UiO : Centre for Entrepreneurship
University of Oslo

Data mining i banksektoren

- Prediksjonsmodellering og analyse av kunder som sier opp boliglån

MSc in Innovation and Entrepreneurship

Fredrik Broch Elgaaen & Nicholas Mowatt Larssen

19.05.2017

Høgskulen på Vestlandet

Oppgavens tittel:	**Data mining i banksektoren**	**Levert dato: 19.05.2017**
Forfatter:	**Fredrik Broch Elgaaen & Nicholas Mowatt Larssen**	
Mastergrad:	**Master of Science in Innovation and Entrepreneurship**	**Tall sider u/vedlegg: 79**
Veileder:	**Olav Kvitastein**	**Tall sider m/vedlegg: 85**
Studieobjekt:	**Kundeavgang i Skandiabanken**	
Metodevalg:	**Cross Industry Standard Process for Data Mining (CRISP-DM)**	

Sammendrag: This study shows how data mining can be used in the banking sector to reduce churn among mortgage customers. A churned customer is defined as a customer who have terminated their mortgage agreement. Our contribution to reduce customer churn is divided into two key actions: predicting customers who will churn and key insights on those who churn.

In a competitive environment, a key to success is keeping your profitable customers. By applying machine learning on data from a major bank in Norway, we have shown that it is possible to predict customers who churn with a precision of 77%. After experimenting with several models we found that XGBoost turned out to be the best fit for this problem.

The customers who churn are younger and have been a customer for a shorter period of time compared to those who do not churn. In addition, they are also less wealthy and use the bank's services less in contrast to the customers who do not churn. By combining predictions with insight we believe that customers in risk of terminating their agreement can be identified at an early stage and retained with the proper measures, which in return will increase the bank's profitability.

Stikkord for bibliotek: Data mining, datavitenskap, maskinlæring, bank, kundeavgang, boliglån

2017

Data mining i banksektoren

Fredrik Broch Elgaaen & Nicholas Mowatt Larssen

http://www.duo.uio.no/

Reprosentralen, Universitetet i Oslo

Innholdsfortegnelse

Forord

Oppgaven er avslutningen på en 2-årig MSc i Innovasjon og entreprenørskap ved Høgskolen på Vestlandet i samarbeid med Universitet i Oslo.

Vi ønsker å takke alle som har gjort oppgaven mulig.

Skandiabanken /v Andreas Øye for tema til oppgaven samt tilgang til data fra Skandiabanken.

Vi har vært så heldig å ha tilgang på tre veiledere som alle har bidratt med gode og konstruktive tilbakemeldinger. Olav Kvitastein, Chunyan Xie og Terje Kristensen.

1. Introduksjon

Den teknologiske utviklingen har gjort banktjenester mer tilgjengelig. Det har aldri vært lettere å bytte bank eller opprette et nytt kundeforhold hos en bank enn det er i dag. Dette har ført til at det er stadig vanligere å ha flere bankforhold. Den store tilgjengeligheten har ført til at kunder er mer sensitiv overfor pris. Media minner en stadig på om at det er penger å spare dersom man flytter boliglånet til den billigste banken, noe som kan gjøres via et par tastetrykk på internett.

I en tid med modne markeder og intens konkurranse innser flere og flere banker at deres viktigste ressurs er de eksisterende kundene. Som et resultat av dette har interessen for kundebevaring økt. De økonomiske fordelene ved å aktivt drive kundebevaring er vidt beskrevet i litteratur. I følge Van del Poel (2004) reduserer suksessfull kundebevaring behovet for å tiltrekke seg nye og potensielt risikable kunder, og lar en fokusere på å tilfredsstille de eksisterende kundene, som igjen kan anbefale banken videre. Å tiltrekke seg nye kunder er fem til seks ganger dyrere enn å bevare eksisterende kunder. Langtidskunder kjøper mer, er billigere å betjene og er mindre sensitiv til konkurrerende tilbud (Van del Poel, 2004).

Etter tusenårsskiftet er det blitt gjort store investeringer i næringslivets infrastruktur, noe som tillater bedriftene å samle inn store mengder data. Digitaliserte prosesser, tjenester og forretningsmodeller er alle kilder til datainnsamling. Den store tilgjengeligheten til data har ført til en økning i interesse for metoder som trekker ut verdifull informasjon og kunnskap fra dataene, som igjen er en viktig ressurs i beslutningstaking.

Med så mye tilgjengelig data har det blitt viktig å utnytte dataene for å oppnå konkurransemessige fortrinn. Tidligere kunne en ta i bruk manuelle metoder for å analysere dataene, men i de senere årene har volumet og variasjonen i dataene overgått kapasitet ved manuell analyse. På samme tid har datamaskinene blitt kraftigere og algoritmer for å oppnå en bredere og dypere forståelse av datasett blitt utviklet. Utviklingen har gitt en mer utbredt anvendelse av datavitenskapelige prinsipper og teknikker.

1.1. Problemstilling

En studie utført av Brynjolfsson mfl. (2011) undersøkte effekten av datadreven beslutningstaking for bedrifter. Studien viste at jo mer datadreven en bedrift er, jo mer

produktiv er den. Forskerne utviklet et mål som rangerer bedrifter etter hvor sterkt data blir brukt i beslutningstakingen. I tillegg til store forskjeller i produktivitet, fant de også at datadreven beslutningstaking er korrelert med høyere avkastning på eiendeler, avkastning på egenkapital, utnyttelse av ressurser og markedsverdi.

I denne oppgaven vil vi forsøke å ta i bruk datavitenskapelige teknikker og maskinlæring for å predikere kundeavgang og hente ut verdifull informasjon om boliglånskundene til Skandiabanken. Innsikten vi oppnår håper vi kan bidra til forbedret kundebevaring i Skandiabanken gjennom datadrevne beslutninger og tidlig identifikasjon av kunder som potensielt skal slutte. Problemstillingen vi ønsker å undersøke er som følger:

Hvordan kan en bruke maskinlæring og data mining til å forbedre kundebevaring i banksektoren?

Problemstillingen deler vi videre inn i to delproblemer som vi mener har stor betydning for effektiviteten av kundebevaringen.

1. *Hvordan kan en predikere kunder som skal slutte?*

Det å nå ut til en kunde før han/hun har bestemt seg for slutte eller har inngått en ny avtale med en annen tilbyder er kritisk for å beholde kunden. Jo lengre kunden har kommet i prosessen, jo vanskeligere blir kunden å beholde. Ved å identifisere en kunde som sannsynligvis vil slutte får en mulighet til å tilnærme seg kunden med tiltak som øker sannsynligheten for at kunden blir værende.

2. *Hva kjennetegner kundene som slutter?*

Ved å skaffe seg innsikt hos kundene som slutter oppnår en muligheten til å ta datadrevne beslutninger når man legger strategi for hvordan en skal bevare kundene. Innsikten kan brukes når man tilnærmer seg en kunde som har blitt identifisert som sannsynlig å slutte. Innsikten kan også være viktig i utviklingsarbeidet til banken. Ved å lære av kundene som har sluttet kan en tilpasse seg denne gruppen som igjen kan øke lojaliteten.

1.2. Oppbygning

Oppgaven er bygget opp ved at vi begynner å se på litt overordnet teori og tidligere arbeid i kapittel 2 før vi går videre til å presentere metoden vi skal bruke i kapittel 3.

I de påfølgende kapitlene vil vi gå gjennom stegene vi har beskrevet i metoden. I kapittel 4 prøver vi å skape en forståelse rundt forretningsproblemet vi skal løse, mens i kapittel 5 dykker vi dypere ned i dataene vi har tilgjengelig. Kapittel 6 tar for seg prosesseringen av dataene slik at de passer overens med modellene vi har valgt og presentert i kapittel 7.

I kapittel 8 skriver vi om eksperimentene vi har gjennomført og evaluerer modellene, mens vi i kapittel 9 analyserer forskjeller mellom de som har sagt opp boliglånet og de som ikke har det.

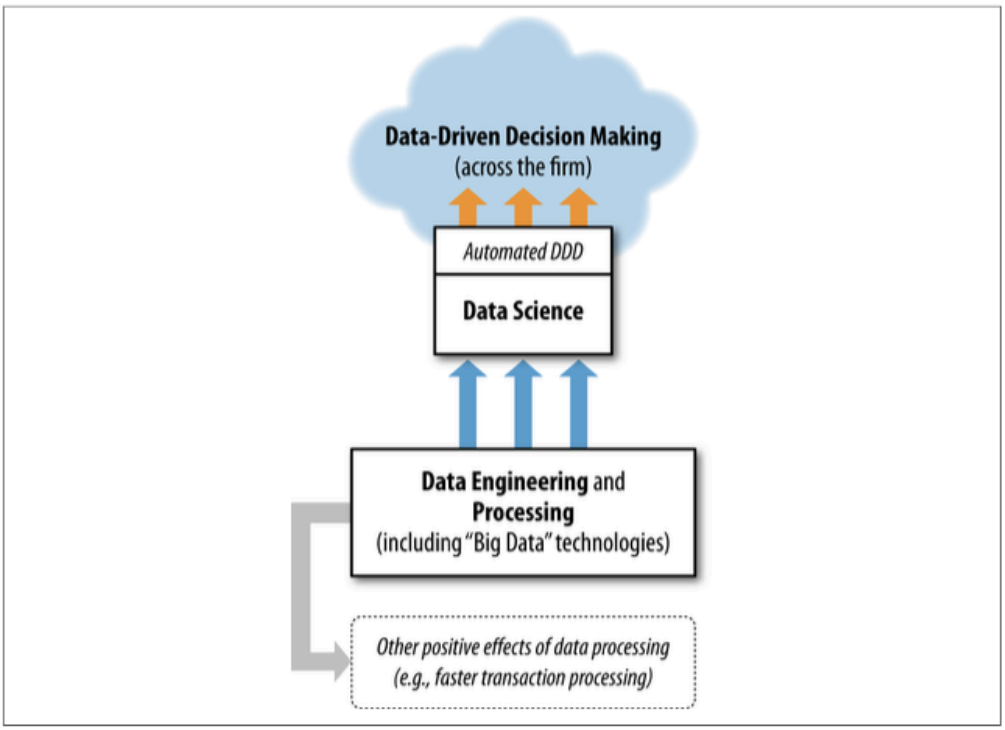
Kapittel 10 tar for seg diskusjon rundt funnene før vi i kapittel 11 oppsummerer og konkluderer.

2. Teori

I dette kapittelet vil vi presentere sentral teori for oppgaven og se på tidligere studier vi anser som relevant.

2.1. Datavitenskap

I følge Provost og Fawcett (2013) involverer datavitenskap prinsipper, prosesser og teknikker for å forstå fenomen gjennom automatisert analyse av data. I forretningssammenheng kan en si at hovedmålet med datavitenskap er å forbedre beslutningstakingen (Provost & Fawcett, 2013). Konseptet er illustrert i figur 1 hvor det begynner med dataprosessering før en går videre med datavitenskap som igjen blir brukt til å underbygge beslutninger i selskapet.



Figur 1 Datadreven beslutningstaking (Provost & Fawcett, 2013)

Provost og Fawcett (2013) skriver at datadreven beslutningstaking referer til å ta beslutninger basert på analysert data, i stedet for intuisjon. Data og evnen til å ta ut verdifull informasjon bør ansees som nøkkelressurser for bedrifter. Selskaper må ha den rette kompetansen for å hente ut verdifull informasjon fra dataene. En kan også ha den rette kompetansen, men dårlige data, noe som også vil gi lite verdi. Det er derfor viktig å anse data og kompetanse som komplementære

ressurser når en skal investere i datavitenskap. Det hjelper ikke investere i det beste teamet om en ikke har gode nok data å jobbe med (Provost & Fawcett, 2013).

Hvert datadrevne forretningsproblem er unikt, bestående av en egen kombinasjon av mål og rammer. Selv om forretningsproblemet er unikt for et gitt selskap, kan det deles inn i flere underoppgaver som er mer generelle. Et eksempel på et slikt forretningsproblem er kundeavgang i en bedrift. Hva som fører til kundeavgang er unikt for bedriften, mens det å predikere hvem som avslutter kundeforholdet sitt er en mer generell underoppgave. De forskjellige underoppgavene må løses hver for seg, og deretter settes sammen til en løsning på forretningsproblemet.

Selv om det er utviklet et stort antall algoritmer for å hente ut verdifull informasjon fra data gjennom årene er det kun en håndfull problemer disse tar for seg å løse. Et vanlig problem er å finne korrelasjoner mellom en bestemt attributt som beskriver et individ eller andre attributter. Den beste måten å løse dette problemet på er ved hjelp av algoritmer for klassifisering og regresjon (Provost & Fawcett, 2013). Forskjellen på de to metodene er at klassifisering predikerer om noe vil skje, mens regresjon predikerer hvor mye noe vil skje (Provost & Fawcett, 2013).

1. Klassifisering prøver å predikere, hvor hvert individ i en populasjon hører hjemme i et sett av klasser. Vanligvis er disse klassene gjensidig utelukkende. For et klassifiseringsproblem produserer man en modell, som gitt et nytt individ vil plassere dette individet i en klasse.
2. Regresjon prøver å estimere eller predikere, for hvert individ, en numerisk verdi for en attributt knyttet til individet. Modellen for dette genereres ved å se på andre, lignende individer i populasjonen og deres historiske data.
3. Gruppering prøver å gruppere individer i en populasjon basert på likheter, men uten en spesifikk hensikt. Metoden er nyttig i en eksplorativ fase hvor en ønsker å se hvilke naturlige grupper som eksisterer og kan fungere som en forløper for andre analytiske oppgaver.

(Provost & Fawcett, 2013).

2.2. Maskinlæring

Maskinlæring er design og studie av programvare som baserer seg på erfaring for å ta fremtidige beslutninger (Hackeling, 2014). I maskinlæringsverden er det i hovedsak to typer problem det skilles mellom. På den ene siden har man problemer hvor man har på forhånd vet hva man leter etter. Eksempelvis, et sykehus har store mengder data fra sykdomsforløp hos en rekke kreftpasienter, hvor krefttypen hos hver enkelt pasient er kjent. Basert på data om disse kundene ønsker sykehuset å identifisere ulike typer kreft hos nye pasienter. På den andre siden har man problemer hvor det man leter etter er ukjent. For eksempel, politiet sitter på tusenvis av overvåkningsbilder av et fåtall ulike personer fra et åsted, men de har ikke informasjon som knytter et bilde til en enkelt person. De ønsker derfor å gruppere bilder av samme person sammen, slik at de lettere kan knytte en gruppe bilder til én enkelt person.

Modeller som anvendes på problemer slik som det første eksempelet, hvor fasit for et datautvalg er kjent, er kjent som *overvåket læring*. Trening av en slik modell foregår ved at en modell blir presentert for en mengde data hvor fasit er kjent. Modellen vil i den grad den kan prøve å lære seg hvilke tilfeller som fører til en gitt output. På denne måten vil den kunne presenteres for ukjente data og generere en output for input som den aldri har sett før. Modeller som anvendes på problemer slik som det andre eksempelet, hvor fasit for et datautvalg *ikke* er kjent, betegnes som *uovervåket læring*. Trening av en slik modell foregår ved at modellen prøver å finne likheter og ulikheter mellom utvalg fra et datasett, og på den måten lage grupperinger i datasettet. Modellen vil da generere grupperinger hvor utvalg som er i samme gruppe er lik, og ulik utvalg som befinner seg i andre grupper.

Til tross for at det er ett spesifikt problem vi skal undersøke, vil vi ikke bare anvende modeller som er av typen overvåket læring, men også uovervåket læring. I utgangspunktet kan dette virke som en merkelig beslutning da vi i vårt problem sitter på fasit. Årsaken for anvendelsen av en slik modell er at den effektivt kan visualisere høydimensjonale data. Tatt i betraktning at slike modeller streber etter å gruppere dataene kan det gi oss interessant innsikt ved visualisering. Om modellen er i stand til å danne slike grupperinger er det en god indikator på at det er likheter mellom kunder i samme gruppering og ulikheter mellom kunder i ulike grupperinger.

2.3. Tidligere studier

Det er gjort lignende studier som prøver å predikere kundeavgang ved hjelp av forskjellige metoder og modeller tidligere. Selv om de i hovedsak gjør det samme som vi skal, er de *ikke* overførbare av flere grunner. Hver bank vil ha et unikt sett av data som varierer i form av struktur og innhold. I tillegg er markedet forskjellige på tvers av landegrenser. Hensikten med dette kapittelet er å danne et bilde av hvilke attributter som er brukt og hvilke funn som er gjort.

Generelt er fordelene som følger med kundebevaring og lojalitet avspeilet i årsakene til at en konkurrent er mer lønnsom enn en annen (Reichheld, 1993). Derfor støtter selskaper seg på to alternativer når en ønsker å beholde eller øke markedsandelen sin: kundebevaring og nye kunder. Selskaper investerer i forhold, ikke bare for å tiltrekke seg nye kunder, men for å bevare og forbedre forholdet med eksisterende kunder.

Van Del Poel (2004) tar i bruk proporsjonale hazard modeller[1] for å analysere kundeavgang. En kunde som har churnet blir definert slik: en kunde som har avsluttet alle kontoene sine hos en bank. Definisjonen avviker fra det inntrykket vi har fått fra Skandiabanken om at det er sjelden kunder avslutter forholdet, men heller flytter pengene sine. Fire typer attributter er som regel nevnt i kundebevaringslitteratur: Adferd, oppfattelse, demografi og makromiljø. Van Del Poel undersøker tre av disse predikatorene: adferd, demografi og makromiljøet. Dataene er hentet fra en database hvor kundene er observert over tid hos et europeisk selskap innen bank og forsikring. Van Del Poel omtaler banken som en sekundærbank. Dataene tar for seg nærmere 50 000 kunder hvor 47% har churnet over en periode på 77 år.

Innenfor hver gruppe finner vi mange prediktorer. Studien er veldig omfattende med mange interessante funn. Lengden på kundeforholdet påvirker sannsynligheten for å avslutte det samme kundeforholdet. I de første årene etter en har blitt kunde er sannsynligheten for å slutte høy, før den etter syv år stabiliserer seg. Sannsynligheten forholder seg stabil frem til år 20 før sannsynligheten for å slutte igjen øker.

Selv om ikke studien tar for seg gruppen med prediktorer for oppfattelse av banken har Van Del Poel (2004) gjort en grundig gjennomgang av tidligere forskning som viser at generelt sett

[1] Statistisk modell brukt for å modellere risikoen for at en hendelse skal inntreffe hos et individ

så har kundens oppfattelse av banken enten negativ eller ingen effekt på sannsynligheten for å slutte.

Dersom tiden mellom kjøp øker for en kunde øker også sannsynligheten for å slutte. Enkelt fortalt betyr dette at dersom tiden mellom hver gang en kunde kjøper et billån øker, øker også sannsynligheten for at kunden kjøper et billån hos en konkurrent. Dette understreker viktigheten av at banker fokuserer på mersalg hos sine eksisterende kunder.

Antall produkter en kunde besitter påvirker sannsynligheten for å slutte. En kunde med flere produkter har lavere sannsynlighet for å slutte enn en som bare har ett produkt. Nettbank, telefonbank og bankkort viser seg å ikke ha noen direkte påvirkning på sannsynligheten for å slutte.

Innenfor alder og kjønn er det også funnet forskjeller. Menn har større sannsynlighet for å slutte enn kvinner, mens alderen på virker sannsynligheten i form av at eldre har mindre sannsynlighet for å slutte. Hvor gammel en er når en blir kunde har også påvirkning. Kunder som har opprettet kundeforholdet i en mer voksen alder har en lavere sannsynlighet for å slutte, noe som kan forklares med at eldre tar mer overveide valg.

Høyere utdanning vil også redusere sannsynligheten for at kunden slutter. Det samme gjelder geografiske lokasjon sett i form av status, altså om man bor i et område med høyere status. Van Del Poel (2004) antyder at den høyere utdanningen fører til at man tar mer overveide valg, mens om man bor i et område med høy status så har man gjerne mer penger og derfor investerer i flere enn et produkt.

Keramti mfl. (2016) tar sikte på å utvikle en modell som predikerer kundeavgang ved hjelp av beslutningstrær. Metoden de tar i bruk er CRISP-DM eller ”Cross industry standard process for data mining”, en metode som går igjen i flere lignende studier og kilder rundt data mining. Mens Van Del Poel så på tre forskjellige typer attributter tar Keramti mfl. (2016) for seg to av typene, adferd og demografi. De ser på demografiske attributter som kjønn, alder, utdanningsnivå og yrke. Lengden på kundeforholdet, samt antall transaksjoner innenfor fem forskjellige tjenester. Dataene er hentet ut fra bankens database og strekker seg over to år. Ut ifra utdraget de har tatt, kan en se at de har valgt data som de anser som relevant for å løse forretningsproblemet. Studien tar i bruk relativt få attributter, men får gode resultater. Bakgrunn

for utvalget av attributter er ”data som er nødvendig for å løse forretningsproblemet”. Hvorfor de er relevant er ikke begrunnet.

En ting å merke seg ved datasettet er at det er veldig skjevt fordelt med over 98,5% som ikke har churnet. Beslutningstreet gir gode resultater i studien med over 90% riktig, men med et så skjevt datasett kan en stille spørsmål ved studiens validitet. Studien resulterer i fem beslutningsregler som klassifiserer churnere. Yrke blir fjernet fra datasettet etter at undersøkelser viser at attributten ikke har innflytelse på resultatet. Attributtene som går igjen i reglene er alder, lengde på kundeforhold og antall transaksjoner gjennom USSD-basert mobilbank. Utover disse er det en kombinasjon av de andre attributtene.

