskaper.

Syftet är att en predikterad sannolikhet på till exempel 0,7 också ska innebära att cirka 70 % av kunderna i den gruppen faktiskt churnar i praktiken. Detta ökar modellens användbarhet som beslutsstöd, eftersom risknivåerna kan tolkas som faktiska sannolikheter snarare än relativa poäng.

3.5 Förklarbarhet

För att förstå vilka faktorer som har störst påverkan på churn-risken används metoder för modellförklarbarhet.

Linjära modeller analyseras med *Permutation Importance*, som mäter hur mycket modellens prestanda försämras när en variabel slumpas om. Det ger en uppskattning av varje variabels globala betydelse och visar vilka beteendemått som har störst påverkan på sannolikheten för churn.

För trädmodellerna används *SHAP* vid behov för att visa varje variabels bidrag till prediktionen på individnivå, vilket möjliggör mer detaljerad analys av icke-linjära samband.

3.6 Loggning och testning

För att säkerställa spårbarhet, transparens och reproducerbarhet implementerades både loggning och enhetstester i projektets pipeline. Syftet är att skapa en robust lösning där varje steg i analysflödet kan följas, verifieras och upprepas vid behov.

En central loggkonfiguration definieras i modulen *log_config.py,* som används av samtliga delmoduler i projektet. Loggningen dokumenterar nyckelsteg i pipeline-flödet såsom datainläsning, rensning, feature-generering, modellträning och export.

Terminalutskriften är färgkodad för att tydligt skilja informationsmeddelanden, varningar och fel, medan en detaljerad loggfil automatiskt sparas i mappen *logs/*. Detta underlättar felsökning och möjliggör analys av pipelinekörningar i efterhand.

För att verifiera att pipeline-komponenterna fungerar korrekt även vid förändringar i källdata genomfördes enhetstester med `pytest`. Testerna säkerställer att centrala funktioner beter sig som förväntat och returnerar giltiga resultat.

De omfattar tre huvudområden:

- test_data_prep.py - verifierar att inläsning, rensning och feature engineering returnerar dataramar med korrekta format och datatyper.
- test_model.py - kontrollerar att modellträning, utvärdering och valideringsmått fungerar enligt förväntan.
- test_export.py - säkerställer att exportfiler skapas korrekt, innehåller rätt kolumner och sparas i avsedda format.

Genom loggning och testning utvecklas och utvärderas pipelinen kontinuerligt utan att stabiliteten påverkas. Arbetssättet ökar tillförlitligheten och säkerställer att pipeline-flödet är robust och reproducerbart i praktisk användning.

3.7 Riskscore och export till BI

Efter att modellen tränats och kalibrerats beräknas ett individuellt risk score för varje kund, vilket motsvarar den predikterade sannolikheten för churn. För att göra resultaten mer begripliga i ett affärssammanhang delas dessa sannolikheter in i fyra kategorier (*Low, Medium, High och Critical*) baserat på percentiler av riskfördelningen.

Riskbanden sätts efter percentilgränserna: Low (0–60 %), Medium (60–85 %), High (85–95 %) och Critical (95–100 %). Detta fungerar som en förenklad tolkning av modellens sannolikheter och gör det möjligt att snabbt identifiera kundgrupper med högst churnrisk. Indelningen exporteras tillsammans med modellens riskpoäng till Power BI, där den används för visualisering och segmentanalys.

För att säkerställa spårbarhet och reproducerbarhet sparas även modellens metadata, såsom versionsinformation, träningsdatum och använda variabler, i en separat JSON-fil. Den slutliga modellen exporteras i joblib-format för framtida användning.

Dessutom skapas index i SQLite-databasen för att optimera prestanda vid analys i Power BI. På detta sätt blir hela exportflödet automatiserat, dokumenterat och redo för integration i ett beslutsstödsystem.

4 Resultat och diskussion

Detta kapitel presenterar resultaten från den prediktiva analysen av churn baserat på den slutliga modellen, Logistic Regression. Fokus ligger på modellens prestanda och de viktigaste insikterna kring vilka faktorer som påverkar churn-risken.

4.1 Modelljämförelse

Tre modeller utvärderades med 5-fold cross-validation. Logistic Regression uppnådde högst genomsnittlig AUC, följt av Random Forest och XGBoost. Skillnaderna mellan modellerna är små, vilket delvis kan förklaras av att ingen hyperparameter-optimering genomförts och att datan är begränsad i volym. De mer komplexa modellerna har sannolikt inte kunnat utnyttja sin fulla kapacitet under standardinställningarna.

Logistic Regression valdes därför inte för att den presterade markant bättre, utan för att den uppvisar stabil och reproducerbar prestanda samt är enkel att tolka. Resultaten visar att enklare modeller kan vara mer robusta och praktiskt användbara när datan är begränsad, medan avancerade ensemblemetoder kräver större och mer varierad data samt optimerade parametrar för att komma till sin rätt.

Modell	AUC (mean)	AUC (std)	folds
LogReg (baseline)	0.743	0.018	5
RandomForest	0.726	0.017	5
XGBoost	0.714	0.013	5

Tabell 1: Genomsnittlig AUC per modell med standardavvikelse över 5 folds.

4.2 Testresultat

Den slutliga modellen, Logistic Regression, utvärderas på testdatan med flera tröskelvärden och kompletterande mått. Modellen uppnår AUC ≈ 0.74 och F1 ≈ 0.68, vilket visar en god balans mellan precision och recall. Prestandan är stabil även vid obalanserad data, vilket gör modellen lämplig för riskbedömning i affärssammanhang.

Trots sin enkelhet visar modellen att beteendevariabler baserade på RFM-principen kan förklara en stor del av variationen i churn-beteende. Det tyder på att kundens aktivitet över tid är mer informativ än enskilda monetära värden, vilket är viktigt vid prioritering av marknads- och lojalitetsinsatser.

Modell	AUC	F1@0.5	Best F1	Best Threshold	Precision@10%
LogReg (baseline)	0.739	0.666	0.676	0.344	0.671

Tabell 2: Utvärdering av den slutliga modellen, Logistic Regression, på testdata.