Zoric (2016) tar i bruk nevrale nettverk (vi kommer nærmere inn på nevrale nettverk i kapittel 7 og 8) i et forsøk på å predikere kundeavgang i en kroatisk bank. Hypotesen er at kunder som bruker flere bankprodukter eller tjenester har mindre sannsynlighet for å slutte. På bakgrunn av hypotesen kan vi si at Zoric undersøker en prediktor av typen adferd. For hver kunde er det seks attributter: kjønn, yrkesstatus, alder, månedlig inntekt, bruk av nettbank og om kunden bruker to eller flere banktjenester.

Funnene underbygger hypotesen. Kunder som bruker under to banktjenester har høyere sannsynlighet for å slutte enn kunder med to eller flere. Studenter utgjør en utfordrende gruppe da de ofte bruker færre enn to banktjenester og har da større sannsynlighet for å slutte. Samtidig er studenter en gruppe som er lite lønnsom i dag, men kan bli lønnsomme i framtiden. Zoric (2016) foreslår derfor å skreddersy produkter og tjenester til studenter for å få dem til å bli mer lojale.

Zoric (2016) sin studie kan vi etterprøve ved å se på gjennomsnittlig antall produkter for churn og ikke-churn i vårt datasett. En forskjell vil være at vi kun har boliglånskunder, noe som betyr at andelen studenter gjerne er veldig lav. Vi har heller ikke data på yrkesstatusen til kunden.

Clapp (2009) skriver om bankene i USA hvor det er opp til syv ganger større avgangsrate for kunder sammenlignet med Europa. Clapp mener videre at årsaken til dette er kampanjer for gratis brukskonto rundt hvert gatehjørne. Slik markedsføring har ført til at en stadig får flere kontoer og listen for å bytte til det beste tilbudet er blitt lavere. Selv om listen for å bytte er lav, bytter typisk ikke folk hovedbrukskontoen sin, men heller husholdningens brukskonto.

Forskjellen er at hovedbrukskontoen har debitkort og blir brukt gjennom nettbank for å betale regninger, noe som tilsier at kontoen blir brukt som primærkonto. Har du en kundes primærkonto kan du forvente lengre levetid på kunden. Clapp (2009) skriver videre at for å få en kunde til å ha primærkontoen sin hos banken din må du ta i bruk insentiver for å fremme bruken av debitkort og nettbank. Enten ved å gi belønning for slik bruk eller ved å legge på kostnader dersom kontoen bare er "en av de andre" brukskontoene.

De tidligere studiene er en god pekepinn for hvor vi kan begynne og hvilke attributter som muligens kan påvirke om en kunde kommer til å slutte eller ikke. Det er brukt forskjellige metoder som nevrale nettverk, beslutningstrær og hazard-modeller. Hva som passer best til våre data og forretningsproblem kan en ikke si ut ifra tidligere studier, men ved hjelp av prøving og feiling. Hvor overførbar funnene fra de tidligere studiene er for våre data er vanskelig å anslå. Det er flere momenter som taler mot overførbarheten. Van del Poel (2004) definerte churn som kunder som aktivt har gått frem og avsluttet forholdet, samtidig som han hadde data fra en finansiell institusjon ansett som en typisk sekundærbank. Keramati mfl. (2016) bruker data som vi stort sett har tilgjengelig og oppnår gode resultater. Dataene har en veldig skjev fordeling av kunder som har sluttet og ikke sluttet. Hovedproblemet for begge studiene er hvilke kunder som blir analysert. Vi ser på kunder som har boliglån, mens de tidligere studiene skiller ikke mellom hvilken type kunde som blir sett på. Dette gjør funnene til Keramati mfl. (2016), hvor studentene utgjorde en risikogruppe da de hadde få produkter i banken, mindre aktuell for vår studie. Årsaken er at det er rimelig å anta at relativt få studenter har boliglån. Vi har heller ikke mulighet til å klassifisere ut ifra yrkesstatus da dette ikke er data vi innehar.

Basert på de tidligere studiene kan vi utforme følgende hypoteser:

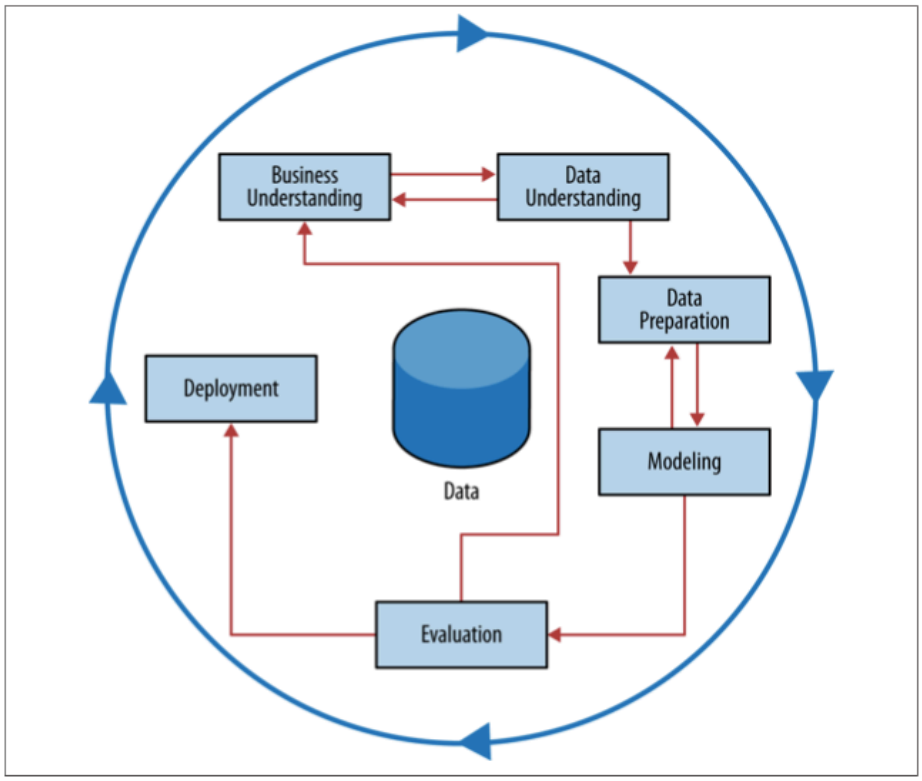
- Flere produkter reduserer risikoen for å slutte.
- Eldre har lavere risiko for å slutte.
- Kunder som har vært kunde lenge har lavere risiko for å slutte.
- Menn har større sannsynlighet for å slutte.
- Høyere inntekt reduserer risikoen for å slutte.

Det er også mulig å lage hypoteser innenfor yrkesstatus og hvor en bor, men da vi ikke sitter på data til å undersøke dette, har det lite for seg. En kan også se at Van del Poel (2004) skriver at

bruken av spesifikke produkter ikke har noe og si, mens Keramati mfl. (2016) finner at antall transaksjoner på mobilbank er en utslagsgivende attributt. Det kan derfor være interessant å se hvilke av hypotesene som stemmer for våre data. Sammenlignet med de tidligere studiene har vi en god del flere attributter. Hvordan størrelsen på boliglånet påvirker sannsynligheten for å si opp sitt boliglån har etter vår kjennskap *ikke* tidligere vært undersøkt.

3. Metode

Vi tar utgangspunkt i stegene definert i Cross Industry Standard Process for Data Mining, gjengitt i boken Data Science for Business(Provost & Fawcett, 2013). Metoden tar sikte på å strukturere problemet og oppnå konsistens, repeterbarhet og objektivitet. CRISP-DM, vist i figur 2, er delt inn i stegene, forretningsforståelse, dataforståelse, forberedelse av data, modellering, evaluering og til slutt ta i bruk løsningen. Som en kan se på figuren under så er dette en iterativ prosess hvor en som oftest ikke får det til på første forsøk. Man går da flere runder for å hele tiden forbedre resultatet med kunnskapen man tilegner seg underveis.



Figur 2 CRISP-DM, (Provost & Fawcett, 2013)

3.1. Forretningsforståelse

Til å begynne med er det viktig å forstå forretningsproblemet en skal løse. For å ta i bruk data mining i et forretningsproblem må det oversettes og brytes ned til et datavitenskapelig problem. Som regel er dette en jobb for en forretningsanalytiker i samarbeid med utviklere. Nøkkelen til suksess er ofte en god problemformulering som oversetter forretningsproblemet til et eller flere datavitenskapelige problem som involverer modeller for klassifisering, regresjon eller sannsynlighetsestimering (Provost & Fawcett, 2013).

Et av det viktigste konseptene ved datavitenskap er bruksscenario. Hva ønsker en å oppnå? Hvordan ønsker vi å oppnå det? Hvilke deler av dette bruksscenarioet inneholder potensielle data mining modeller?

3.2. Dataforståelse

Når en ønsker å løse et forretningsproblem vil dataene bestå av rådata som er tilgjengelig for bedriften. Det er viktig å forstå styrkene og begrensningene til dataene ettersom det sjelden er en direkte sammenheng mellom dataene og problemet. Historiske data er ofte samlet inn av hensyn som ikke er relatert til problemet en ønsker å løse. En kan ikke forutse problemene som oppstår i fremtiden, så det er derfor vanskelig å samle inn data for dette formålet. En kundedatabase og transaksjonsdatabase inneholder forskjellige data og kan bestå av forskjellige populasjoner. En kan spekulere i hvilke data en vil trenge i fremtiden, ettersom lagringsplass blir billigere er det blitt mer og mer vanlig å samle inn så mye data som mulig.

En kritisk del av dataforståelsesfasen er å estimere kostnader og verdier til forskjellige datakilder og vurdere om det er grunnlag for videre investering. Etter en har fått samlet inn alle datasettene, kreves det vanligvis stor innsats for å sortere dem. Som regel inneholder kunde- og produktregister mange attributter og mye støy, eksempelvis i form av manglende verdier. Å rydde opp i slike registre slik at det kun er én rad per kunde er i seg selv et komplisert problem.

Etterhvert som dataforståelsen øker, kan veien videre endre seg. For eksempel ved å endre maskinlæringsalgoritme til en som er mer tilpasset dataene og forretningsproblemet enn det som initielt var planlagt. Kan man tydelig identifisere et mål (sluttet/aktiv), så kan man ta i bruk metoder som klassifiserer, mens hvis man ikke kan sette et entydig mål må man ta i bruk grupperingsmetoder.

For å oppnå forståelse for dataene må man grave under overflaten og oppdage sammenhengene mellom forretningsproblemet og den tilgjengelige dataen, og deretter knytte de til en eller flere data mining oppgaver. Det er ikke uvanlig at et forretningsproblem inneholder flere ulike slike oppgaver av forskjellige typer, som klassifisering og gruppering (Provost & Fawcett, 2013).

3.3. Forberedelse av data

Analyseverktøyene og maskinlæringsalgoritmene er kraftige verktøy, men kommer som regel med begrensninger på dataene de tar i bruk. Den naturlige strukturen på dataene må gjerne endres for å passe et format som algoritmene ønsker. Det er en tidkrevende prosess hvor man ønsker å beholde mest mulig informasjon i dataene. For å bevare informasjonen i dataene kreves god dataforståelse og innsikt, noe som gjør det naturlig å kombinere forberedelsen av dataene med steget for dataforståelse.

Typiske eksempler på forberedelse av data er å konvertere dataene til et tabellformat hvor hver rad representerer en enhet, mens kolonnene holder attributtene tilhørende enheten. Videre fjerner man eller beregner manglende verdier. En kan også analysere hver enkelt attributt og se om den inneholder nyttig informasjon. Dersom attributten inneholder mye støy eller lite variasjon kan det være hensiktsmessig å fjerne attributten.

Formatet på dataene er ofte avhengig av hvilke modell man tar i bruk. Enkelte modeller tar kun imot numeriske attributter, mens andre kan ta imot flere former for data. Numeriske attributter må som regel normaliseres slik at de blir sammenlignbare.

3.4. Modellering

Modellering er steget hvor en implementerer og anvender modeller på dataene. Steget må sees i tett sammenheng med evalueringen da en gjerne vurderer presisjonen på modellene underveis og jobber med å optimalisere presisjonen. Forskjellige modeller passer til forskjellige typer data, så det er også hensiktsmessig å prøve flere modeller for å finne den som passer best til dataene. Resultatet av modelleringen er som regel en modell eller et mønster som fanger regulariteter i dataene (Provost & Fawcett, 2013).

3.5. Evaluering

Hensikten med evaluringssteget er å vurdere påliteligheten og validiteten til modellen. En ønsker å kunne si med sikkerhet at funnene er sanne regulariteter og ikke bare særegenheter eller anomalier i gruppen som blir undersøkt. Modellen skal også ha tilnærmet lik presisjon over gjentatte forsøk over tid. Like viktig er det å evaluere at man har oppnådd det man ønsket. Dette gjør man ved å se tilbake på forretningsmålene. Hovedmålet med data mining er som nevnt tidligere, å støtte opp under beslutningstaking. Data mining er kun en del av en større

løsning. Selv om en evaluering av data miningen gir gode resultater er det ikke gitt at den egner seg som en del av den større løsningen. En kan for eksempel ha en løsning som predikerer de som har sluttet med stor nøyaktighet, men samtidig gir for mange falske alarmer til å være økonomisk lønnsom.

Evaluering av data mining-resultatene involverer både kvantitative og kvalitative vurderinger. Ulike interessenter har interesse i beslutningstakingen som blir støtte opp av modellen. Som regel vil interessentene godkjenne modellen før den tas i bruk. Hva som skal til for at en modell blir godkjent varierer fra situasjon til situasjon, men en kan som regel si at modellen skal gi flere fordeler enn ulemper.

Evalueringen fra interessenter kan by på en utfordring da dette gjerne ikke er tekniske personer. Det er derfor viktig å lage et forståelig evalueringsrammeverk som forklarer modellen. En annen utfordring med evalueringen er at en ikke har tilgang til systemet modellen skal tas i bruk i. En må derfor ofte ta i bruk et annet miljø for å evaluere modellen. Her kommer evalueringsrammeverket inn igjen; sammen med en tilnærmet lik kopi av miljøet løsningen skal rulles ut i (Provost & Fawcett, 2013).

3.6. Utrulling

Utrulling er steget hvor løsningen tas i bruk i virkelige situasjoner for å realisere avkastning på investeringen. Den enkleste formen for utrulling er å implementere en prediksjonsmodell i et IT-system eller forretningsprosess. I økende grad er det nå blitt mer vanlig å implementere data mining-teknikkene i stedet for modellen som blir produsert. Trenden fører til at man har et system som automatisk bygger modeller underveis basert på de nyeste dataene. Ved å implementere et slikt system i stedet for en ferdig modell er det lettere å henge med på endringer i markedet. Har man flere modeller er det fare for at verden endrer seg fortere enn man klarer å vedlikeholde modellene (Provost & Fawcett, 2013).

4. Forretningsforståelse

I dette kapittelet ønsker vi å skape et bilde av situasjonen til Skandiabanken og hvordan denne studien passer inn i deres forretningsmål.

Skandiabanken startet i 2000, som den første rene nettbanken i Norge. Frem til 2015 var banken en filial av Skandiabanken AB, før den ble omdannet til et eget selskap og registrert på Oslo Børs. Banken konsentrer seg om privatmarkedet og hadde i 2015 2,5 prosent av lånemarkedet for norske husholdninger fordelt på 380 000 kunder med saldo på konto. Banken er hel-digital, noe som betyr at den ikke har filialer og kun tilbyr tjenester og produkter gjennom nettbank (Skandiabanken, 2015).

”Folk flest har sterkest forhold til den banken der de har boliglånet sitt. Vi satser derfor på å vokse i boliglånsmarkedet, både blant nye og eksisterende kunder” (Skandiabanken, s.12, 2015).

I EPSIs årlige bankundersøkelse for ”De mest fornøyde kundene” i Norge har banken vunnet hvert år siden 2005. I følge Norsk Kundebarometers årlige undersøkelse har banken vært best på kundetilfredshet i den norske bank- og finanssektoren siden 2002. Undersøkelsen fra 2015 viser også at bankens kunder er mer lojale enn andre bankkunder i Norge (Skandiabanken, 2015).

Forbruker- og finanstrender er en undersøkelse som Kantar TNS gjennomfører årlig i samarbeid med Finans Norge innenfor markedene bank, skadeforsikring og livsforsikring. Undersøkelsen omfatter ca. 2200 intervju gjennomført blant befolkningen over 18 år. Resultatene fra 2017 viser at nordmenn er flinke til å utnytte konkurransen i markedet. Ni prosent av respondentene har byttet boliglånsbank i løpet av det siste året, mens 15 prosent forhandlet seg til lavere rente hos sin nåværende bank (Finans Norge/Kantar TNS, 2017).

I gjennomsnitt har en nordmann 1,9 banker i bruk. Og forholdene som er viktigst når en skal bytte bank er en god nettbank og konkurransedyktige betingelser på lån over tid. Undersøkelsen viser også at respondentene opplever at det blitt lettere å bytte bank enn tidligere. En av tre

svarer også at det kan være aktuelt å bytte bank i løpet av de tre neste årene (Finans Norge/Kantar TNS, 2017).

Å bytte bank gjør man ved å registrere seg som kunde hos en annen bank, noe som kan gjøres enkelt på nettet og tar under fem minutter. Flytting av pengene og avtaler kan man enten gjøre selv eller be den nye banken ordne opp. For å flytte boliglånet må man selvfølgelig få innvilget lån hos den nye banken. Den store utfordringen for bankene er at kunden ikke behøver å ta kontakt med banken en flytter fra. En trenger heller ikke avslutte kontoene eller kundeforholdet hos den gamle banken, men bli værende som en kunde uten aktivitet. Fremgangsmåten gjør at det nesten blir umulig for banken å iverksette tiltak for å bevare kunden i god nok tid. En registrerer først at kunden har sluttet når den har inngått en ny avtale med en annen bank.

For å vokse i boliglånsmarkedet er det vel så viktig å beholde de kundene de har i dag, samtidig som de tiltrekker seg nye. Forbedring av kundebevaringen kan deles inn i tre delproblemer:

- Identifisere kunder som skal slutte.
- Innsikt hos kunder som slutter.
- Tiltak for å bevare kunder.

Å identifisere kunder som skal slutte kan bli løst ved hjelp av data mining. Utfordringen er å identifisere kunden tidlig nok, slik at man får gjort tiltak for å motvirke dette. Er banken for sent ute, kan kunden ha inngått en avtale med en ny bank og være lite mottagelig for forsøk på å få kunden tilbake.

Det andre delproblemet, innsikt hos kundene som slutter, kan også løses ved hjelp av data mining. Innsikten kan bli brukt til å underbygge tiltakene for å bevare kundene både gjennom direkte kontakt med enkelt kunder, men også mer indirekte ved at en kan tilpasse banken etter det vi lærer om kundene som slutter.

Det siste delproblemet, tiltak for å bevare kunden, skal vi ikke gå så mye inn på da dette er noe banken har faste rutiner for, men med løsninger på de to første delproblemene kan det være hensiktsmessig for banken å tilpasse sin strategi for kundebevaring etter løsningene på disse problemene.

Med utgangspunkt i delproblemene kan vi utforme to bruksscenarier:

1. Direkte
    a. Identifisere kunde som skal slutte.
    b. Opprette tiltak basert på innsikt i grupper denne kunden tilhører.
    c. Iverksette tiltak.
2. Indirekte
    a. Oppnå innsikt i gruppen som har sluttet.
    b. Tilpasse produkter og tjenester etter trekk ved denne gruppen.

5. Dataforståelse

I dette kapittelet vil vi studere dataene vi har tilgjengelig fra Skandiabanken for å gjøre oss kjent med hva vi har å jobbe med. Dataene er hentet ut fra databaser og lagret i tabulatorseparerte tekstfiler.

5.1. Transaksjoner

Transaksjonsfilen inneholder 2672249 rader og syv attributter (kolonner). Tabell 1 viser en kort beskrivelse av attributtene. Hver rad i tabellen beskriver det totale beløpet og antallet for en type transaksjon i en gitt måned, år og valuta for en kunde. En kunde kan altså forekomme på flere rader i tabellen. Hver rad er en unik kombinasjon med tanke på innholdet i de forskjellige attributtene. Filen inneholder ikke informasjon om hvor transaksjonene er gjennomført, en attributt som kunne vært interessant å se nærmere på.

Metodene og modellene vi skal bruke senere i oppgaven krever at informasjon om en enkelt kunde er samlet på én rad. Dette gjør at formatet på dataene i filen må endres slik at den passer med modelleringen og analysen senere i oppgaven.