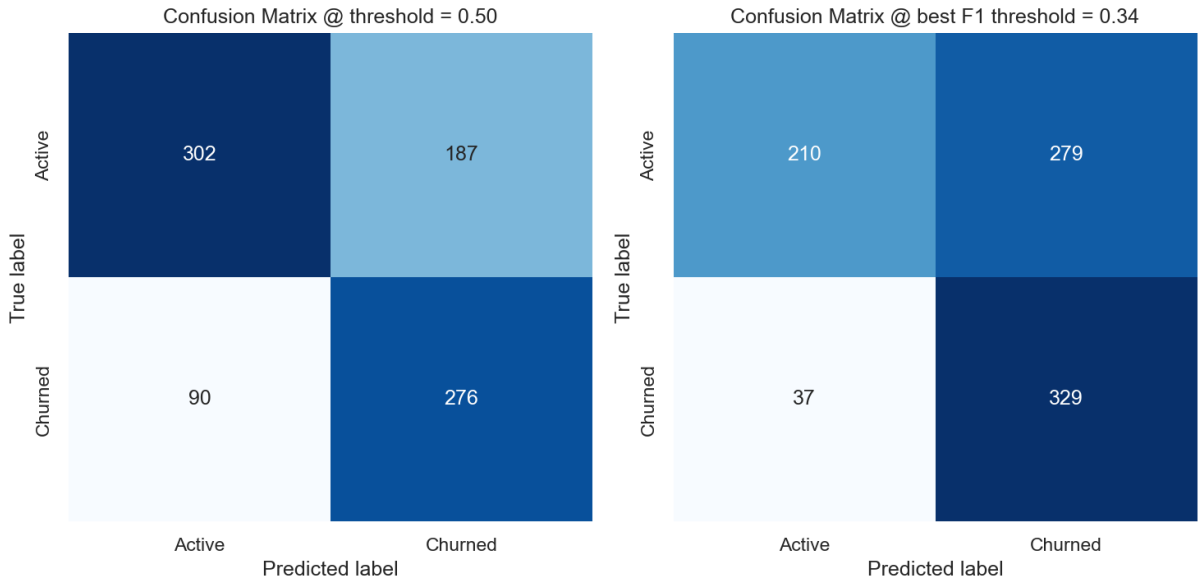
4.3 Visualiseringar

Detta avsnitt presenterar ett urval av visualiseringar som används för att tolka modellens beteende och utvärdera dess praktiska användbarhet.

4.3.1 Confusion matrix

Confusion-matriserna visar modellens träffsäkerhet vid olika tröskelvärden. När tröskeln sänks från 0.50 till 0.34 identifieras fler churnade kunder *(True Positive*) från 276 till 329, medan antalet felaktigt klassificerade aktiva kunder *(False Positive*) ökar från 187 till 279.

Detta visar att modellen blir mer känslig (högre recall) men något mindre exakt, vilket är en medveten och ofta önskvärd avvägning i churn-analys. I praktiken är det bättre att identifiera en riskkund i onödan än att missa en kund som faktiskt är på väg att lämna.



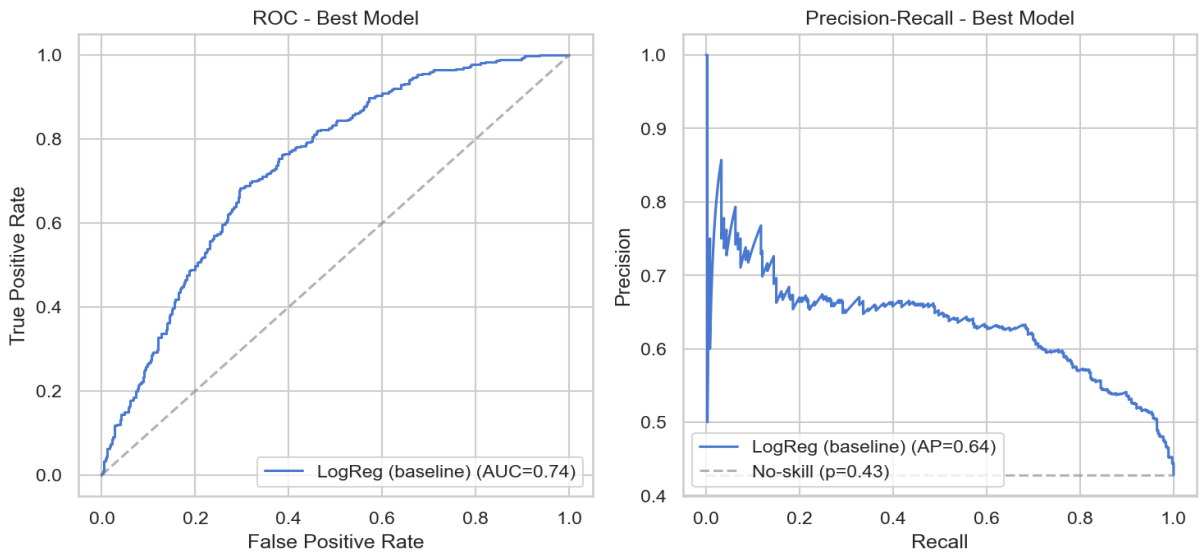
Figur 1: Confusion-matriser för Logistic Regression vid två olika tröskelvärden (0.50 och 0.34).

4.3.2 ROC- och Precision-Recall kurvor

ROC- och Precision–Recall kurvor används för att utvärdera modellens förmåga att skilja churnade kunder från aktiva över olika tröskelvärden (se *Figur 2*). ROC-kurvan visar modellens diskrimineringsförmåga, det vill säga hur väl den skiljer churnade kunder från aktiva.