Attributt	**Beskrivelse**
KundeId	Unik nøkkel som identifiserer en kunde
År	År 2010-2014
Mnd	Måned 1-12
Type	Type transaksjon. Eks. Avtalegiro og VISA Varekjøp
Valuta	Valuta for transaksjoner
Beløp	Størrelse på beløp, både positive og negative verdier
Antall_trans	Antall transaksjoner for en unik type, kunde, mnd og år

Tabell 1 – Transaksjoner

5.2. Kundeinfo

Tabellen inneholder grunnleggende informasjon om en kunde. Hver rad inneholder en kunde med tilhørende informasjon, noe som gir 5354 rader. Tabellen inneholder ti attributter vist i tabell 2. PostadresseLand har stort sett NULL, med noen unntak. Det er ikke samsvar mellom

Postnummer og Poststed og PostadresseLand. Vi har blant annet oppdaget kunder med postnummer og poststed i Norge, mens PostadresseLand er et annet land enn Norge. En feil i postnummer-attributten er at postnummer som begynner med 0 blir tolket som et tresifret tall. Kundestatus er som regel enten aktiv eller avsluttet, med noen få unntak som f.eks. inkasso. Hvorfor et kundeforhold er avsluttet er vanskelig å si på bakgrunn av den informasjonen vi sitter på. Årsaken kan være på initiativ fra banken, kunden eller grunnet dødsfall. Attributtene som viser dato for første innlogging på enten mobil eller nettbank inneholder NULL dersom kunden aldri har logget inn.

Attributt	**Beskrivelse**
KundeId	Unik nøkkel som identifiserer kunde
Postnummer	Postnummer
Poststed	Poststed
Kundestatus	Status på kundeforholdet. Aktiv, avsluttet
PostadresseLand	Land
Kjønn	Kjønnet til kunde
Fødselsår	Fødselsår til kunde
OpprettetDato	Dato for kundeopprettelse
FørsteInnloggingDato	Dato for første gang kunde logget inn i nettbank
FørstInnloggetMobil	Dato for første gang kunde logget inn i nettbank

Tabell 2 - Kundeinfo

5.3. Innlogging

Tabellen viser innloggingsdata hvor hver rad står for antall innlogginger av en kunde i en måned på de forskjellige kanalene og ved hjelp av de forskjellige innloggingstypene. Tabellen har seks attributter, vist i tabell 3. Dersom en kunde har logget inn på ulike kanaler eller ved hjelp av forskjellige innloggingstyper vil dette forekomme på forskjellige rader. Hver rad er altså en unik kombinasjon med tanke på innholdet i de forskjellige attributtene. Strukturen gjør at filen blir på 384338 rader. På samme måte som med transaksjoner er utfordringen med filen å samle informasjon om en kunde på en rad.

Attributt	**Beskrivelse**
År	År 2010-2014
Mnd	Mnd 1-12

KundeId	Unik nøkkel som identifiserer en kunde
Kanalnavn	Kanal logget inn på
Innloggingstype	Type autentisering
Antall_Logid	Antall forekomster

Tabell 3 - Innlogging

5.4. Produktbeholdning

Tabellen, vist i vedlegg 13.1, inneholder informasjon om produktbeholdningen for hver enkelt kunde ved utgangen av et år. Tidsperioden går fra 2010 til 2014, så hver kunde forekommer fem ganger. Én rad for hvert år. Resultatet er en fil med 25719 rader, fem ganger antall kunder. Tabellen inneholder ekstremt mye informasjon med 67 attributter, gjerne på et unødvendig format. En kunne for eksempel slått sammen positiv og negativ saldo til saldo, som kunne holdt både positive og negative verdier. I tillegg inneholder flere av attributtene lite variasjon og ujevn fordeling, hvor for eksempel alle radene har samme verdi. Attributtene blir dermed overflødig ettersom en ikke kan hente ut noe nyttig informasjon. Som eksempel kan vi trekke frem NegativSaldoBSU. Det er ingen kunder som har negativ saldo på BSU, så attributten gir oss ingen informasjon.

Formatet på tabellen passer godt overens med metodene og modellene vi ønsker å bruke senere i oppgaven. Ved å hente ut et enkelt år, har vi all dataene for en kunde i det året samlet på en linje. Når det gjelder antall attributter så er dette veldig høyt. Ved å se på hver enkelt attributt, og om denne inneholder informasjon som kan være nyttig eller ikke, vil vi kunne redusere antall attributter ved å fjerne de som inneholder lite informasjon.

6. Databehandling

I dette kapittelet vil vi beskrive hva vi har gjort med dataene vi fikk fra Skandiabanken for å klargjøre de til modellene og analysen vi skal gjøre senere.

6.1. Churn

Det første vi gjør er å finne ut hvem som har churnet. For å finne dette tar vi i bruk tabellen produktbeholdning som inneholder år 2010-2014 for hver kunde og de respektive produktene de hadde i disse årene. Vi har definert at de som har churnet er kunder som har følgende egenskaper:

AktivAntallBoliglån = 0 & InAktivAntallBoliglån > 0.

Ettersom vi har fem år for hver kunde betyr det at predikatet kan være sann for flere av årene. Løsningen er å ta det tidligste året hvor predikatet er sann for hver kunde. Resultatet er en tabell med attributtenene, KundeId og ChurnÅr, som inneholder alle kunder som har churnet identifisert med Id'en til kunden og årstallet han/hun churnet.

6.2. Valg av år

Det at dataene vi har tilgang til har en begrenset tidshorisont byr på problemer. Ettersom vi kun kan identifisere churnere på årsbasis kan vi ikke fastslå når på året de har churnet. For de som har churnet i 2010 forårsaker det et problem. De kan ha churnet i januar 2010, noe som betyr at vi ikke har data på tiden før de churnet. Vi har derfor valgt å ekskludere alle som har churnet i 2010.

For å få likt datagrunnlag for alle kundene har vi valgt å trekke ut ett år for hver kunde som vi bruker i analysen og predikeringen. For de som har churnet blir dette ChurnÅr fratrukket 1, mens for de som ikke har churnet trekker vi ut år 2013. Grunnen til at vi tar 2013 og ikke 2014, som er det siste året vi har data på er at vi ikke vet om en kunde har churnet i 2015. Hadde vi tatt ut 2014 for de som ikke har churnet kunne vi potensielt trukket ut en kunde som churnet i 2015 og ansett denne kunden som ikke churnet.

Attributten VelgÅr blir lagt til i tabellen og får verdi basert på ChurnÅr-attributten. Dersom ChurnÅr er større en 0, blir verdien i VelgÅr lik verdien til ChurnÅr fratrukket 1, ellers blir den 2013. Et problem med denne utvelgelsen er at vi ikke sammenligner kunder i samme tidsperiode. Det kan være forskjeller i dataene basert på tidsperioden det er hentet fra. Nye produkter kan bli lagt til et år, andre kan forsvinne og lignende. Ettersom de som ikke har churnet har data fra 2013 og de som har churnet har data fra 2010-2013 kan det være ulikheter som skyldes endringer fra banken sin side. For å kompensere for forskjellen i tid, undersøker vi om attributtene har forekomster i de ulike årene. Dersom en attributt kun har forekomster fra én tidsperiode, vil denne bli slettet.

6.3. Kundeinfo

Nå som vi har tabellen med de som har churnet kan vi begynne å utvide den med nye attributter. Utvidelsen gjør vi ved å slå sammen HarChurnet med Kundeinfo. Ettersom HarChurnet-tabellen kun inneholder de kundene som har churnet vil det være færre kunder i denne tabellen enn Kundeinfo. Det er derfor viktig at vi slår de sammen på en måte som beholder alle kundene fra Kundeinfo. Vi ønsker å bevare Kundeinfo slik den er, men legge til attributten ChurnÅr. Dermed slår vi sammen tabellene basert på KundeId fra Kundeinfo. Dersom KundeId eksisterer i HarChurnet vil den få verdien fra HarChurnet, men om den ikke finnes vil den få verdien NULL. NULL verdien bytter vi med 0 slik at kunder som har churnet får årstallet de churnet, mens de som ikke har churnet får verdien 0.

I tabellen Kundeinfo har vi en attributt som heter Kundestatus. Attributten er i de fleste tilfeller Aktiv, men i noen tilfeller Avsluttet eller Inkasso. Siden vi ikke vet årsaken til hvorfor noen av kundene har status som Avsluttet eller Inkasso velger vi å ekskludere disse. Resultatet gjør at vi da kun sitter igjen med kunder som har status som aktiv i banken. Slik vi har oppfattet det fra Skandiabanken er det uvanlig å avslutte kundeforholdet, men heller flytte pengene og tilbakebetale lån. Spesielle forhold som dødsfall kan for eksempel være en årsak til et avsluttet kundeforhold.

Det er tre attributter i tabellen som sier noe om geografisk tilhørighet. PostadresseLand inneholder stort sett NULL-verdier, og i de tilfellene attributten har en verdi er det ikke sammenheng mellom Postnummer, Poststed og PostadresseLand. Vi har derfor valgt å fjerne attributten PostadresseLand. Når det gjelder Postnummer og Poststed er dette attributter som

inneholder for mange unike verdier for at det skal være nyttig for analyse og predikering med tanke på at vi kun har et utvalg på ca. 5000. Modellene forstår ikke hva disse attributtene står for eller sammenhengen mellom dem, noe som gjør de til et forstyrrende element. En mulighet er å dele de forskjellige postnumrene inne i større geografiske områder som f.eks. fylker, men grunnet begrenset med tid og en feil hvor postnumre som begynner på 0 blir representert som et tresifret postnummer har vi valgt å fjerne disse attributtene.

Tabellen inneholder også tre attributter som representerer en dato i form av en tekststreng: OpprettetDato, ForsteInnloggingDato og ForstInnloggetMobil. ForstInnloggetMobil har mange NULL-verdier da det er mange som ikke har logget inn, mens de som har logget inn har en dato. Attributten omgjøres til 1 for har logget inn og 0 for har *ikke* logget inn. Attributtene OpprettetDato og ForsteinnloggingDato blir gjort om til å kun inneholde årstallet. I tillegg legges det til ArSomKunde og ArSidenForsteInnlogging som indikerer antall år mellom den originale attributten og VelgÅr. Videre legger vi til attributten AlderIValgtAr som indikerer alderen til en kunde i det året vi har hentet ut. Attributten finner vi ved å trekke fra Fødselsår fra VelgÅr.

Det er gjerne typisk at man logger inn i banken ganske raskt etter man er blitt kunde. Vi sjekker derfor korrelasjonen mellom ArSomKunde og ArSidenForsteInnlogging som viser seg å være veldig høy med 0,9653. Vi velger derfor å slette ArSidenForsteInnlogging.

For å oversette attributtene til et format de fleste modeller forstår har vi valgt å endre de til numeriske verdier som kun inneholder årstallet. Kunder som ikke har logget inn i nettbank (ForsteInnloggingDato) eller mobilbank (ForstInnloggetMobil) har fått verdien 0 i disse attributtene.

6.4. Innlogging

I tabellen som har innloggingsdata (illustrert ved tabell 3) er vi interessert i antall innlogginger på de forskjellige kanalene og med de forskjellige innloggingstypene. Slik tabellen er i dag inneholder en rad en kundes innlogginger i en gitt måned i et gitt år på en gitt kanal og med en gitt type. Det vil si at for hver unike kombinasjon av disse vil det forekomme en egen rad. Vi ønsker å samle denne informasjonen på én rad for det året vi skal hente ut basert på attributten VelgÅr. Vi lager derfor en egen attributt for hver verdi i både Kanalnavn og Innloggingstype.

Etterpå aggregerer vi opp månedene for det valgte året slik at attributtene inneholder antall forekomster for hele året.

6.5. Transaksjoner

Transaksjonstabellen (illustrert ved tabell 1) har et forholdsvis likt format som innlogging ved at en rad inneholder kundens transaksjoner i beløp og antall for en type, i en gitt måned, i et gitt år. Attributten Valuta inneholder kun verdien NOK, så denne fjerner vi. For hver unike kombinasjon vil det forekomme en egen rad, noe som betyr at informasjon om en enkelt kunde er spredd over flere rader. Vi ønsker å hente ut antall transaksjoner for de forskjellige typene. For å samle dataene på én rad, lager vi attributter av de forskjellige typene. Videre summerer vi alle transaksjonene for det valgte året slik at en rad inneholder antall transaksjoner av hver type for dette året.

Under prosessen gjorde vi flere interessante funn om dataene. Lønn, Barnetrygd og Pensjon har stort sett 12 forekomster i året for alle kundene. Det er derfor mer interessant å se på beløp for disse attributtene. Vi byttet derfor ut antallet med sum av transaksjonene for et år i disse attributtene.

Varekjøp Verified by Visa forekommer kun i 2010, noe som byr på problemer i og med at det er kun de kundene som er hentet fra år 2010 som har verdi for denne attributten. Kundene som ikke har churnet er blitt plukket fra år 2013, og har dermed ikke verdi i attributten. For klassifiseringen betyr dette at modellen kan plassere kunder som har verdi i Varekjøp Verified by Visa i klassen som har churnet. Ettersom vi ikke ønsker å ha attributter som er påvirket av året de er hentet fra, velger vi å fjerne den.

6.6. Produktbeholdning

Den siste tabellen er produktbeholdning som inneholder 67 attributter. Hver kunde har fem rader, én for hvert år. Som tidligere trekker vi ut ett år for hver kunde, basert på VelgÅr. Utvelgelsen byr på et problem da 31 av kundene ikke fikk boliglån før 2014. Disse har da verdien 0 i AktivAntallBoliglån, InAktivAntallBoliglånn og NegativSaldoBoliglån, og er teknisk sett ikke en boliglånskunde i det utvalget vi har tatt. Vi sletter derfor disse kundene.

Tabellen inneholder en del overflødige attributter. For hvert produkt er det delt inn i positivt antall, negativt antall, positiv saldo og negativ saldo. Hvis man tar BSU som eksempel så er det ikke lov å ha mer enn én BSU, og i og med at det er en sparekonto for boligkjøp kan den ikke ha negativt beløp. Den eneste informasjonen som er interessant for produktet er positiv saldo.

Det er flere slike attributter. Et annet eksempel er de forskjellige låneproduktene. Det er ingen som har positiv saldo på lånet, så denne attributten er overflødig. Vi har redusert dimensjonen på tabellen ved å slå sammen negativ og positiv saldo til saldo. Saldo kan da ha både positiv og negativ verdi. For de produktene som man enten har ett eller ingen av har vi fjernet attributten som inneholder antall, da saldo vil gi den samme informasjonen den samme informasjonen. Er saldoen lik null kan vi anta at kunden ikke har produktet. Hvis saldoen er ulik null kan vi videre anta at en kunde har produktet.

I enkelte tilfeller har vi beholdt attributten med antall, da denne har flere unike verdier enn null og en. Et eksempel på dette er PositivAntallAIE, som sier hvor mange brukskontoer med positiv saldo en kunde har. Etter prosesseringen sitter vi igjen med 21 av 67 attributter.

I tabellen er det mange verdier i form av beløp ned til øre, og som resultat gjør at slike attributter blir et sett av unike verdier. For å redusere antall unike verdier runder vi av til nærmeste 10 000 noe som betyr at vi deler attributtene inn i bøtter med intervaller på 10 000.

6.7. Reduksjon av dimensjoner

Etter alle tabellene er prosessert hver for seg og slått sammen sitter vi igjen med én enkelt tabell med 97 attributter. For å redusere dimensjonene på tabellen vil vi se på hver enkelt attributt og hvordan verdiene er fordelt. Lavt standardavvik og lav varians betyr lite informasjon. Vi begynner med en grov filtrering hvor vi ser på antall forekomster av verdier i en attributt. Dersom en attributt har en dominerende verdi hvor over 99% av tilfellene har samme verdi, sletter vi attributten. Vi sitter nå igjen med 4789 rader med individuelle kunder, noe som betyr at 1% utgjør 47,89.

Gjennom filtreringen fikk vi redusert tabellen med 22 attributter, noe som betyr at vi sitter igjen med 75 attributter. Hvilke attributter som ble slettet er vist i vedlegg 13.3. Det er fortsatt en god

del og en finere filtrering vil bli gjort ved hjelp av beslutningstrær som kan si oss hvilke attributter som påvirker resultatet mest.

Etter prosesseringen sitter vi som sagt igjen med 4789 kunder. Av disse har 3588 ikke churnet, mens 1201 har churnet. Dette gir oss en skjev normalfordeling. Vi lager derfor et nytt sett hvor vi tar de 1201 som har churnet og et tilfedlig utvalg på 1201 for de som ikke har churnet. Da ender vi opp med en symmetrisk fordeling.

6.8. Prosessert datasett

Etter prosesseringen sitter vi igjen med to datasett. Et sett med 2402 kunder hvor fordelingen mellom churn og ikke churn er lik og et sett med 4789 kunder hvor fordelingen er 25% churnere og 75% ikke-churnere. Begge settene har 75 attributter (illustrert i vedlegg 13.2) inkludert identifikatorattributter som KundeId og Ar. FodselsAr og OpprettetDato er beholdt i settet for analytiske formål, selv om de er erstattet med attributter med lignende informasjon. De ovennevnte attributtene vil bli ekskludert når settet anvendes i modellering og er markert med * i vedlegg 13.2.

Alle kundene i datasettet regnes som boliglånskunder da de har hatt boliglån i løpet av de fem årene vi undersøker. Videre har alle kundene status som aktiv, noe som vil si at de fortsatt er kunde er banken. Når vi skriver om en kunde som har *churnet* eller *sluttet* så er dette en kunde som har hatt boliglån, men enten har betalt det ned eller flyttet lånet til en annen bank.

7. Modellering

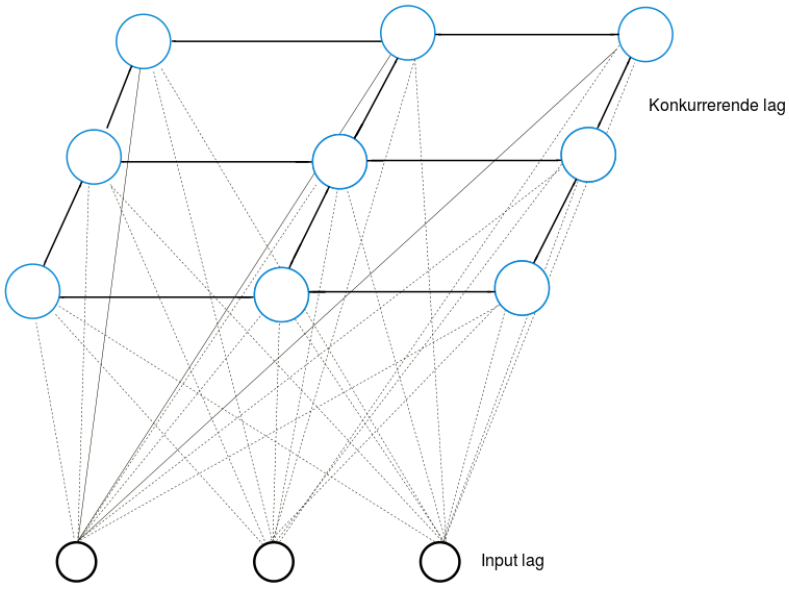
I dette kapitelet vil vi presentere de forskjellige modellene vi skal ta i bruk. Vi ønsker å først anvende en modell for å enkelt visualisere datasettet og potensielle grupperinger i datasettet. Videre skal vi se på ulike algoritmer som i ulik grad vil være i stand til å predikere kundeavgang basert på dataene vi sitter på. Blant modellene vi skal se på er nevrale nettverk. Et nevralt nettverk er modeller som er sterkt inspirert av strukturen til hjernen (Kristensen, 1997).

7.1. Kohonen nettverk

Dataene vi sitter på er representert i et høydimensjonalt rom, nærmere bestemt 70 dimensjoner etter preprosessering. For å effektivt visualisere et slikt datasett må antall dimensjoner reduseres ned til maks tre dimensjoner. For å se om det er noen synlige grupperinger i dataene ønsker vi å visualisere dataene i et 2D-plan. Til dette bruker vi et nevralt nettverk kjent som *Kohonen*. Et Kohonen nettverk er effektiv for representasjon av høydimensjonale data ettersom man kan få visualisert et slikt høydimensjonalt datasett i to dimensjoner. Det er et to-lags selvorganiserende nettverk som kan organisere et topologisk kart ut fra et tilfeldig startpunkt (Kristensen, 1997). Et Kohonen nettverk består av ett input-lag og ett konkurrerende lag som er fullt forbundet. Med *fullt forbundet* menes altså at alle nodene i nettverket har en vekt mellom hverandre. Nodene i nettverket kan vi se for oss er organisert som et todimensjonalt gitter.

Enkelt forklart kan algoritmen forklares som følger: Innledningsvis får vektene i nettverket tilfeldige verdier. Vektene oppdateres så ved trening. Deretter presenteres en inputvektor for inputlaget. Inputvektor i dette tilfellet er en vektor bestående av n komponenter, hvor n er antall attributter i eksempelet. I vårt tilfellet 70. Til slutt summerer nodene i det konkurrerende laget vektet input og konkurrerer for å finne den vinnende noden.

Eksempel på et Kohonen nettverk er illustrert ved figur 3. Nettverket tar en tre-dimensjonal vektor og representerer denne i et to-dimensjonalt plan.



Figur 3 – Kohonen Nettverk

Algoritmen kan mer presist forklares slik:

(1) Vektmatrisen $\overrightarrow{W_{ji}}$ som betegner vektene mellom en node i fra input-laget og node j i det konkurrerende laget blir initialisert med tilfeldige verdier mellom 0 og 1.

(2) En tilfeldig vektor $\vec{X}$ fra treningsdataene representert for nettverket.

(3) Finn noden i det konkurrerende laget som har en vekt-vektor mest lik inputvektoren. Likhet i denne sammenhengen betegnes som euklidsk distanse mellom inputvektor $\vec{X}$ og den tilhørende vekt-vektoren $\overrightarrow{W_j}$ til node j i det konkurrerende laget. Euklidsk distanse defineres som

$$|| \vec{X} - \overrightarrow{W_j} || = \sqrt{\sum_i (x_i - w_{ji})^2}.$$

(1)

Noden i det konkurrerende laget som har vekt-vektoren med kortest euklidske avstand til inputvektoren betegnes som Best Matching Unit (BMU), og blir den vinnende noden.

(4) Identifiser nabomengden N til den vinnende noden. Radiusen som bestemmer antall noder som skal inkluderes i N er et hyperparameter som kan justeres, men typisk settes

denne initielt til

$$\frac{\min(n, m)}{2},$$

(2)

Hvor n og m er dimensjonene på gitteret. Nodene som befinner seg innenfor denne radiusen anses som en del av *N*.

(5) For hver node i *N*, oppdater vektene med følgende formel:

$$w_{ji} = \alpha(x_i - w_{ji}) + w_{ji},$$

(3)

hvor α er læringsraten.
Jo nærmere en node er BMU, desto mer blir vektene til gjeldende node endret.

(6) Gjenta steg 2-6.

7.2. Backpropagation

Backpropagation er en algoritme for å trene et kunstig nevralt nettverk. Algoritmen ser etter et minimum av en kostnadsfunksjon i et vektrom. I backpropagation sier vi vanligvis at vi har en løsning på problemet når vi har et sett med vekter og biaser[2] som minimerer kostnadsfunksjonen. Backpropagation algoritmen kan deles inn i to faser. En *fremover-strømningsfase,* og en *bakover-strømningsfase*. Fremover-strømningsfasen starter med å sette aktiveringen i inputlaget. For hvert lag *l*, beregner vi så outputvektoren

$$a^l = \sigma(w^l + x^{l-1} + b^l)$$

(4)

hvor w betegner vektmatrisen, x er outputvektoren, b er bias og σ er aktiveringsfunksjonen. Dette gjentas for hvert lag i nettverket til vi har beregnet y^ i outputlaget (vi kaller den y^, ettersom output ikke er lik y, men et estimat av y). Når vi trener nettverket trenger vi en metrikk på hvor bra (eller dårlig) nettet presterer. En slik metrikk er kjent som en kostnadsfunksjon, og

[2] En bias er en node i et nevralt nettverk med en konstant verdi (ofte 1)

er et uttrykk for den partiellderiverte $\frac{\partial C}{\partial w}$ (og $\frac{\partial C}{\partial b}$), hvor C er kostnadsfunksjonen med hensyn på en vekt w (eller bias b) i nettverket. Uttrykket forteller oss hvor fort kostnaden endres når vi endrer vektene og biasene, og hvordan en endring i vektene eller biasene endrer den helhetlige oppførelsen til nettverket (Nielsen, 2015). I fremover-strømningsfasen, propagerer vi feilen beregnet av kostnadsfunksjonen bakover gjennom nettverket (derav bakoverstrømning) ved bruk av en iterativ optimeringsalgoritme. Algoritmen vil sakte men sikkert jobbe seg mot et lokalt minima ved å endre på vektene i en retning bestemt av den beregnete gradienten. Gradienten er utregnet basert på kostnadsfunksjonen slik at kostnadsfunksjonen sin output er redusert.

I backpropagation er det en rekke hyperparametere som kan justeres. Blant annet antall skjulte lag, antall noder i hvert skjulte lag, aktiveringsfunksjon, optimeringsalgoritme, kostnadsfunksjon, læringsrate og treningsiterasjoner. Noen av parameterne vil vi justere underveis i forsøkene, mens valg av optimeringsalgoritme, aktiveringsfunksjon, kostnadsfunksjon vil vi bestemme på forhånd. Optimeringsalgoritmen vi skal bruke er kalt Stokastisk Gradient Descent (SGD) og brukes hvis treningssettet er stort (mer enn noen hundre utvalg) og overflødig, og hvis problemet er et klassifiseringsproblem (LeCun mfl., 1998). En aktiveringsfunksjon som har blitt ganske populær de senere årene er Rectified Linear Unit (ReLU) diskutert av Zeiler (2012), og er definert som

$$f(x) = \begin{cases} 0, & x < 0 \\ x, & x \geq 0. \end{cases} \quad (5)$$

Mens andre aktiveringsfunksjoner som for eksempel sigmoid (logistisk funksjon) og tangens hyperbolicus har intervaller på henholdsvis (0,1) og (-1,1), har ReLU et intervall på [0,∞], noe som resulterer i at den ikke møter på problemer som ”the vanishing gradient problem” (Zeiler, 2012). Tatt i betraktning at ReLU returnerer *x* for verdier >= *x*, så kan den ikke brukes i output laget i klassifiseringsproblem siden det da vil være vanskelig å assosiere output med en sannsynlighet. Vi introduserer derfor *softmax,* en aktiveringsfunksjon som vi bare bruker i output laget. Softmax funksjonen beregner sannsynligheten for at et treningseksempel tilhører en spesifikk klasse og fungerer på den måten at den omformer en n-dimensjonal vektor *z* til en n-dimensjonal vektor *k* med verdier i intervallet (0,1) hvor verdiene i *k* summerer til 1. Dette gjør oss i stand til å tolke output fra nettverket som en sannsynlighetsfordeling. Softmax er definert som

$$f_j(z) = \frac{e^{z_j}}{\sum_j e^{f_j}}.$$

(6)

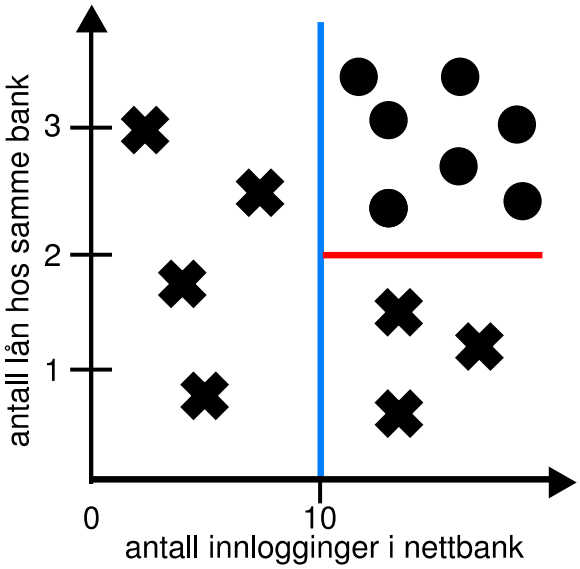
7.3. Beslutningstrær

Trebaserte læringsalgoritmer er blant de beste og mest populære overvåket-læringsmetodene (Analytics Vidhya, 2016), og har en fordel sammenlignet med for eksempel nevrale nettverk ved at de kan være lettere å tolke (Scikit-Learn [2], 2017). Et beslutningstre er en type overvåket lærings algoritme som er mest brukt i klassifiseringsproblem (Analytics Vidhya, 2016). Med denne teknikken splittes utvalget inn i to eller flere homogene sett basert på den attributten og dens verdi som gir det "beste" homogene settet. De finnes en rekke ulike algoritmer for beslutningstrær. Scikit-learn, Python-biblioteket som holder på de fleste algoritmene vi skal anvende, bruker en optimalisert versjon av en algoritme kjent som CART (Classification and Regression Trees) (Scikit-Learn [2], 2017).

CART algoritmen er en av de mest populære og veldig suksessfulle metodene av beslutningstrær (Grabczewski, 2014) og dekker to typer beslutningstrær: klassifiseringstrær og regresjonstrær. Klassifiseringstrær anvendes på klassifiseringsproblem og regresjonstrær tilsvarende for regresjonsproblem.

Representasjonen for algoritmen er et binærtre (Grabczewski, 2014). Et binærtre er en trestruktur hvor hver node har maksimalt to barn. Hver rotnode[3] representerer en input attributt og en verdi hos denne attributten som skal splittes på. Hver bladnode[4] i treet representerer en output attributt med en merkelapp som blir brukt til å klassifisere. La oss ta utgangspunkt i et fiktivt (men relevant) eksempel hvor vi har et datasett og ønsker å klassifisere dataene som enten churn eller ikke-churn. Figuren 4 representerer dataene i et 2D-plott.

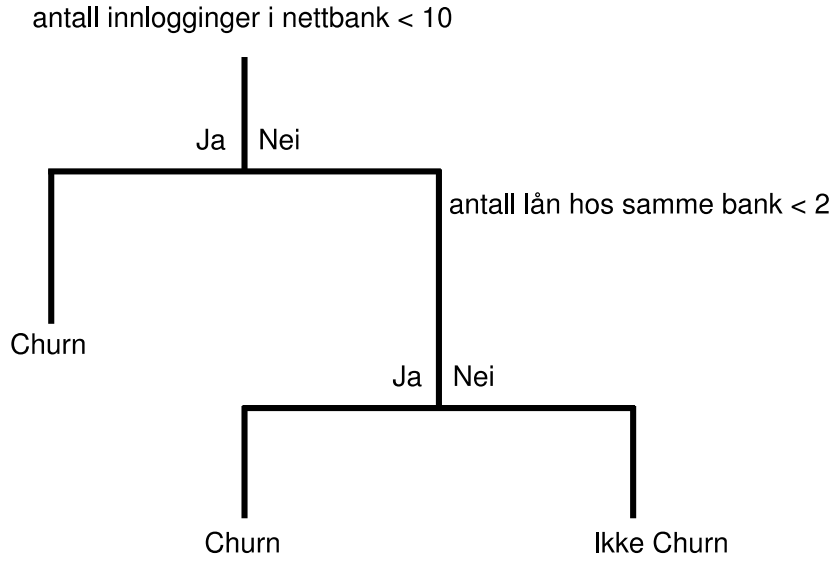
[3] En rotnode er den øverste noden i en trestruktur
[4] En bladnode er en node som ikke har noen barn



Figur 4 –Illustrasjon av et beslutningstrær "splitter" i et 2D-plott

Markøren "x" representerer et datapunkt som er fra klassen ikke-churn, markøren representert som en sirkel viser at datapunktet tilhører klassen churn. Et beslutningstre er en sekvens av binære splitt i datasettet. Intuitivt kan vi se at en god splitt for datasettet illustrert i 4 kan være å starte med å dra en loddrett linje ut fra punktet (10,0).

Videre i figur 5 ser vi hvordan et beslutningstre for datasettet kan se ut. Basert på beslutningstreet kan vi enkelt se hvordan vi kan lage regler for hvordan et utdrag skal klassifiseres.



Figur 5 – Beslutningstre

Beslutningstreet ovenfor kan uttrykkes som et sett av regler:

hvis (antall innlogginger i nettbank < 10) => så churn
hvis (antall innlogginger i nettbank ≥ 10) og (har andre lån hos samme bank) => så ikke-churn
hvis (antall innlogginger i nettbank ≥ 10) og (har ikke andre lån hos samme bank) => så churn

Når ny input blir presentert for treet, ved å starte fra rotnoden, blir treet traversert gjennom å evaluere en spesifikk input helt til utdraget ender opp hos en bladnode og får tilegnet en klasse av modellen. Å lage en CART modell involverer å velge input attributter og splittpunkter til et stoppkriterie er nådd og et tre er konstruert.

Målet med beslutningstrær er å oppnå sett som er mest mulig homogene. Intuitivt kan vi se at en test som produserer helt homogene klasser er bedre enn tester som produserer klasser med både churn/ikke-churn tilfeller. Hvis medlemmene i et sett er fra begge klassene, sitter vi igjen med sett som ikke er ”rene” og vi er dermed usikker på deres klasse . Et mål på en slik usikkerhet kan betegnes ved *entropi* eller *Gini urenhet*. CART algoritmen implementert i Scikit-Learn bruker som standard Gini urenhet, og ettersom valget av type måling på usikkerhet har lite effekt på beslutningstrær (Tan mfl., 2005) tar vi utgangspunkt i Gini urenhet som mål på usikkerhet.

Gini urenhet er et mål på proporsjonene av klasser i et sett (Hackeling, 2014). Gini urenhet er gitt ved følgende likning, hvor *j* er antall klasser, *t* er delmengden av instanser for noden, og P(*i* | *t*)^2 er sannsynligheten for å velge et element fra klasse *i* fra nodens delmengde (Hackeling, 2014):

$$Gini(t) = 1 - \sum_{i=1}^{j} P(i|t)^2$$

Formel 7

Intuitivt er Gini urenhet lik 0 når alle elementene i settet er fra samme klasse. Da sannsynligheten for å velge et element fra den klassen er lik 1. I likhet med entropi, er Gini urenhet størst når hver klasse har lik sannsynlighet for å bli valgt. Maksimum verdien for Gini urenhet er avhengig av antall mulige klasser, og er gitt ved følgende likning (Hackeling, 2014):

$$Gini_{\max} = 1 - \frac{1}{n}$$

Formel 8

I vårt tilfelle, ettersom vi har to klasser (n = 2), vil maksimal verdi for Gini urenhet være

$$1 - \frac{1}{2} = 0.5$$

Formel 9

Når er beslutningstre ikke er altfor komplekst er det enkelt å forstå beslutningsreglene gjennom visualisering av treet. I tillegg er det en rask måte å identifisere attributter med størst innflytelse på resultatet, ettersom algoritmen vil splitte først på de attributtene som reduserer usikkerheten mest.

En ulempe med beslutningstrær er at de har en tendens til å generere altfor komplekse trær som ikke generaliserer særlig godt, også kjent som *overfitting* (Scikit-Learn [2], 2016). Beslutningstrær kan også være ustabile i den sammenheng at små variasjoner i dataene kan resultere i helt ulike trær. Slike problemer kan dempes ved å bruke en samling av beslutningstrær - en *ensemble*.

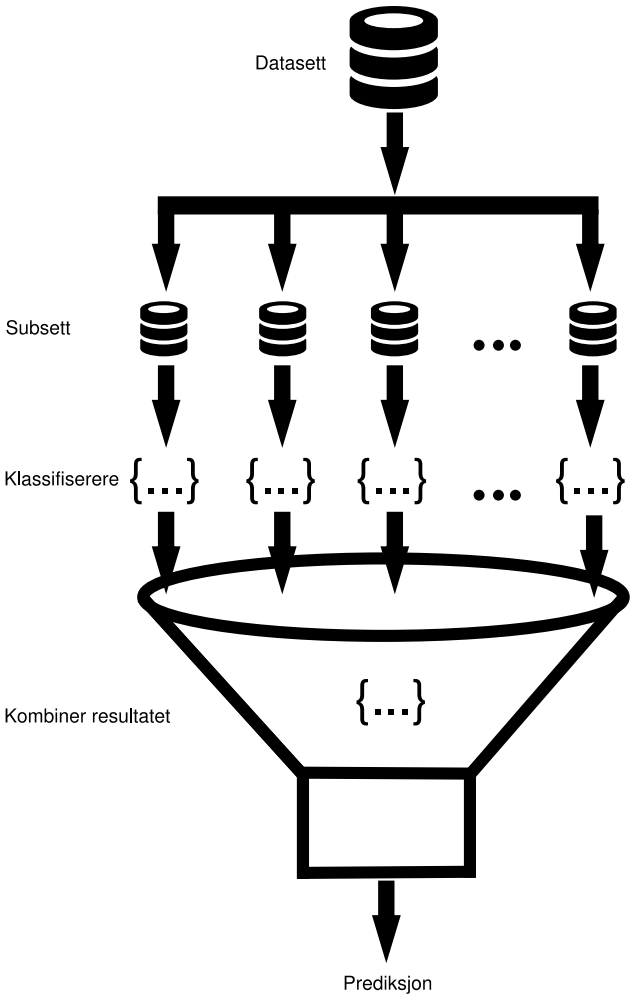
7.4. Ensemble algoritmer

Ensemble algoritmer kombinerer en mengde modeller for å produsere en modell som har bedre prediktiv nøyaktighet enn sine individuelle komponenter. En ensemble kan være en kombinasjon av ulike typer modeller, eksempelvis en kombinasjon av et nevralt nettverk og et beslutningstre. En slik modell er kjent som *stacking*. En ensemble kan også være en kombinasjon av modeller av samme type, eksempelvis en rekke beslutningstrær. En slik modell implementerer da typisk *bagging* eller *boosting* metoden. Ensemble algoritmer får sin styrke ved at de kombinerer flere modeller, enten like eller ulike, og hver modell stemmer på hva den mener riktig output skal være.

7.4.1. Bagging

Bagging (Bootstrap Aggregating) er en metode for å generere flere versjoner av en modell og bruke disse til å få en aggregert modell (Breiman, 1996). Aggregeringen tar gjennomsnittet av output hos de ulike modellene og utfører en majoritetsstemming når den skal predikere en klasse (Breiman, 1996).

Figur 6 illustrerer hvordan Bagging metoden virker. Først tas *t* antall tilfeldige delmengder fra treningsdataene (med tilbakelegging), hvor *t* representerer antall versjoner av modellen vi ønsker å generere. Basert på hver delmengde genereres en modell for hvert utvalg. Til slutt blir resultatene fra hver modell kombinert og output fra modellen bestemmes. I klassifiseringsproblem bestemmes output ved majoritetsstemming. I regresjonsproblem kan man ta et gjennomsnitt av output fra alle modellene, eventuelt et vektet gjennomsnitt om man ønsker å gi mer (eller mindre) ”stemmerett” til enkelte modeller.



Figur 6 – Bagging metoden

En mer presis algoritme for klassifiseringsproblem kan beskrives ved:

Algorithm Bagging

```
1:  procedure MODEL GENERATION
2:      S ← ∅
3:      n ← number of random samples to be drawn from the training data
4:      i ← 0
5:      repeat
6:          r ← random sample from the training data
7:          S.add(r)
8:          apply learning algorithm to r
9:          store the resulting model
10:         i += 1
11:     until i = n
12: procedure CLASSIFICATION
13:     V ← ∅
14:     for all s ∈ S do
15:         v ← predicted class of s
16:         V.add(v)
17:     return mode(V)
```

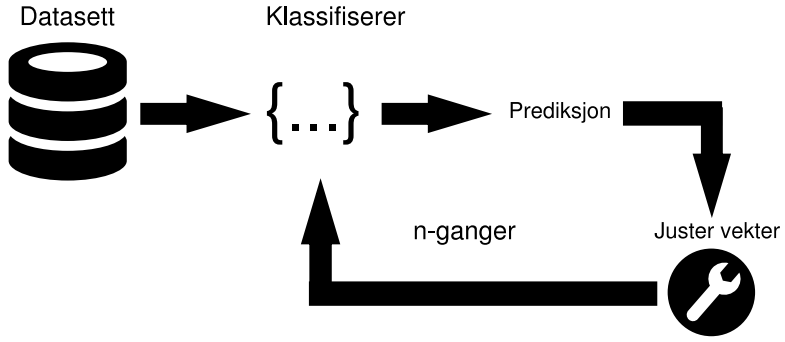
Algoritme 1 – Bagging, basert på Witten (2011)

7.4.1.1. Random Forest

En ensemble algoritme som implementerer bagging metoden er *Random Forest* (RF). RF er en ensemble av beslutningstrær som er trent på tilfeldige delmengder av treningsdataene. Algoritmen er mye anvendt på klassifiseringsproblem og regresjonsproblem innenfor maskinlæring og statistikk (Lakshminarayanan mfl., 2014). RF oppnår konkurransedyktig, prediktiv ytelse og er effektiv å trene og teste med tanke på prosesseringstid, noe som gjør at algoritmen er en utmerket kandidat for reelle prediksjonsoppgaver (Lakshminarayanan mfl., 2014). Tidligere nevnte vi at en ulempe med beslutningstrær er at de har en tendens til å overtilpasse seg treningsdataene. Intuitivt kan vi da se at RF i kontrast til beslutningstrær er lite utsatt for overtilpasning, grunnet at hvert beslutningstre vil overtilpasse seg på ulike delmengder av treningsdataene. Når man da genererer nok trær (hundrevis, eller tusenvis) vil en slik overtilpasning bli jevnet ut gjennom majoritetsstemming. Med flere trær desto mindre vil overtilpasning være i stand til å påvirke modellen.

7.4.2. Boosting

Boosting er en ensemble metode hvor hver modell i "ensemblen" blir påvirket av prestasjonen til den forrige modellen (Witten, 2011). Den første modellen er lært basert på hele datasettet, mens de påfølgende lærer basert på utførelsen av den forrige. Den starter med å klassifisere de originale dataene og gir like vekter til hver observasjon. Hvis klassene er predikert feil av den første modellen, så gir den høyere vekter til den feilklassifiserte observasjonen. Som en iterativ prosess, fortsetter den med å legge til modeller til den når en grense gitt ved antall modeller eller nøyaktighet. Boosting har vist seg å gi bedre resultater enn bagging, men den har også en tendens til å "overfitte" (Witten, 2011). To eksempler på boosting er AdaBoost og XGBoost.