Ett AUC-värde på 0.74 indikerar god prestanda, tydligt bättre än slumpnivån (0.5). Precision-Recall kurvan är särskilt informativ vid obalanserad data. Modellen uppnår ett Average Precision (AP) på 0.64 jämfört med baslinjen på 0.43, vilket visar att precisionen förblir stabil även när recall ökar.

Sammantaget visar kurvorna att modellen kan rangordna kunder på ett sätt som är praktiskt användbart för kampanjstyrning och riskhantering, där resurser kan riktas mot de kunder som har högst churnrisk.

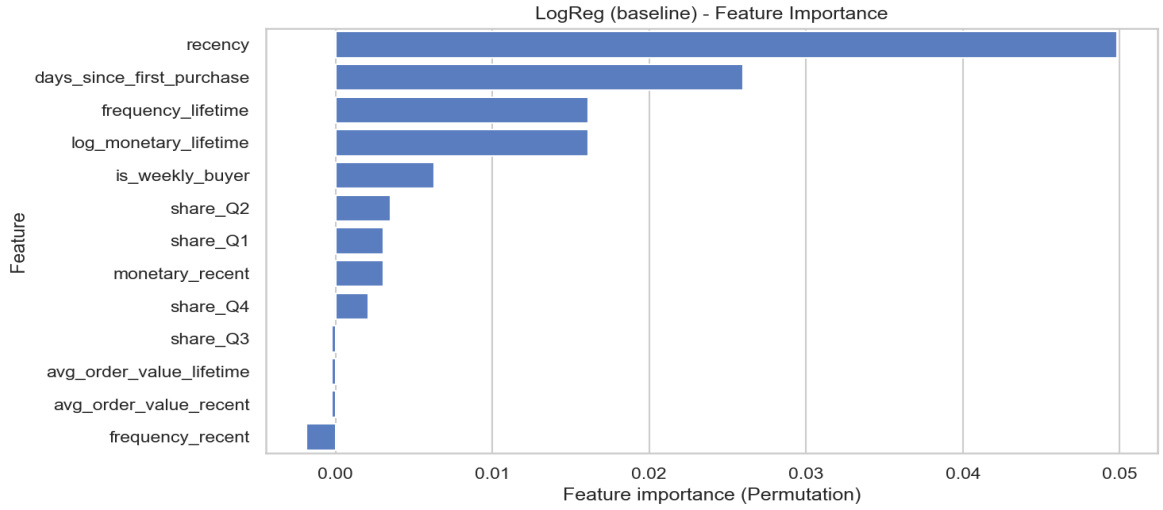


Figur 2: ROC- och Precision–Recall-kurvor för den slutliga Logistic Regression-modellen.

4.3.3 Feature importance

Feature importance visar vilka variabler som bidrar mest till modellens prediktioner. R*ecency* (antal dagar sedan senaste köp) är den mest betydelsefulla faktorn, följt av *days_since_first_purchase* och *frequency_lifetime*. Detta indikerar att både kundens aktivitet över tid och relationens längd är centrala för att förutsäga churn.

Monetära mått har viss betydelse, medan säsongsrelaterade variabler påverkar mindre. Resultatet är affärsmässigt rimligt, det är inte hur mycket kunden köpt utan hur nyligen och hur ofta, som främst signalerar risk. Detta stärker modellens trovärdighet och gör resultaten direkt användbara i lojalitets- och kundrelationsarbete.



Figur 3: De mest betydelsefulla variablerna enligt permutation importance för Logistic Regression. Recency och relationens längd har störst inverkan på sannolikheten för churn.