Figur 7 – Boosting metoden

En mer presis algoritme for klassifiseringsproblem kan beskrives ved:

Algorithm Boosting

```
1:  procedure MODEL GENERATION
2:      D ← training data
3:      e ← ∅
4:      M ← ∅
5:      i ← 0
6:      assign equal weight to each training instance in D
7:      repeat
8:          m ← learning algorithm applied to D
9:          e ← error of m
10:         M.add(m)
11:         i += 1
12:     until i = n
13:     if e = 0 or e ≥ 0.5 then
14:         terminate model generation
15:     for all d ∈ D do
16:         if d classified correctly then
17:             multiply weight of d by e/(1-e)
18:     normalize weight of all d in D
19: procedure CLASSIFICATION
20:     assign weight of zero to all classes
21:     for all m ∈ M do
22:         add -log(e / (1-e)) to weight of class predicted by m
23:     return class of instance with max(weight)
```

Algoritme 2 - Boosting, basert på Witten (2011)

7.4.2.1. AdaBoost

AdaBoost er en algoritme introdusert av Freund og Schapire (1995) (Scikit-Learn [1], 2016). Kjerneprinsippet i AdaBoost er å passe en sekvens av svake lærere på gjentatte modifiserte versjoner av dataene. Med en svak lærer her menes modeller som bare er litt bedre enn tilfeldig gjetting, slik som små beslutningstrær (Scikit-Learn [1], 2017).

7.4.2.2. XGBoost

Boosting er en veldig effektiv og utbredt maskinlæringsmetode (Chen & Guestring 2016). Boosting er de facto valget av ensemble metoder og er brukt i konkurranser som ”The Netflix Prize” (Chen & Guestring, 2016). På maskinlæringsplattformen Kaggle[5] brukte 17 av 29 vinnende løsninger XGBoost til å trene sin modell, mens mesteparten av de resterende brukte ensembler av XGBoost og nevrale nettverk (Chen & Guestring, 2016). XGBoost har vist seg å gi utmerket resultater på en rekke problemer deriblant butikksalgprediksjon, kundeatferd, produktkategorisering og prediksjon av miljøkatastrofer (Chen & Guestring, 2016).

[5] https://www.kaggle.com

8. Evaluering

Videre i oppgaven skal vi se på anvendelsen av ulike modeller. Vi skal først se på en modell som implementerer uovervåket læring, før vi går videre på resterende modeller som er av typen overvåket læring. For å implementere modellene tar vi i bruk programmeringsspråket *Python* som har etablert seg som et av de mest populære språkene for vitenskapelig databehandling (Pedregosa mfl., 2011), og *Scikit-learn* som er en Python modul som eksponerer en rekke maskinlæringsalgoritmer, både for overvåket og uovervåket læring (Pedregosa mfl., 2011).

Ved evaluering av en ferdigtrent modell bruker vi en form for *kryssvalidering*. Kryssvalidering er en statistisk metode for å evaluere en modell sin generaliseringsevne på en mer stabil måte enn å dele dataene inn i et treningssett og et testsett (Mueller & Guido, 2016). I kryssvalidering er dataene delt opp gjentatte ganger og flere modeller er trent. Den mest vanlige brukte versjonen av kryssvalidering er *k-veis kryssvalidering* (Mueller & Guido, 2016). I k-veis kryssvalidering representerer *k* antall deler dataene skal deles opp i. Hvis k er lik 5, deles dataene opp i fem like deler. Modellen vil da testes totalt fem ganger. Anta at vi nummerer delene fra 1-5. Ved første iterasjon vil modellen trenes på del 2,3,4 og 5, og testes på del 1. Når modellen testes beregnes en presisjonsverdi som indikerer hvor mange utvalg modellen klarte å klassifiserer riktig. Ved andre iterasjon brukes del 1,3,4 og 5 til treningsdata, og del 2 til testdata. Igjen lagres presisjonsverdien for testdataene. Slik gjentas prosessen til modellen har brukt alle delene (1-5) som testdata. Vi sitter igjen med fem presisjonsverdier, og en vanlig måte å oppsummere nøyaktigheten til kryssvalideringen er å ta gjennomsnittet av verdiene (Mueller & Guido, 2016).

I Scikit-Learn er k-veis kryssvalidering enkelt tilgjengelig og illustrert ved kodeeksempel 1. ”model” representerer en vilkårlig maskinlæringsmodell av typen overvåket læring fra Scikit-Learn biblioteket. ”X” representerer inputdata og ”Y” representerer fasit for inputdataene. Verdien for parameteret ”cv” settes til ”kfold” som da er k-veis kryssvalideringen (cross validation) vi introduserte over.

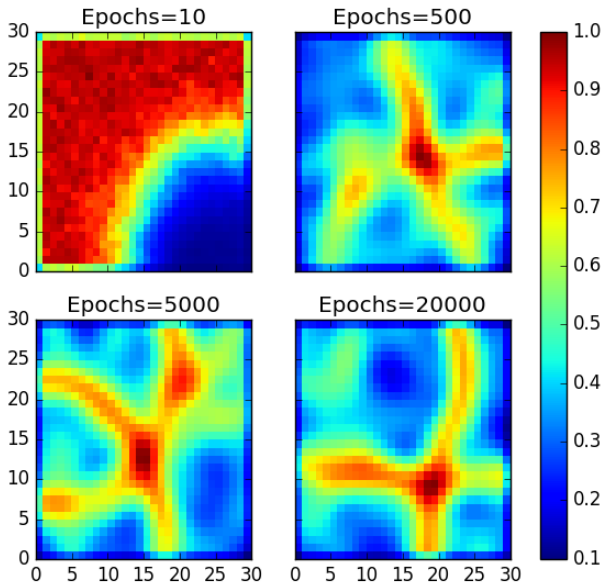
```
from sklearn import model_selection

results = model_selection.cross_val_score(model, X, Y, cv=kfold)
print(results.mean())
```

Kode 1 – k-veis kryssvalidering vha. Scikit-Learn

8.1. Kohonen

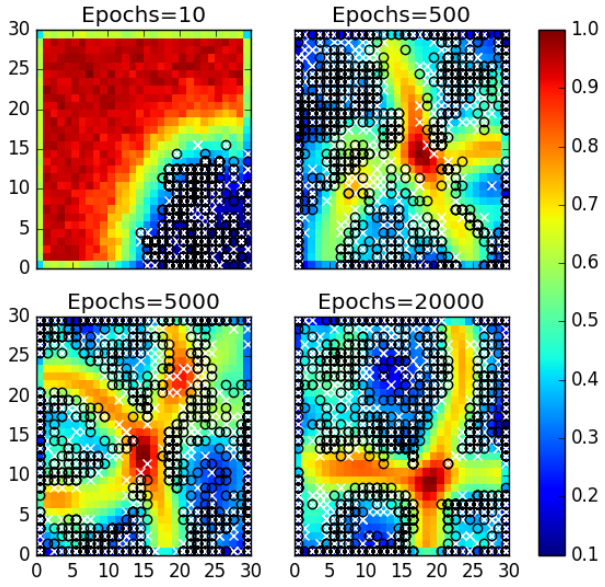
I anvendelsen av datasettet har vi tatt i bruk et Python-bibliotek kalt MiniSom[6]. Vi har eksperimentert litt med ulike parametere. Herunder læringsrate (α), gitterdimensjonene, antall treningsiterasjoner (epochs) og radius for nabomengde. Hvis vi ser tilbake på figur 3 ser vi at det konkurrerende laget i nettverket danner et gitter. Av algoritmen ser vi at når en BMU i det vinnende laget er funnet, oppdateres vektene til nodene i BMU sin nabomengde for at de skal bli mer lik inputvektoren. Her kan vi tenke oss at BMU og alle nodene i N blir dratt mot inputvektoren. Noder som er lik en inputvektor, vil bli dratt mot denne, og ved gjentakelse vil det etter hvert kunne bli dannet grupperinger i det selvorganiserende laget. Fargene i figur 8 indikerer den normaliserte summen av avstanden mellom en node og node i nabomengden. Fargen indikerer altså hvor tett nodene i et område ligger. I et svart/mørkt område, er node nærmest helt inntil hverandre, noe som indikerer at det er funnet likheter mellom flere inputvektorer i dataene, mens i et rødt område er nodene et stykke fra hverandre, noe som indikerer at det ikke er noe grupperinger i dette området. I forsøket som illustreres i figur 8 brukte vi en læringsrate lik 0.3, og en radius på 5. Som nevnt er en vanlig startverdi for radius $\frac{\min(n,m)}{2}$, men en så høy radius gav dårlige resultater. Størrelsen på gitteret har vi (etter prøving og feiling) satt til 30x30, som resulterer i et konkurrerende lag med 900 neuroner.

[6] https://github.com/JustGlowing/minisom



Figur 8 – Illustrasjon av hvordan strukturen i Kohonen ser ut ved ulike iterasjoner

Vi kan se hvordan gitteret endrer strukturer etter hvert som antall treningsiterasjoner øker. Videre ser vi hvordan det danner en tydelig struktur, og at den finner likheter blant radene i datasettet. Hvis vi legger på markører med fasit (churn, ikke-churn) for de ulike punktene i datasettet kan vi se på figur 9 hvordan nettverket har klassifiserer disse.



Figur 9 – Illustrasjon av hvordan strukturen i Kohonen ser ut ved ulike iterasjoner med markører for churn, ikke-churn

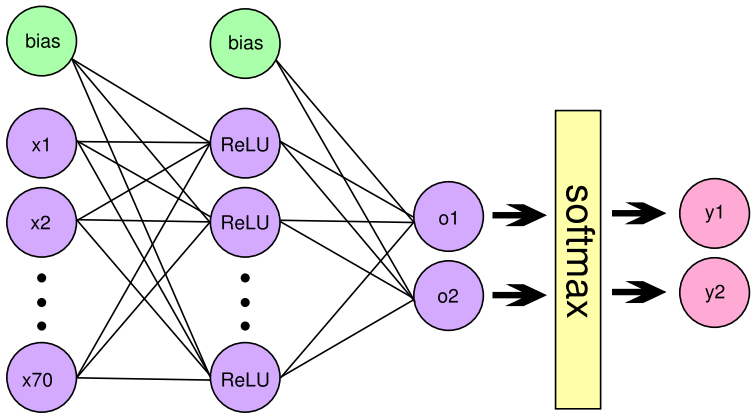
I figur 9 har vi da to ulike markører. Hvitt kryss representerer en kunde som ikke har sagt opp, mens svart sirkel representerer en kunde som har sagt opp. Vi ser altså at det ikke er tydelige

grupperinger mellom disse to klassene, til tross for at nettverket har klart å finne noe mønster i dataene. Nettverket finner altså likheter mellom noen datapunkter, og ulikheter mellom noen datapunkter, men det som skiller disse grupperingene er tilsynelatende *ikke* om de har sagt opp eller ikke.

Den neste modellen vi skal se på er også et nevralt nettverk, men som implementerer overvåket læring, nettverket er kjent som Backpropagation.

8.2. Backpropagation

Figur 10 illustrerer det nevrale nettverket når vi har konstruert det med ett skjult lag.



Figur 10 – Illustrasjon av vårt nevrale nettverk

For å implementere nettverket bruker vi et bibliotek kjent som *Tensorflow*[7]. Tensorflow er et "open source" programvarebibliotek utviklet av Google for numerisk prosessering (Tensorflow, 2017). På lik linje med Scikit-Learn gir biblioteket oss et grensesnitt mot ulike modeller, vi kan anvende for å utvikle robuste nevrale nettverk.

Tabell 4 viser kjøringer med ulike verdier for de ulike hyperparameterne.

Antall skjulte lag	**Gjennomsnitt antall noder i skjulte lag**	**Læringsrate**	**Treningsiterasjoner**	**Treffprosent**
1	100	0.001	1000	69 %
2	100	0.001	1000	60%

[7] https://www.tensorflow.org/

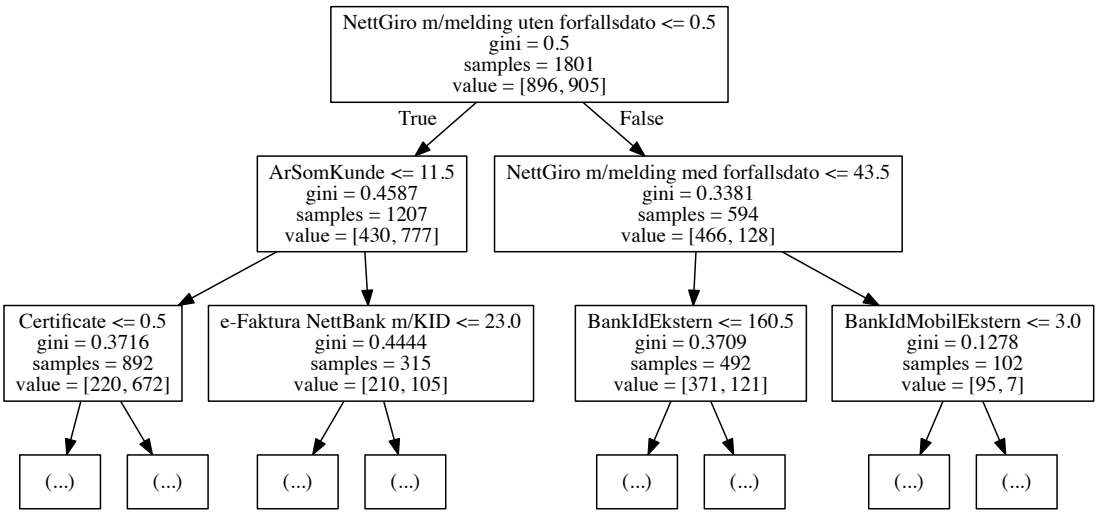
3	100	0.001	1000	61%
3	100	0.01	1000	59%
1	20	0.01	1000	64%
1	50	0.01	1000	67%
2	25	0.01	2000	65%
1	50	0.01	5000	62%

Tabell 4 – Resultat av kjøringer med ulike hyperparametere

Vi ser fra tabellen over at nettverket ikke klarer å oppnå en bedre presisjon enn 69%, og at det er en mindre kompleks utgave med bare ett skjult lag som tilsynelatende har den beste generaliseringsevnen.

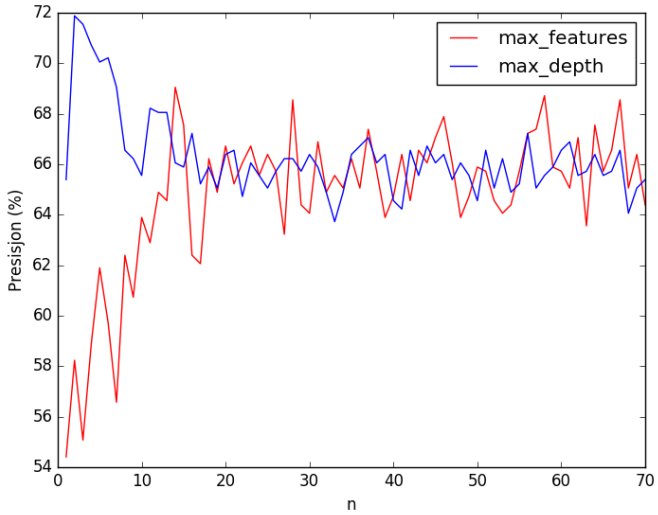
8.3. Beslutningstre

Som nevnt bruker vi Scikit-Learn sin implementasjon av CART algoritmen for å modellere et beslutningstre. Det første vi gjør er å konstruere et beslutningstre fra datasettet vi har. Figur 11 illustrerer beslutningstreet med en dybde lik 2. Vi har altså droppet å vise resten av treet da det i vårt tilfelle med 70 trekk ville tatt flere sider å visualisere. Vi kan se av figuren at vi starter med en Gini urenhet på 0.5, noe som stemmer bra da vi har en like mange utdrag fra begge klasser. Modellen anser attributten NettGiro m/melding uten forfallsdato som den viktigste faktoren gitt at dette er attributten den splitter først på. Husk fra tidligere at modellen er en grådig algoritme som ønsker å minimere Gini urenhet, og vil som følge av dette splitte på den faktoren som (lokalt) gir den minste Gini urenheten.



Figur 11 - Beslutningstre

Beslutningstrær har ulike parametere som kan justeres for å passe datasettet bedre (uten at det nødvendigvis er noe garanti for at resultatet vil bli bedre). Blant disse finner vi maks dybde på treet (max_depth i Scikit-Learn sin DecisionTreeClassifier), hvor dybde defineres som lengden av den lengste stien fra rotnoden til en bladnode, og maks antall attributter (max_features i Scikit-Learn) som skal vurderes når en node skal se etter den beste splitten for å minimere Gini urenhet. Figur 12 illustrerer modellen sin presisjon ved ulike verdier for disse parameterne.



Figur 12 – Illustrasjon av hvordan beslutningstreet sin presisjon endrer seg ved ulike verdier for max_features og max_depth

Vi ser at isolert sett, så er de beste verdiene for parameterne max_depth og max_features henholdsvis omtrent 4 og omtrent 13.

8.4. Random Forest

RF algoritmen er tilgjengelig for oss ved bruk av Scikit-Learn biblioteket. Kode 2 illustrerer en standard RF klassifiserer. Som tidligere nevnt bruker vi k-veis kryssvalidering for validering av modellens presisjon.

```
from sklearn import model_selection
from sklearn.ensemble import RandomForestClassifier
from data_loader import DataLoader

X,Y = DataLoader().load_data()

seed = 7
kfold = model_selection.KFold(n_splits=10, random_state=seed)
model = RandomForestClassifier(500)
results = model_selection.cross_val_score(model, X, Y, cv=kfold)
print(results.mean())
```

Kode 2 – Random Forest vha. Scikit-Learn

I Scikit-Learn sin implementasjon av RF finnes en rekke parametere som kan påvirke modellen sin presisjon:

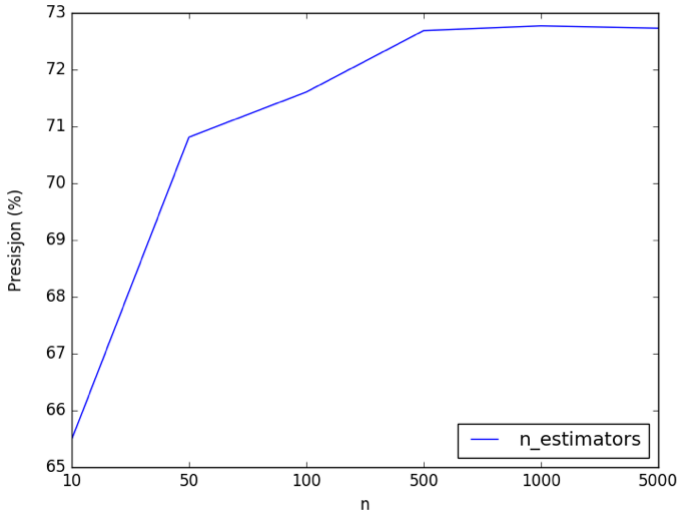
- *n_estimators*: Antall trær som skal generes. Flere trær gir som regel en modell som er mer nøyaktig på ukjente data, men som konsekvens tar modellen lengre tid å trene.
- *criterion*: Hvilken funksjon som skal brukes som mål på usikkerhet. Gini urenhet er standard criterion, og den funksjonen vi vil bruke.
- *max_features*: Antall attributter som skal tas i betraktning når det letes etter den beste splitten.
- *max_depth*: Maksimal dybde til et tre.
- *min_samples_split*: Minste antall størrelse på utvalg krevd for å splitte en intern-node[8].
- *min_samples_leaf*: Minste antall prøver krevd for å være en bladnode.
- *max_leaf_nodes*: Maksimall antall bladnoder.
- *min_impurity_split*: Terskel for å tidlig stoppe ”treveksten”. En node vil splittes hvis urenheten er over terskelen, ellers vil noden bli en bladnode.

(Scikit-Learn [3], 2017)

Med utgangspunkt i resultatene fra beslutningstreet representert ved figur 12 prøver vi oss frem

[8] En intern-node er en node som har ett eller flere barn

med RF. Fra figur 12 ser vi at den beste verdien isolert sett for max_features er ca 15, mens den beste verdien isolert sett for max_depth er ca. 4. RF med 10 trær resulterte i en presisjon på 66,2%. Ettersom RF implementerer bagging modellen hvor resultatet er et gjennomsnitt av en mengde beslutningstrær, vil variansen endre seg med antall trær i ensemblen. Flere trær resulterer i mindre varians. Med en lavere varians kan vi intuitivt se at modellen oppnår en større generaliseringsevne, men som konsekvens gir lengre prosesseringstid. Figur 13 illustrerer hvordan modellen sin presisjon endrer seg med ulike mengder trær.



Figur 13 – Illustrasjon av hvordan RF sin presisjon endrer seg ved ulike verdier for n_estimators

Vi ser at presisjonen forbedrer seg betraktelig når vi går fra 10 til 50 trær. Videre ser vi at presisjonen stadig forbedrer seg etter hvert som antall trær øker, men at den stabiliserer seg når RF genereres med 1000 eller flere beslutningstrær. Noe som passer bra med utsagnet tidligere hvor vi nevnte at variansen reduseres etter hvert som antall beslutningstrær i en RF øker.

RF har som nevnt en rekke parametere som kan justeres. I et senere avsnitt vil vi bruke en metode kalt GridSearchCV i Scikit-Learn for å effektivt optimalisere en mengde parametere.

8.5. AdaBoost

I likhet med RF er AdaBoost enkelt tilgjengelig, og en enkel implementasjon som anvender AdaBoost kan illustreres ved kode 3.

```
from sklearn import model_selection
from sklearn.ensemble import AdaBoostClassifier
from data_loader import DataLoader

X,Y = DataLoader().load_data()

seed = 7
kfold = model_selection.KFold(n_splits=10, random_state=seed)
model = AdaBoostClassifier(500)
results = model_selection.cross_val_score(model, X, Y, cv=kfold)
print(results.mean())
```

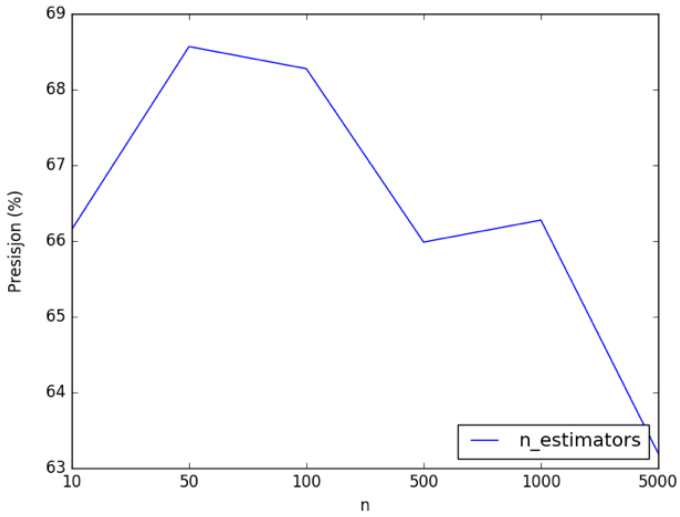
Kode 3 – AdaBoost vha. Scikit-Learn

Scikit-Learn sin implementasjon av AdaBoost har ulike parametere som kan justeres, deriblant:

- *base_estimator*: Hvilken modell som skal brukes til å generere ensemblen. Standard er beslutningstre.
- max_estimators: Maks antall modeller som brukes i ensemblen.
- *learning_rate*: Si at læringsraten er gitt ved α, læringsraten vil så redusere hver modell sitt bidrag til ensemblen med α.

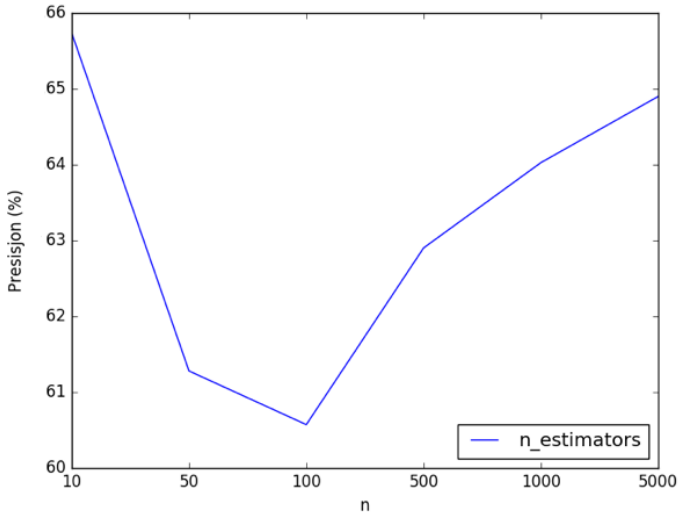
(Scikit-Learn [4], 2017)

En kjøring med AdaBoost og en størrelse på ”ensemblen” på ti beslutningstrær, resulterte i en presisjon på 66%. Figur 14 viser hvordan presisjonen til AdaBoost endrer seg etter hvert som vi endrer størrelsen på ”ensemblen” (endring i max_estimators).



Figur 14 - Illustrasjon av hvordan AdaBoost sin presisjon endrer seg ved ulike verdier for n_estimators

Vi ser at AdaBoost faktisk gir best resultat med en ensemble bestående av 50 beslutningstrær, men likevel betraktelig dårligere resultat sammenlignet med RF. Tidligere nevnte vi at AdaBoost gir oss muligheten til å sette hvilken type modell man ønsker å bruke i "ensemblen" vår. Figur 15 viser hvordan presisjonen til AdaBoost endrer seg ved endring i antall modeller i "ensemblen". Denne gangen bruker vi de samme parameterne satt til de samme verdiene for beslutningstrærene vi brukte i RF (som vi illustrerte i figur 13).



Figur 15 - Illustrasjon av hvordan AdaBoost sin presisjon endrer seg ved ulike verdier for n_estimators (merk: ulik figur 14)

Fra figur 15 ser vi at AdaBoost er et stykke unna når det kommer til presisjon sammenlignet med RF.

8.6. XGBoost

XGBoost er ”open source” og tilgjengelig for oss på GitHub[9]. XGBoost har en Scikit-Learn ”wrapper” rundt den originale implementasjonen for å gi oss et kjent grensesnitt å jobbe med. Illustrert ved kode 4 ser vi at anvendelsen av XGBoost er påfallende lik RF og AdaBoost.

```
1  from sklearn import model_selection
2  from xgboost import XGBClassifier
3  from data_loader import DataLoader
4
5  X,Y = DataLoader().load_data()
6
7  seed = 7
8  kfold = model_selection.KFold(n_splits=10, random_state=seed)
9  model = XGBClassifier(n_estimators=500)
10 results = model_selection.cross_val_score(model, X, Y, cv=kfold)
11 print(results.mean())
```

Kode 4 – XGBoost vha. et Scikit-Learn grensesnitt

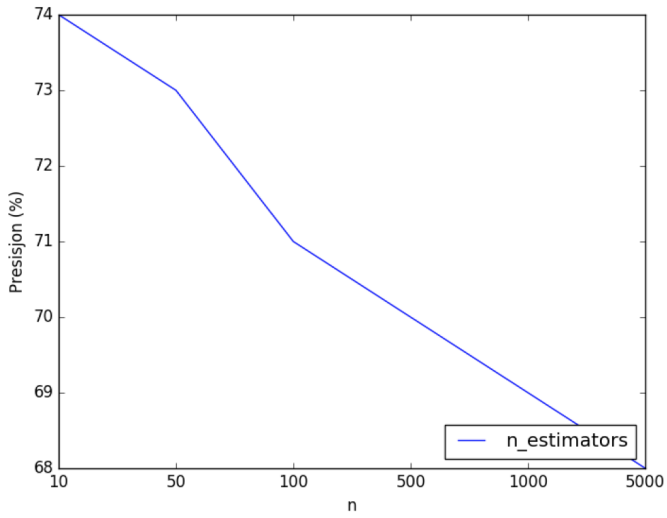
Grensesnittet har en rekke parametere som er blitt gjort tilgjengelig for oss, deriblant:

- *max_depth:* Maksimal tredybde for baselærerne.
- *learning_rate:* Læringsraten. Påvirker hvor mye vektene skal justeres.
- *n_estimators:* Antall modeller i ”ensemblen”.
- *objective:* Hvilken objektfunksjon som skal anvendes.
  For eksempel:
  - *binary:logistic* for binære klassifiseringsproblem.
  - *multi:softmax* for fler-klassifiseringsproblem.
- *gamma:* Minimum reduksjon i ”loss” som er krevd for å splitte en bladnode.
- *min_child_weight:* Minste sum av instansevekt som er nødvendig som er nødvendig i et barn. Hvis tredelingssteget resulterer i en bladnode hvor summen av instansevekter er mindre enn min_child_weight, så stoppes videre deling av noden.
- *subsample:* Rate som kontrollerer størrelsen på delutvalget fra datasettet når trær skal genereres. 0.5 resulterer i at halve datasettet blir inkludert.
- *colsample_bytree:* Rate som kontrollerer størrelsen på delutvalget av attributter når et tre skal genereres.
- *colsample_bylevel:* Rate som kontrollerer størrelsen på deultvalget av attributter for hver splitt.

[9] GitHub er et web-basert kodearkiv med versjonskontroll

(XGBoost, 2017)

Figur 16 illustrerer hvordan XGBoost sin presisjon endrer seg ved endring i størrelsen på ”ensemblen”.



Figur 16 - Illustrasjon av hvordan XGBoost sin presisjon endrer seg ved ulike n_estimators

Ikke overraskende har XGBoost resultert i den beste presisjonen så langt, på hele 74%. Vi ser og at i motsetning til bagging algoritmen som blir bedre etter hvert som størrelsen på ensemblen vokser, så ser vi at vi ikke en slik sammenheng hos boosting algoritmene.

8.7. Forbedring

I dette delkapittelet vil vi forsøke ulike metoder for å øke modellenes presisjon. Først ønsker vi å se på hvordan dette kan oppnås ved å justere en mengde parametere, før vi går videre og anvender attributtseleksjon for å forbedre modellene.

8.7.1. Parameterjustering

Algoritmene vi har vært igjennom har en rekke ulike parametere som kan justeres. Å finne en optimal eller ”god nok” kombinasjon av verdier for de ulike parameterne er tidkrevende. Scikit-learn tilbyr en metode kalt *GridSearchCV* for å enklere kunne utføre parameteroptimalisering på modeller fra Scikit-Learn biblioteket eller som implementerer et slikt grensesnitt.

Kode 5 illustrerer et eksempel på hvordan vi bruker GridSearchCV til å forbedre modellen sin presisjon. Metoden tar inn ulike parameter, deriblant ulike mengder parametere som vi ønsker å teste. I tilfellet under gir vi metoden parameterne max_depth og n_estimators med

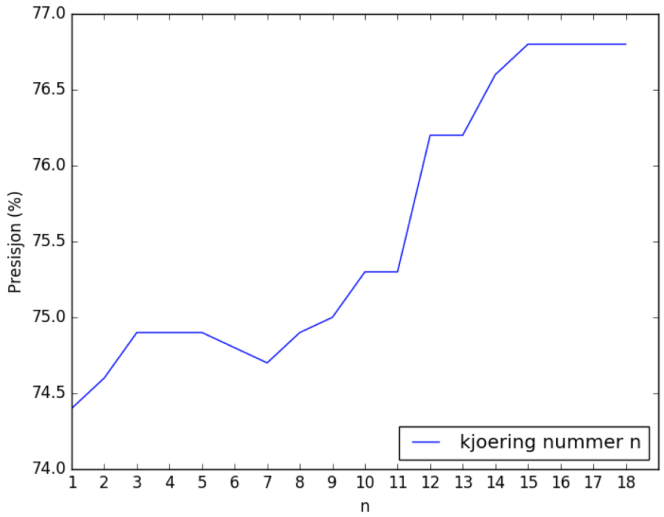
henholdsvis verdiene [3,4,5] og [9,10,11] (årsaken til at vi valgte disse verdiene er at om vi ser tilbake på resultatene fra tidligere så er det verdier i nærheten av disse tallene som har gitt best resultat). Modellen vil så prøve ulike kombinasjoner av disse parameterne og lagre resultatene fra alle kjøringer. Vi kan så enkelt hente ut den beste presisjonen den klarte å oppnå med gitte verdier samt hvilke verdier for parameterne som gav best resultat.

```
1  from xgboost import XGBClassifier
2  from data_loader import DataLoader
3  from sklearn.grid_search import GridSearchCV
4
5  X,Y = DataLoader().load_data()
6
7  cv_params = {'max_depth': [3,4,5], 'n_estimators': [9,10,11]}
8  params = {'objective': 'binary:logistic'}
9  model = GridSearchCV(XGBClassifier(**params), cv_params, scoring='
       accuracy', cv=5)
10 model.fit(X, Y)
11 print model.best_params_, model.best_score_
```

Kode 5 – GridSearchCV og XGBoost vha. Scikit-Learn og XGBoost sitt Scikit-Learn grensesnitt

(Parameteret ”objective” settes til ”binary:logistic” ettersom vi har et binært klassifiseringsproblem.)

Ved kjøring av kodeeksempelet over endte vi opp med det beste resultatet så langt; en presisjon på 75%, og verdiene som resulterte i denne presisjonen var ”n_estimators=10”, og ”max_depth=5”. Videre tar vi med oss disse verdiene og prøver å optimalisere resten av parameterne. Figur 17 viser hvordan modellen sin presisjon endrer seg ved ulike verdier for en rekke parametere. Ved kjøring 1 hadde vi allerede søkt etter forbedringer i presisjonen med hensyn på parameterne max_depth og n_estimators. Fra kjøring 1 til kjøring 11 endret vi på resterende parametere, før vi fra kjøring 11 og utover gikk tilbake og testet nye verdier for max_depth og n_estimators for å se om justeringene vi gjorde på parametere som eksempelvis gamma hadde en innvirkning her. Vi ser at max_depth og n_estimators er de parameterne som gir mest utslag på modellen sin presisjon. Vi endte til slutt opp med følgende verdiene for de to parameterne: ”max_depth=7”, ”n_estimators=500”, som resulterte i en presisjon på 76.8%.

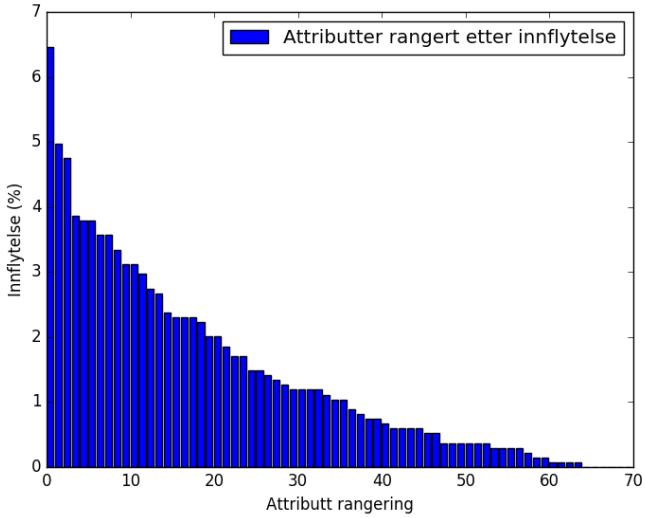


Figur 17 - Illustrasjon av hvordan XGBoost sin presisjon endrer seg ved endring i ulike hyperparametere

8.7.2. Attributtseleksjon

I tillegg til parameteroptimalisering kan vi også anvende attributtseleksjon for å prøve å forbedre modellen sin presisjon. Attributtseleksjon handler om å finne en optimal kombinasjon av attributter for å øke modellen sin presisjon. Samtidig kan det være tilfeller hvor det lønner seg å inngå en kompromiss mellom presisjon og antall attributter. For eksempel innen forretningsinnsikt hvor man ofte kan få mer verdi av å sitte igjen med ti attributter sammenlignet med 70.

XGBoost tilbyr en funksjon som rangerer viktigheten av ulike parametere. Basert på en slik metode kan vi generere en liste med attributter rangert etter viktigheten til en attributt. Listen er illustrert ved figur 18.



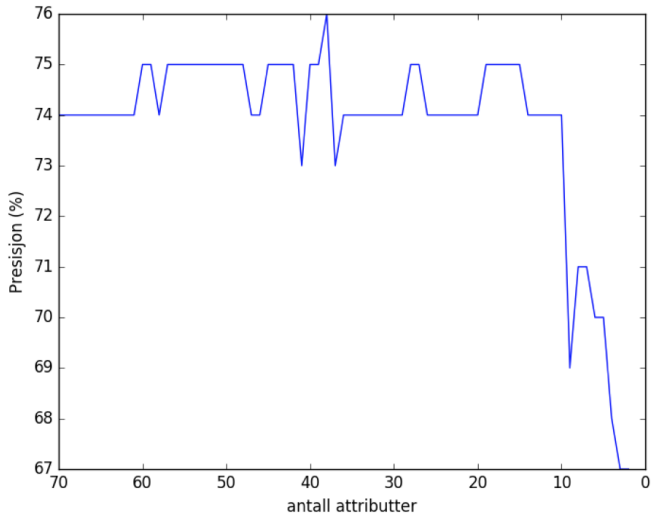
Figur 18 – Illustrasjon av hvordan XGBoost rangerer attributtene med hensyn på innflyelse på resultatet

Vi ser fra figur 18 at modellen mener det er en nokså gradvis reduksjon i innflytelse hos de ulike attributtene. Attributtene som modellen mener er det fem viktigste er illustrert i tabell 5.

Attributt	**Innflytelse (%)**
ArSomKunde	6.5%
Varer/bensin betjent betalingsterminal m/kvittering	5.0%
Nettgiro m/melding med forfallsdato	4.8%
AlderIValgtAr	3.9%
e-Faktura Nettbank m/KID	3.8%

Tabell 5 – Topp 5 attributter fra XGBoost sin attributtrangering

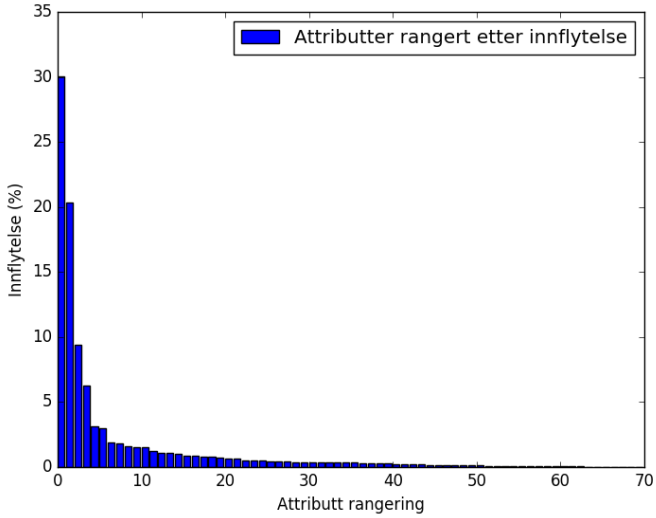
Videre kan vi anvende en algoritme som fjerner attributter én etter én, hvor man starter med den som anses som den minst viktige. Med utgangspunkt i modellen ovenfor illustrerer figur 19 hvordan modellen sin presisjon endrer seg ved kjøringer med ulike antall attributter.



Figur 19 - Illustrasjon av hvordan XGBoost sin presisjon endrer seg ved endring i antall attributter i en kjøring

Fra figur 19 kan vi se at modellen sin presisjon starter stabilt på 74%, og når en topp på 76% med omlag 37 attributter. Videre kan vi se at med bare 11-12 attributter har modellen fortsatt en nokså høy presisjon på like under 74%. Årsaken til at det er et avvik mellom presisjonen her og forsøket illustrert ved figur 17 er at modellen er randomisert av natur, og at det i dette forsøket ikke er brukt k-veis kryssvalidering.

Vi har også undersøkt hvilke attributter RF anser som de viktigste og plottet dette i figur 20. Sammenligner vi tabell 5 (XGBoost) og tabell 6 (RF) som viser de fem attributtene som har størst innflytelse for hver algoritme kan vi se at kun ArSomKunde går igjen i begge tabellene. En annen forskjell er at de to attributtene som har største innflytelse i RF har ekstremt høy innflytelse med henholdsvis 30% og 20%. Dette gjør at det er mange attributtene med liten innflytelse. 52 av attributtene har under 1% innflytelse. Da vi gjorde et forsøk med de 20 attributtene som har mest innflytelse sank presisjonene kraftig, noe som tilsier at RF ikke klarer å klassifisere tilnærmet like godt om man reduserer antall attributter på samme måte som XGBoost.



Figur 20 - Illustrasjon av hvordan RF rangerer attributtene med hensyn på innflyelse på resultatet

Attributt	**Innflytelse (%)**
NettGiro m/melding uten forfallsdato	30.1%
ArSomKunde	20.3%
Certificate	9.4%
Kreditrente	6.2%
Gebyr	3.2%

Tabell 6 - Topp 5 attributter fra RF sin attributtrangering

9. Analyse

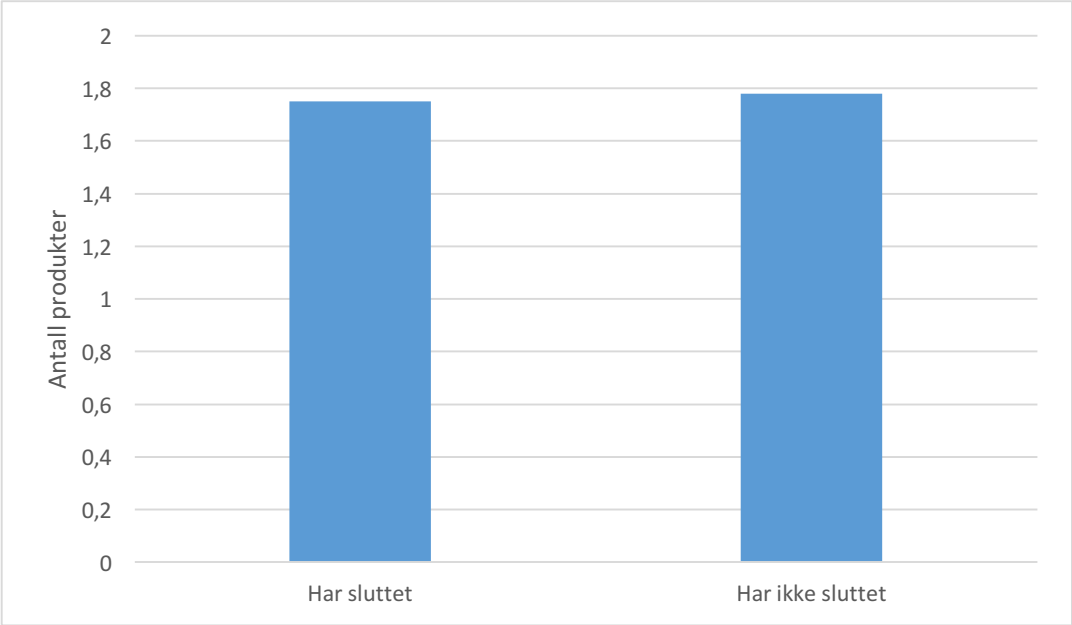
I denne delen vil vi analysere forskjeller og likheter mellom de som ikke har churnet og de som har churnet. Vi vil først se på hvordan hypotesene vi trakk ut fra det tidligere arbeid stemmer overens med våre data. For denne analysen vil vi bruke det balanserte datasettet med 50% churnere og 50% ikke-churnere. Det betyr at den gjennomsnittlige sannsynligheten for at en kunde skal si opp lånet sitt blir 50%.