4.4 Kalibrering

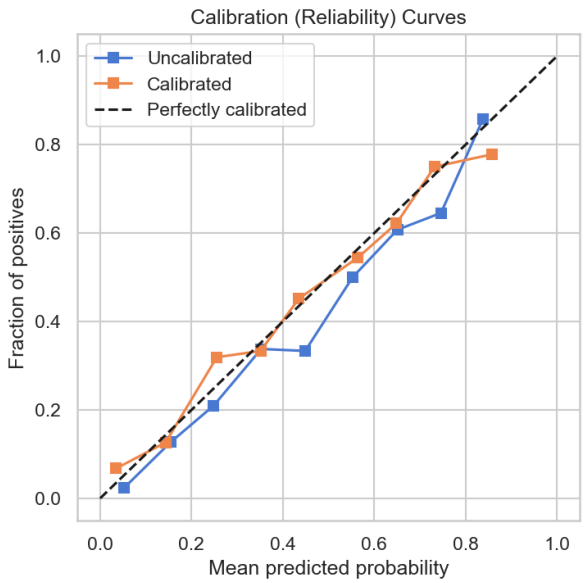
Kalibrering används för att justera modellens sannolikheter så att de bättre motsvarar den faktiska churnrisken. Efter kalibrering med isotonic regression via CalibratedClassifierCV bibehålls AUC-värdet, vilket visar att modellens diskrimineringsförmåga inte försämras. Samtidigt förbättras Precision@ 10 %, vilket innebär att modellen identifierar de mest riskfyllda kunderna mer tillförlitligt.

Den kalibrerade modellen följer den diagonala referenslinjen nära, vilket visar att predikterade sannolikheter motsvarar faktiska utfall. Vid höga sannolikhetsnivåer ses viss överkalibrering, men helheten visar en förbättrad sannolikhetsfördelning som gör modellen mer användbar för praktiska beslut, till exempel kampanjstyrning eller retentionstrategier.

Modell	AUC	F1@0.50	Best F1	Best Threshold	Precision@10%
Uncalibrated	0.739	0.666	0.676	0.344	0.671
Calibrated	0.740	0.606	0.675	0.413	0.706

Tabell 3: Jämförelse av modellens prestanda före och efter kalibrering. AUC förblir stabil medan Precision@10 % ökar, vilket visar en förbättrad identifiering av riskkunder.

Den kalibrerade modellen (isotonic) ligger närmare den diagonala referenslinjen än den okalibrerade, vilket innebär att predikterade sannolikheter bättre motsvarar observerade utfall (70 % risk ≈ 7 av 10 churnar). Vid låga till medelhöga sannolikheter syns tydlig förbättring, medan en viss överkalibrering kan uppträda i det högsta intervallet. Sammantaget ökar kalibreringen sannolikheternas tillförlitlighet utan att nämnvärt ändra AUC (diskrimineringsförmågan), vilket gör modellen mer användbar för tröskelbaserade beslut.



Figur 4: Kalibreringskurvor för Logistic Regression före (blå) och efter (orange) kalibrering med isotonic regression. Den kalibrerade modellen följer referensdiagonalen bättre, särskilt i låga till medelhöga sannolikhetsnivåer.

5 Slutsatser & Lärdomar

Baserat på träningsresultaten presterar samtliga modeller på en liknande nivå. Skillnaderna är små vilket delvis kan förklaras av att datasetet är simulerat med begränsad variation i kundbeteenden. I en sådan kontext har mer komplexa modeller inte mycket att vinna eftersom de icke-linjära mönstren som de normalt fångar saknas i datan.

Logistic Regression väljs som slutlig modell, inte för att den presterar signifikant bättre, utan för att den är stabil, tolkbar och väl lämpad för att demonstrera metodiken. Projektet fungerar därmed som proof of concept som demonstrerar hur churn-prediktion kan implementeras i praktiken.

5.1 Databearbetningens betydelse

Förbehandlingen av data och konstruktionen av RFM-baserade variabler hade en avgörande betydelse för modellernas prestanda. Väl valda beteendevariabler, särskilt *recency* och *frequency*, kunde förklara en stor del av variationen i churn-beteendet.