For å undersøke om det er signifikant forskjell mellom de som har sluttet og de som ikke har sluttet vil vi gjennomføre t-tester. T-test er en statistisk metode for å beregne sannsynligheten for at den observerte forskjellen mellom to grupper skyldes tilfeldigheter. Nullhypotesen er at det er ingen forskjell mellom de som har sluttet og de som ikke har sluttet. Vi velger et signifikansnivå på 5%, og forkaster nullhypotesen dersom p-verdien er mindre enn 0,05. Dersom vi forkaster nullhypotesen betyr dette at det er sannsynlig at det er forskjell mellom de som har sluttet og de som ikke har sluttet, og at dette ikke er tilfeldig.

9.1. Flere produkter reduserer risikoen for å slutte

Vi begynner med den første hypotesen om at flere produkter reduserer risikoen for å slutte. Hypotesen har vi undersøkt ved å se på fem forskjellige attributter, RammeKontokreditt, RammeBoligkreditt, RammeKredittkort, AntallAktiveBoliglån og AntallAktiveBillån. Definisjonen for at en kunde har et av disse produktene er at attributten må ha verdi større enn null.

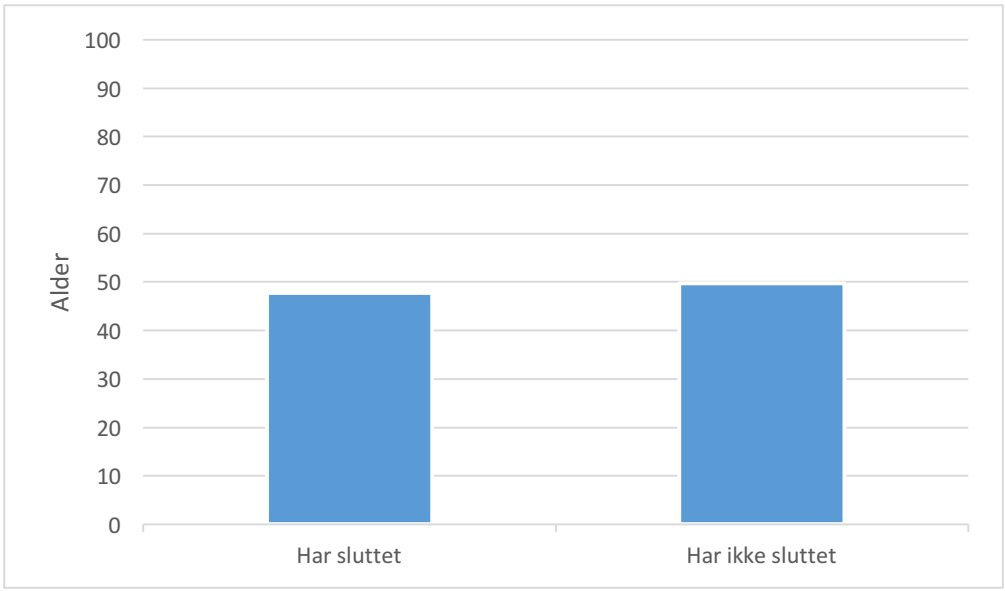
Som vi kan se i figur 21 er det ikke holdepunkt for å underbygge denne hypotesen. T-testen viser at p-verdien er 0,74, noe som betyr at vi må beholde nullhypotesen som sier at det er ingen forskjell mellom de som har sluttet og de som ikke har sluttet. Alle kundene havner i en av kategoriene 1, 2, 3 og 4 (antall produkter), og det er ikke nevneverdig forskjell i sannsynlighet for å slutte i noen av disse kategoriene. Alle ligger rundt det totale gjennomsnittet på 50%.



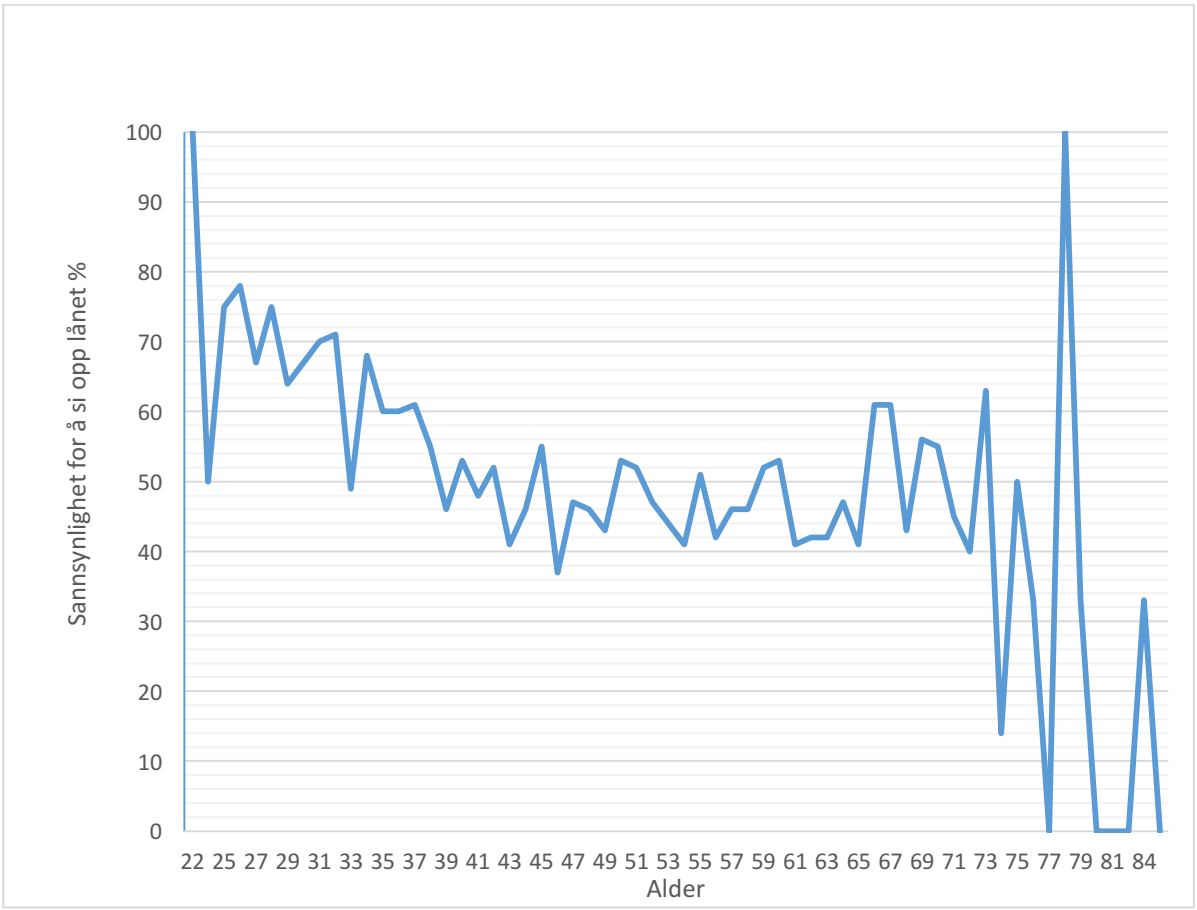
Figur 21 – Gjennomsnittlig antall produkter

9.2. Eldre har lavere risiko for å slutte

Den andre hypotesen var at eldre har lavere risiko for å slutte. Som vi kan se på figur 22 er gjennomsnittsalderen for de som ikke har sluttet noe høyere med 49,79 mot 47,69. T-testen viser en p-verdi på 0,0001, noe som betyr at vi kan forkaste nullhypotesen og si at det sannsynlig at det er forskjell mellom de som har sluttet og de som ikke har sluttet, og at dette ikke er tilfeldig. Figur 23, som viser sannsynligheten for å slutte i en gitt alder er mer interessant. Her kan vi se en tydelig trend for hvilke aldre som er mer utsatt for å slutte. De yngste har størst risiko for å slutte, mens kundene i alderen fra 40-65 år er mer lojale. For kundene over 65 år øker risikoen igjen. Blant de aller eldste er det noe sprikende fra år til år, noe som gjerne kan begrunnes med tynt datagrunnlag i alderen over 75 år.



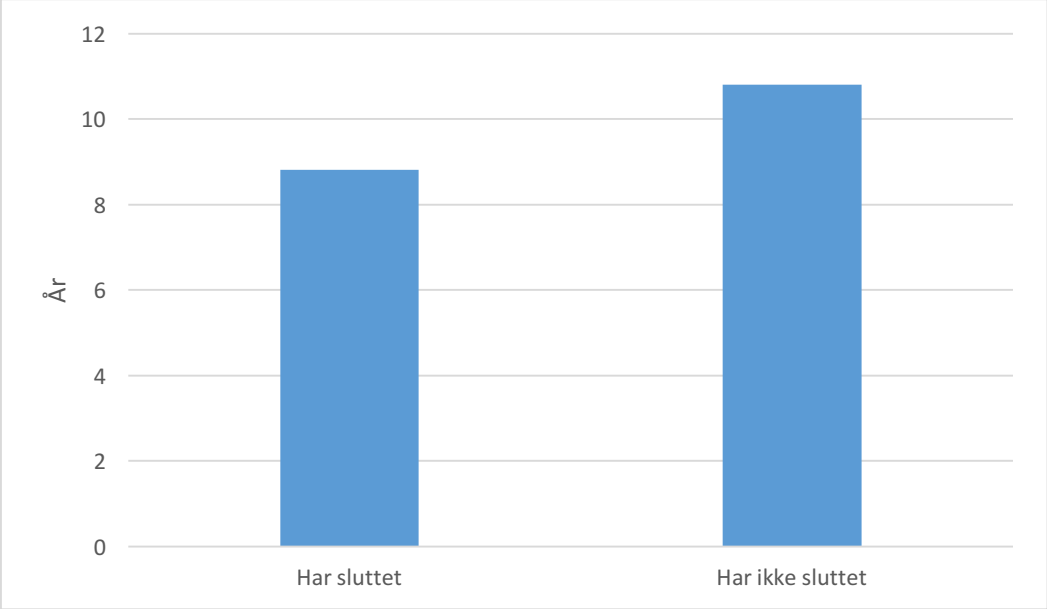
Figur 22 – Gjennomsnittlig alder



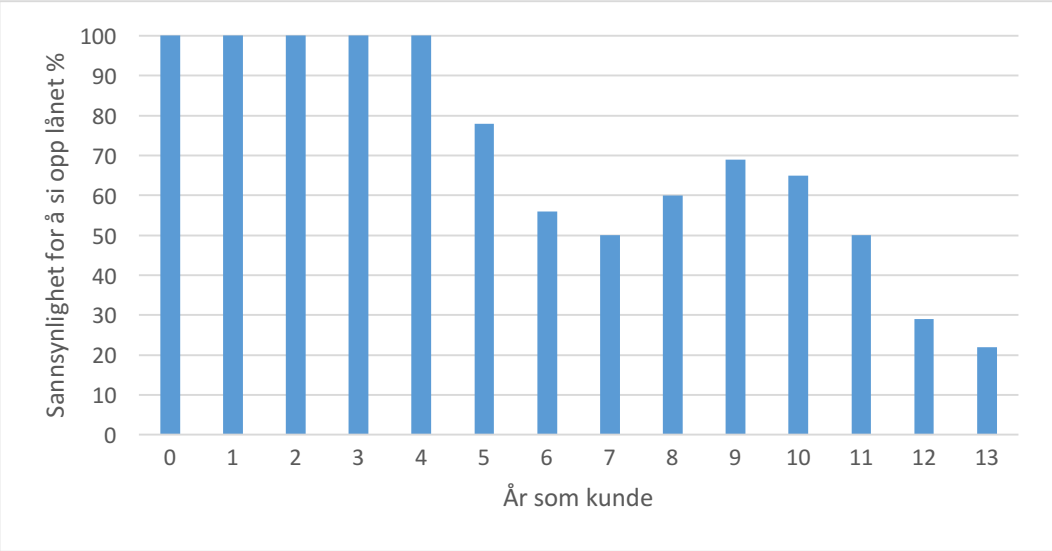
Figur 23 – Sannsynlighet for å slutte fordelt på alder

9.3. Kunder som har vært kunde lenge har lavere risiko for å slutte

Når vi setter opp den gjennomsnittlig lengden over de som har sluttet har vært kunde opp mot de som ikke har sluttet, her vist i figur 24, kan vi se at de som har vært kunde lengre har lavere risiko for å slutte. T-testen viser en p-verdi på 0,0001, noe som betyr at vi kan forkaste nullhypotesen og si at det sannsynlig at det er forskjell mellom de som har sluttet og de som ikke har sluttet, og at dette ikke er tilfeldig. Mer interessant er gjerne figur 25 hvor vi ser på sannsynligheten for å slutte ut ifra antall år som kunde. Her kan vi se at alle som har vært kunde i fire år eller mindre har sagt opp boliglånet sitt. Perioden er altså veldig kritisk. Videre synker sannsynligheten for å slutte frem til år syv, før den øker litt igjen i år åtte og ni. Fra år ti og utover finner vi de mest lojale kundene med lavere og lavere sannsynlighet for hvert år.



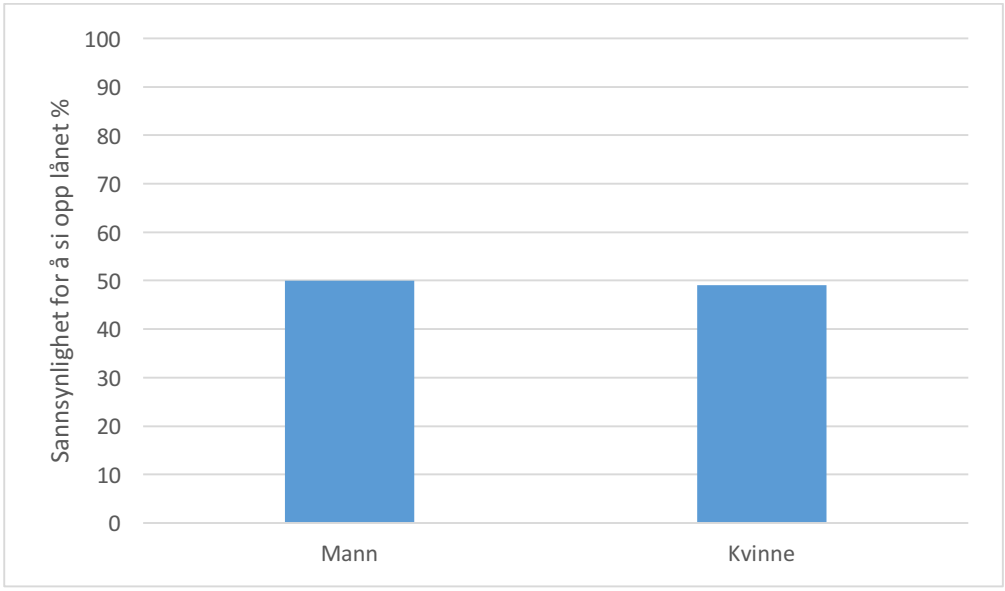
Figur 24 – Gjennomsnittlig år som kunde



Figur 25 – Sannsynlighet for å slutte fordelt på år som kunde

9.4. Menn har større sannsynlighet for å slutte

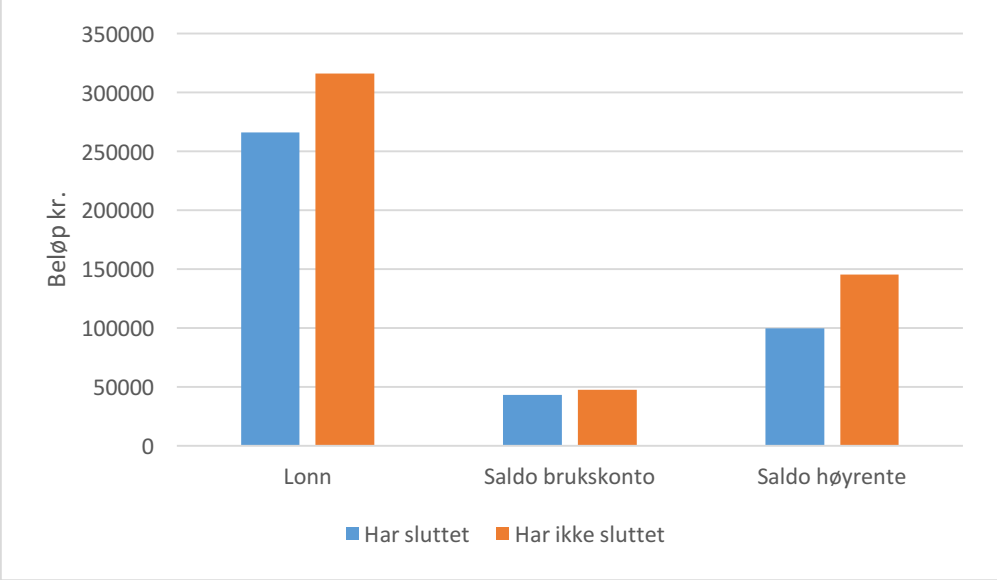
Når det gjelder hypotesen om at menn har større sannsynlighet for å slutte kan vi ikke underbygge dette med våre funn. Som vi kan se i figur 26 er det tilnærmet ingen forskjell mellom sannsynligheten for at en mann skal slutte sammenlignet med en kvinne. I og med at kjønn er en kategorisk variabel benytter vi i dette tilfelle Pearson´s chi-squared test. Testen viser at vi er nødt til å beholde nullhypotesen, noe som betyr at det er ingen forskjell mellom mann og kvinne. Van Del Poel (2004) skrev at menn hadde høyere sannsynlighet for å slutte. En mulig forklaring på forskjellene er at Van Del Poel så på kunder så mye som 77 år tilbake i tid hvor det gjerne var mer vanlig at menn hadde kontroll på økonomien.



Figur 26 – Sannsynlighet for å slutte fordelt på kjønn

9.5. Høyere inntekt reduserer risikoen for å slutte

Når det gjelder hypotesen om at høyere inntekt reduserer risikoen for å slutte kan vi underbygge dette med funnene i dataene, vist i figur 27. De som ikke har sluttet har en gjennomsnittlig lønn på 50 000 mer enn de som har sluttet. Vi undersøkte også om det var noe forskjeller på de som har sluttet og ikke har sluttet når det kommer til saldo på brukskonto og høyrentekonto. På brukskontoen er det liten forskjell mellom de to. Kundene som ikke har sluttet har i gjennomsnittet litt over 10% mer penger på konto. På høyrentekontoen derimot, er det store forskjeller. De som ikke har sluttet har nesten 50% mer penger på konto med 145 000 mot 100 000 kr. T-testen viser at det er signifikant forskjell mellom de to gruppene for lønn og saldo høyrente, mens saldo brukskonto ikke har en signifikant forskjell.

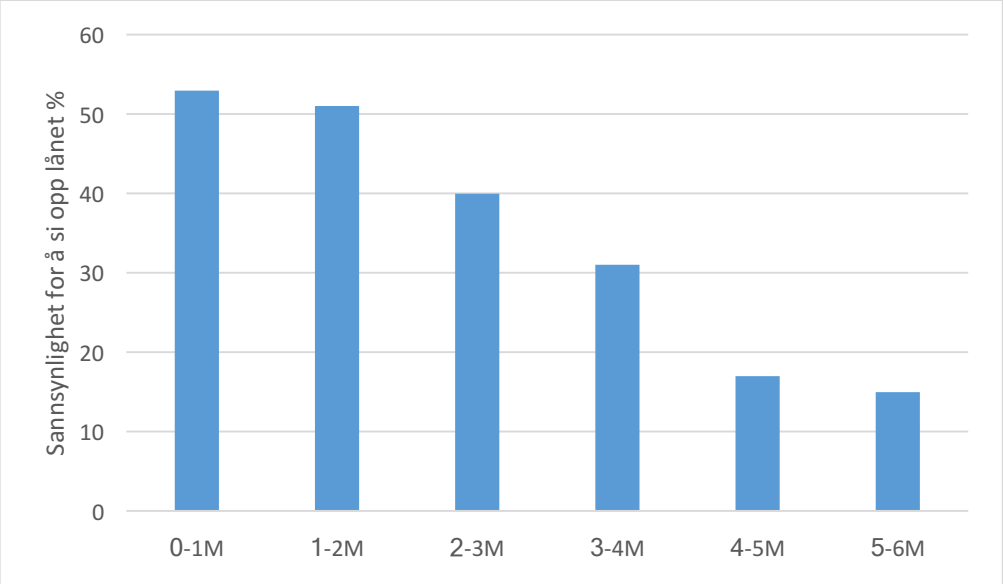


Figur 27 – Inntekt og saldo

9.6. Saldo på boliglån

Hvordan saldoen på boliglån påvirker sannsynligheten for å slutte var ikke en av hypotesene, men allikevel interessant å se på. T-testen viser en p-verdi på 0,0001, noe som betyr at vi kan forkaste nullhypotesen og si at det sannsynlig at det er forskjell mellom de som har sluttet og de som ikke har sluttet, og at dette ikke er tilfeldig. Ikke overraskende er det de som har lavest negativ saldo som har størst risiko for å slutte. Funnet kan begrunnes med at en del av lånene i

kategoriene 0-1M er blitt betalt ned og avsluttet på den måten. I figur 28 kan vi se at sannsynligheten synker fra 1-2M helt ned til 5-6M. Nedgangen kan underbygge det Van Del Poel (2004) skriver om at de som bor i områder med høyere status og har høyere inntekt har lavere risiko for å slutte. Om en har et stort boliglån, har man gjerne en tilsvarende inntekt og bolig. Den gjennomsnittlige sannsynligheten for at en kunde skal si opp boliglånet sitt for alle kategoriene samlet er 50%. Dette indikerer at et stort flertall av kundene er i kategoriene 0-1M og 1-2M, da det kun er disse to som trekker gjennomsnittet opp mot 50%.