Att exkludera variabler med direkt koppling till definitionen av churn, såsom köp de senaste 90 dagarna, minskade risken för dataläckage och bidrog till en mer rättvis utvärdering av modellens generaliserbarhet. Datan täcker endast ett års historik, vilket gör det svårt att identifiera långsiktiga trender eller återköpscykler. Dessutom saknas naturliga variationer som kampanjpåverkan och genuina säsongseffekter mellan segment, även om dessa delvis har simulerats i analysen.

5.2 Modellens praktiska användbarhet

Även om datan inte representerar ett verkligt kundunderlag visar projektet hur en prediktiv modell kan implementeras i ett beslutsstödssystem. Genom att visualisera risknivåer och modellresultat i Power BI kan lösningen användas för att simulera scenarier och analysera kundbeteenden i realtid.

Den kalibrerade Logistic Regression-modellen gav sannolikheter som låg nära observerade utfall, vilket ökar dess praktiska trovärdighet som demonstrationsverktyg. Projektet visar därmed potentialen i att kombinera maskininlärning och affärsanalys i en gemensam process, från data till beslut. Samtidigt finns vissa risker och begränsningar i den nuvarande modellen. Logistic Regression utgår från linjära samband mellan variabler, vilket innebär att mer komplexa mönster i kundbeteendet kan förbises. Modellen är också känslig för obalanserade datamängder och kan överskatta sannolikheten för churn i små segment.

5.3 Sammanfattande reflektion

Den färdiga pipelinen visar hur maskininlärning och visualisering kan samverka för att ge insikter om kundbeteenden. Genom att kombinera datahantering, *feature engineering* och en kalibrerad modell skapas en lösning som inte bara förutsäger kundbortfall utan också ger affärsnytta genom visualisering och beslutsstöd i Power BI. Arbetet demonstrerar hur en praktiskt användbar modell kan implementeras och utvärderas på ett transparent sätt.

Loggning och testning har haft en central roll i utvecklingen och visar vikten av en strukturerad och spårbar pipeline. Genom tydlig loggning och automatiserade tester kunde utvecklingen ske iterativt utan att kompromissa med stabiliteten, vilket ökade tillförlitligheten och bidrog till en mer professionell process.

Projektet har genomförts individuellt med ett iterativt arbetssätt som följer agila principer. Fokus låg på att först etablera en fungerande grundpipeline för databehandling och modellering, varefter mer avancerade moment som kalibrering, visualisering och testning infördes successivt.

Vissa delar, såsom integration av verklig data och vidareutveckling av modellens praktiska tillämpning, genomfördes inte inom projektets tidsram på grund av försenad dataleverans. Erfarenheterna understryker vikten av att planera för alternativa scenarier vid osäker tillgång till data.

Sammantaget visar arbetet fördelen med ett iterativt och modulärt angreppssätt, där varje komponent kan förbättras oberoende av övriga delar, vilket resulterar i en hållbar och effektiv utvecklingsprocess.

6 Framtida arbete

Vid fortsatt utveckling bör metoden testas på större och mer varierad data för att bedöma modellens generaliserbarhet. Fler variabler kopplade till kundinteraktion, kampanjer och produktpreferenser kan öka modellens prediktiva kraft.

Det vore även värdefullt att utvärdera tidsbaserade modeller som fångar förändringar i kundbeteende över tid, vilket kan ge en djupare förståelse för hur churn-mönster utvecklas och hur modellen kan anpassas till nya förutsättningar.

Framtida iterationer kan med fördel baseras på verklig verksamhetsdata för att öka modellens tillförlitlighet och praktiska relevans. Dessutom kan hyperparameteroptimering och mer omfattande kalibrering i ett tidigt skede förbättra prestandan ytterligare.

Slutligen kan framtida arbete fokusera på modellens driftbarhet i praktiken, exempelvis genom att automatisera modelluppdateringar eller skapa varningar och kampanjförslag baserat på risknivåer i realtid baserat på förändrade risknivåer.

Appendix A

A.1 Modellkonfiguration

```
# Reproducerbarhet
RANDOM_STATE = 42
np.random.seed(RANDOM_STATE)

# Train/Test-split 75/25 med stratifiering
X_train, X_test, y_train, y_test, ids_train, ids_test = train_test_split(
    X, y, ids, test_size=0.25, random_state=RANDOM_STATE, stratify=y
)

# Beräkna scale_pos_weight för XGBoost
pos = int(y_train.sum())
neg = int(len(y_train) - pos)
spw = (neg / max(pos, 1)) if pos else 1.0
print(f"Calculated scale_pos_weight={spw:.2f} (neg={neg}, pos={pos})")

models = [
     ("LogReg (baseline)", Pipeline([
        ("scaler", StandardScaler()),
        ("clf", LogisticRegression(
            max_iter=1000,
            class_weight="balanced",
            solver="liblinear",
            random_state=RANDOM_STATE
        )),
    ])),

    ("RandomForest", RandomForestClassifier(
        n_estimators=300,
        max_features="sqrt",
        min_samples_leaf=5,
        class_weight="balanced_subsample",
        n_jobs=-1,
        random_state=RANDOM_STATE,
    )),

    ("XGBoost", XGBClassifier(
        n_estimators=400,
        max_depth=4,
        learning_rate=0.05,
        subsample=0.9,
        colsample_bytree=0.9,
        min_child_weight=1.0,
        reg_lambda=1.0,
        scale_pos_weight=spw,
        eval_metric="auc",
        tree_method="hist",
        n_jobs=-1,
        random_state=RANDOM_STATE,
    )),
]
```

A.2 Kalibrering och utvärdering

```
# Kalibrerar modellens sannolikheter (isotonic, fallback till sigmoid)
base_unfit = clone(best_model)
try:
    calib = CalibratedClassifierCV(
        estimator=base_unfit,
        cv=5,
        method="isotonic"
    )
    calib.fit(X_train_pi, y_train)
    calib_method = "isotonic"

except ValueError:
    calib = CalibratedClassifierCV(
        estimator=clone(best_model),
        cv=5,
        method="sigmoid"
    )
    calib.fit(X_train_pi, y_train)
    calib_method = "sigmoid"


# Utvärdera före/efter kalibrering
res_uncal = evaluate_model(
    "Uncalibrated", best_model,
    X_train, y_train, X_test, y_test
)
res_cal   = evaluate_model(
    "Calibrated", calib,
    X_train, y_train, X_test, y_test
)

eval_compare = pd.DataFrame([
    {
        "Model": r["name"],
        "AUC": r["auc"],
        "F1@0.50": r["f1_05"],
        "BestF1": r["f1_best"],
        "BestThr": r["best_thr"],
        "Precision@10": r["precision_at_k"],
    }
    for r in [res_uncal, res_cal]
]).round(3)
```

A.3 Projektlänkar

GitHub-repo:
https://github.com/lencemajzovska/churn-prediction.git

Källförteckning

EC Utbildning. (2025). *Kursmaterial, föreläsningar och handledning från kursen Machine Learning.* Hämtad från skolans lärplattform.

Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow* (2:a upplagan). O'Reilly Media.

Vijayuv. (2023). *Online Retail* [Dataset]. Kaggle.
Hämtad från https://www.kaggle.com/datasets/vijayuv/onlineretail

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, E. (2011). *Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12*, 2825–2830. Hämtad från https://scikit-learn.org/stable/index.html

Lundberg, S. M., & Lee, S.-I. (2017). *A unified approach to interpreting model predictions (SHAP). Advances in Neural Information Processing Systems, 30* (NeurIPS).
Hämtad från https://arxiv.org/abs/1705.07874

Scikit-learn. (2025). *Permutation Feature Importance - scikit-learn documentation.*
Hämtad från https://scikit-learn.org/stable/modules/permutation_importance.html

Microsoft. (2025). *Power BI Documentation - Data Modeling and Visualization.*
Hämtad från https://learn.microsoft.com/power-bi/