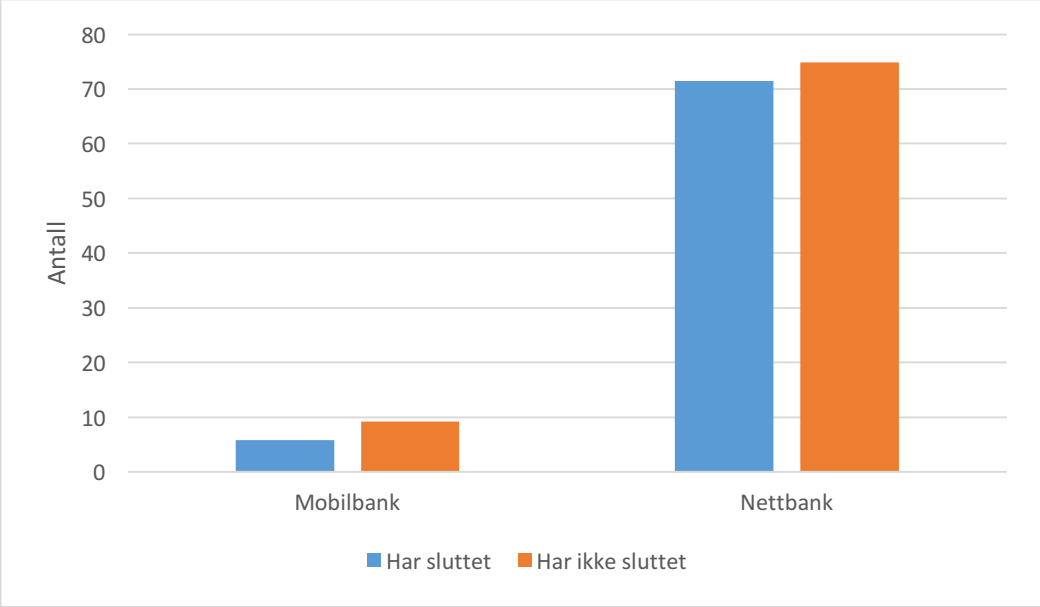


Figur 28 – Sannsynlighet for å slutte fordelt på saldo boliglån

9.7. Spesifikke attributter

I det tidligere arbeidet var det litt varierende funn om at spesifikke attributter når det gjelder bruksmønster hadde noe å si på om man slutter eller ikke. Keramati mfl. (2016) fant at transaksjoner på mobilen var en predikerende attributt, mens Van del Poel (2004) skriver at bruken av spesifikke produkter ikke har noe å si. Vi har ikke mulighet til å undersøke transaksjoner på mobil, og velger derfor å se på innlogginger i mobilbank. De som ikke har sluttet har flere innlogginger i mobilbank, men samtidig er gjennomsnittet for begge to relativt lavt sammenlignet med innlogginger i nettbank. Forskjellen mellom de som har sluttet og de som ikke har sluttet (vist i figur 29), er omtrent tre innlogginger både for nettbank og mobilbank.

T-testen viser at det er signifikant forskjell mellom de som har sluttet og de som ikke har sluttet når det kommer til mobilbank, mens nettbank ikke har en signifikant forskjell.

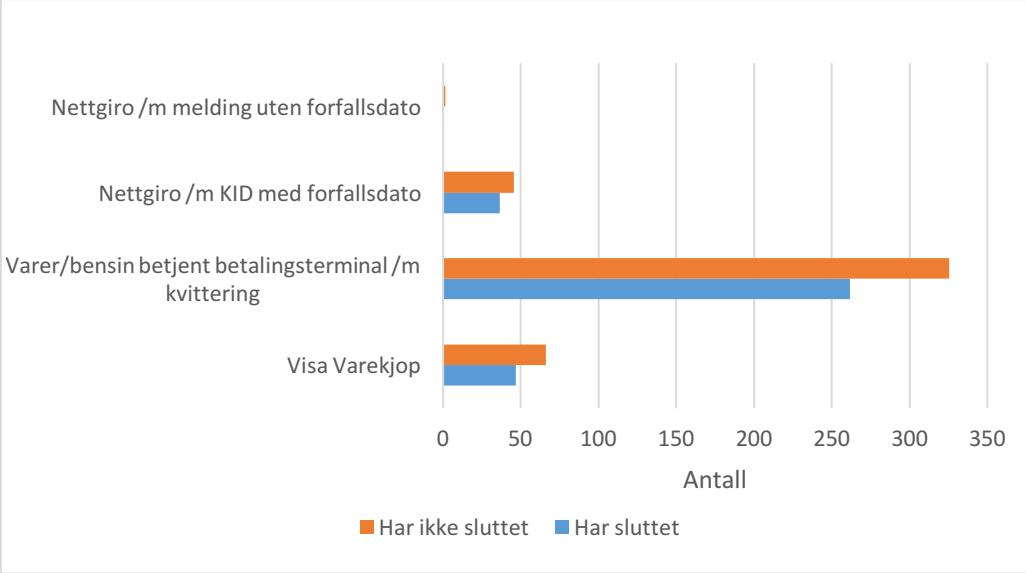


Figur 29 – Antall innlogginger

Ser vi på de mest vanlige transaksjonstypene i figur 30 er det større forskjeller. Vi kan se at de som ikke har sluttet har flere transaksjoner i alle attributtene, hvor Visa Varekjop, Varer/besin betjent betalingsterminal /m kvittering og Nettgiro /m melding uten forfallsdato har de største forskjellene. T-testen viser at det er signifikant forskjell i gjennomsnittet for de som har sluttet og de som ikke har sluttet for alle de fire transaksjonstypene. Hvis vi grupperer Varer/bensin betjent betalingsterminal /m kvittering i ti kategorier ut ifra antall, kan vi tydelig se at den minste gruppen som har færre enn 154 transaksjoner har betydelig høyere sannsynlighet for å si opp lånet med 62% mot det totale gjennomsnittet på 50%. Når det gjelder Nettgiro /m melding uten forfallsdato er det store forskjell. De som ikke har sluttet har i gjennomsnitt 1,6 slike transaksjoner, mens de som har sluttet har 0,4. Ser vi på sannsynligheten for at en kunde skal slutte ut ifra antall Nettgiro /m melding uten forfallsdato så ser vi at det er 64% sannsynlighet for at en kunde som har 0 transaksjoner av denne typen skal slutte. Dersom en kunde har en eller flere transaksjoner synker sannsynligheten betydelig til under 35%.

Hvis vi ser på antall innlogginger og antall transaksjoner samlet kan vi se at de som ikke har sluttet har høyere antall innenfor alle kategoriene, noe som kan tilsi at de som bruker banken mer har mindre sannsynlighet for å slutte. Clapp (2009) skrev at i et marked hvor det stadig ble

vanligere med flere bankforhold så er det blitt viktigere å være primærbanken for en kunde, da en kunde har lavere sannsynlighet for å bytte primærbank. I og med at de som ikke har sluttet bruker banken mer enn de som har sluttet kan vi trekke tråder mot at det er en større andel av de som ikke har sluttet har Skandiabanken som primærbank sammenlignet med de som har sluttet.



Figur 30 – Antall transaksjoner

10. Diskusjon

10.1. Modellering

Innledningsvis startet vi med å anvende det selvorganiserende nettverket Kohonen for å visualisere dataene. Nettverket var i stand til å gruppere dataene, men når vi plottet fasit (churn/ikke-churn) over nettverket sin struktur kunne vi se at grupperingene som nettverket hadde dannet var *ikke* basert på fasiten vi satt med, men heller noen andre attributter. Dette betyr at algoritmen ikke fant noen tydelig fellestrekk eller forskjeller mellom de som har sagt opp boliglånet og de som ikke har sagt opp boliglånet. I og med at algoritmen ikke klarte å organisere dataene i de gruppene vi ønsket kan dette være en indikator på at det vil være utfordrende å utvikle en modell med høy presisjon (>80%).

Etter hvert som vi beveget oss mot ensemble metodene så vi raskt den prediktive styrken i disse modellene. RF som har vist seg å være en kraftig algoritme nådde raskt en presisjon på 73%.

En stor fordel med trebaserte algoritmer i kontrast til for eksempel et nevrale nettverk, er at de tilbyr funksjonalitet for attributtrangering med hensyn på innflytelse på resultatet. Problemstillingen vår var ikke bare å kunne klassifisere kundene, men videre identifisere hvilke trekk som kjennetegner kundene som slutter. RF la stor vekt på NettGiro m/melding uten forfallsdato og ArSomKunde. XGBoost mente også at ArSomKunde var viktig, men resultatet viste at den så på nærmest alle attributtene som ganske så innflytelsesrike med hensyn på utfallet av klassifiseringen.

Ved å fjerne attributter som modellen anser som mindre viktig er det to mulig utfall. Det første utfallet er at modellen mister presisjon, mens det andre utfallet er at andre attributter får større innflytelse. Vi ser at eksempelvis XGBoost, som var den modellen med høyst presisjon, mente at nesten alle attributtene hadde lik innflytelse på utfallet, og da ingen enkelt attributt med høyere enn 6,5%. Videre ser vi fra figur 19 at modellen har en presisjon på nesten 74% med bare 11-12 attributter, og at det er først ved å kun anvende de ni attributtene som modellene anser å ha mest innflytelse på resultatet at modellen sin presisjon er lavere enn 70%. I motsetning klarte ikke RF å kompensere for at vi fjernet mindre viktige attributter og mistet isteden presisjonen.

10.2. Innsikt

Ved å sammenligne attributter for kunder som har sluttet mot de som ikke har sluttet har vi i

analysen funnet flere trekk som kjennetegner de som slutter. Den første faktoren vi fant er alderen til kunden. Her ser vi at vi har en gruppe i alderen opp til 40 år som har større sannsynlighet for å si opp. Videre er antall år som kunde en annen faktor hvor vi finner grupper som er mer sannsynlig til å si opp boliglån. Den første gruppen som har vært kunde i 0-5 år er ekstremt kritisk. Her sier 100% av kundene i 0-4 opp boliglånet sitt, mens de som har vært kunde i fem år har 77% sannsynlighet for å si opp. Den andre gruppen er de som har vært kunde i 8-10 år. Denne gruppen er ikke like kritisk, men har fortsatt over 60% sannsynlighet til å si opp lånet.

Et tredje trekk som kjennetegner de som slutter er velstand. Vi kan se at de som slutter både tjener mindre penger og har mindre penger oppspart fordelt på brukskonto og høyrentekonto. Størrelsen på boliglånet viser det samme. Jo høyere boliglån en kunde har, jo mindre sannsynlig er det at kunden sier opp. Med dagens renter er det slik at det er gunstig å ha boliglån, noe som betyr at det er vanlig å låne opp mot det man kan håndtere. Det er derfor naturlig å anta at de som har stort boliglån også tjener mer penger og har høyere velstand enn de med et lavere boliglån.

Det var ikke noen store forskjeller mellom de som har sluttet og de som ikke har sluttet når det gjelder spesifikke attributter rundt innlogging og bruk. Størst var forskjellen på antall transaksjoner innenfor Varer/bensin betjent betalingsterminal /m kvittering. Selv om det er tydelig forskjell i gjennomsnittet er det ingen tydelige grupper. Hvis vi ser på alle attributtene for innlogging og bruk, ser vi at de som ikke har sluttet har høyere gjennomsnitt på alle attributtene. Vi kan derfor si at de som sier opp lånet bruker banken jevnt over mindre enn de som ikke sier opp lånet.

Ut ifra innsikten kan vi foreslå flere strategier for å forbedre kundebevaringen. Den første går ut på å ta i bruk insentiver for å øke kundenes bruk av banken slik som Clapp (2009) skrev. Tiltaket er for å få kundene til å ha Skandiabanken som primærbank, noe som skal redusere sannsynligheten for å si opp boliglånet.

Årsaken til at kunder som har vært kunder lenge har mindre sannsynlighet for å si opp kan være kundeforholdet. Når man har vært kunde over lengre tid har en fått bygget opp et godt kundeforhold til banken, noe som gjør det vanskeligere å bytte. Ettersom det er stor sannsynlighet for at de som har vært kunder i under fem år sier opp lånet sitt bør det fra bankens side være en prioritet å investere i å tidlig bygge et godt kundeforhold.

At de mindre velstående har større sannsynlighet for å si opp lånet kan være en følge av prissensitivitet eller at andre banker er villig til å strekke seg lengre når de skal ha nytt lån. I og med at Skandiabanken bruker automatiserte løsninger for å innvilge lån er det mulig at andre banker er mer raus når de tildeler boliglån. For yngre kunder er det gjerne ikke det billigste lånet, men det største lånet som er mest attraktivt.

I denne studien har vi ikke undersøkt hvorfor kundene som slutter har de trekkene de har, noe som vil være et godt utgangspunkt for videre arbeid. Ved å undersøke hvorfor, kan en utvikle banken slik at den passer bedre og øker lojaliteten til gruppene som viser seg å være mer utsatt for å si opp lånet.

Attributtene som modellene har ansett som sentrale kan i en markedsføringssammenheng være utfordrende å ta tak i, spesielt hvis antall attributter er 70. Modellene plasserer en kunde i en klasse uten å fortelle hvorfor denne kunden hører til i denne klassen. Å identifisere årsakene til hva som har ført til at en kunde er blitt plassert i en klasse er en stor oppgave. Årsaken kan enten være én enkelt attributts verdi, men som oftest en kombinasjon av mange attributter, kanskje alle 70. Det er mulig å illustrere beslutningstrærne hver for seg slik at man kan se beslutningsgrunnlaget som er tatt, men i og med at det blir brukt et stort antall trær er det vanskelig å finne frem til det beste treet. Intensjonen til slike algoritmer er ikke å gi best mulig innsikt, men best mulig prediksjon.

Ettersom en modell ikke forteller oss hvorfor en kunde har havnet i en klasse kan ikke modellen brukes aktivt for å spisse markedsføringstiltakene mot enkelt kunder, men kun klassifisering. Hvis man ønsker å tilpasse en strategi opp mot en enkelt kunde, må dataene for denne enkelt kunden hentes ut for å så bli sammenlignet med innsikten fra analysen. På denne måten vil man kunne se hvilke attributter for kunden som passer inn i gruppen som sier opp. Eksempelvis blir det hentet ut en kunde på 26 år som har vært kunde i tre år. Ser vi på figur 23 så ser vi at de som er 26 år har 77% sannsynlighet for å si opp, mens figur 25 viser at de som har vært kunde i tre år har 100% sannsynlighet for å si opp lånet. Med denne informasjonen kan man tilpasse strategien for å bevare denne enkeltkunden etter både alderen og hvor lenge kundeforholdet har vart.

Når modellene anser alle attributtene som sentrale kan det være vanskelig å ta dette i bruk i en markedsføringssammenheng. Spesielt når antall attributter er 70. Enkelte attributter er også vanskelig å gjøre noe med, som for eksempel NettGiro m/melding uten forfallsdato. Hvordan

denne kan brukes til å utforme tiltak for å bevare kunder er ikke lett å si. For å redusere antallet attributter en undersøker og baserer markedsføringstiltakene på kan en inngå et kompromiss mellom presisjon på modellen og antall attributter. Dette gjøres ved å basere markedsføringstiltakene på analysen av resultatet til modellen med lavere presisjon og få attributter, mens predikeringen blir gjort med mange attributter og høyere presisjon. I en forretningskontekst vil det intuitivt være enklere å forholde seg til få attributter sammenlignet med mange, og den praktiske verdien av å skulle forholde seg til 10 attributter sammenlignet med 70 er enorm.

Største delen av tiden vi har brukt på denne oppgaven har vi brukt på preprosessering av dataene. Dersom en skal bruke teknologien vi har brukt i oppgaven i en større forretningssammenheng bør en investere i en bedre infrastruktur, slik at flere av de manuelle stegene vi gjorde innledningsvis (eksempelvis å fjerne ubetydelige attributter), kan bli tatt ut av prosessen. Gitt at man har et ferdigprosessert datasett (om dette er en investering i infrastruktur rundt innhenting av dataene, eller en automatisert prosess av stegene vi har utført manuelt er utover oppgavens omfang), kunne man anvendt modellene vi har utviklet på en eksisterende kundebase, for å så slå seg til ro med at de kundene som modellen plasserer i riktig klasse er korrekt 77 % av tilfellene.

Om 77% er en god nok klassifiseringsrate må evalueres av banken. Da må man regne på om kostnadene ved implementering og utrulling av modellen kan rettferdiggjøres med gevinstene den bringer inn. Det er flere momenter som må inkluderes. Kostnadene ved å bevare en kunde som feilaktig har blitt plassert i gruppen som skal slutte, sannsynligheten for at en klarer å bevare en kunde som en når ut til og hvor mye en tjener på å bevare en kunde. Hvor mye en tjener på å bevare en kunde vil variere fra kunde til kunde og enkelte kunder vil ha liten verdi for banken å beholde, selv om modellen sier at en bør bevare denne kunden. Om en skal ta med kundene som feilaktig blir plassert i gruppen som ikke skal slutte kan diskuteres. Ettersom alternativet er å ikke bruke en prediksjonsmodell så kan en anta at denne kunden ville ha sluttet uansett, noe som betyr at det ikke påløper noen ekstra kostnader for kundene som feilaktig blir plassert i gruppen som ikke slutter.

Under eksperimentene våre opplevde vi å få rundt 15% falske positive. Altså, 15% av kundene som vi har definert som ikke har churnet, ble plassert i klassen churn. Det var også rundt 10% falske negative hvor de som har churnet ble plassert i klassen ikke-churn. I og med at de falske positive vil resultere i en unødvendig kostnad dersom banken investerer i å beholde disse

kundene kan det være en idé å justere modellen slik at en oppnår høyest mulig sanne positive resultater. En slik justering kan gå på bekostning av den totale presisjonen til modellen, men kan være lønnsom dersom man med stor sikkerhet kan si at de som er klassifisert som churn er riktig.

I og med at vi hadde tilgang til så mye data hadde vi håpet på høyere presisjon på predikeringen. Hvorfor vi ikke har oppnådd høyere presisjon kan kanskje begrunnes med at vi har hatt tilgang til en stor mengde data rundt bruksmønster og transaksjoner, men lite når det kommer til menneskene bak. Attributter som utdanning, arbeidsgiver, sivilstatus og kundetilfredshet kan potensielt øke presisjonen. Disse attributtene er også lettere å ta i bruk når det kommer til segmentering, sammenlignet med for eksempel antall transaksjoner med NettGiro.

11. Konklusjon

I denne studien har vi sett på hvordan vi kan forbedre kundebevaring ved hjelp av data mining og prediksjonsmodellering av bankkunder med boliglån. Problemstillingen delte vi inn i to underproblemer hvor det første var hva som kjennetegner kundene som sier opp lånet. Vi har gjort flere interessante funn. Kundene som slutter kjennetegnes ved at de er under 40 år og har vært kunde i banken i inntil 5 eller 8-10 år. Videre bruker kundene banken noe mindre enn de som ikke slutter, fordelt på innlogging i nett- og mobilbank og antall transaksjoner av forskjellige typer. Det siste som kjennetegner de som slutter er at de har lavere velstand. De både tjener mindre, har mindre penger på brukskonto og høyrentekonto og lavere boliglån.

Det andre delproblemet tok for seg hvordan vi kan predikere kunder som skal si opp boliglånet sitt**.** Her undersøkte vi flere modeller med varierende resultat. Det første vi prøvde var å ta i bruk en selvorganiserende algoritme for å visualisere potensielle grupperinger i datasettet. Denne algoritmen delte settet inn i fire grupper uten å ta hensyn til om kunden hadde sagt opp lånet eller ikke. Videre forsøkte vi med flere algoritmer for å klassifisere om kunden hadde sagt opp lånet eller ikke sagt opp lånet. De to som pekte seg ut som de beste algoritmene var Random Forest med 74% og XGBoost og 77%. Ved å kjøre attributt seleksjon på disse to algoritmene fikk vi undersøkt hvilke attributter som hadde størst innflytelse på klassifiseringen. Mens attributtene hadde jevn fordeling av innflytelsene i XGBoost hadde Random Forest enkelte attributter med ekstremt høy innflytelse og et stort antall attributter med liten innflytelse. Denne forskjellen gjør at man kan redusere antall attributter i XGBoost uten å miste noe særlig presisjon, noe som kan være hensiktsmessig om man ønsker mer forretningsinnsikt og en mer forståelig modell. På bakgrunn av dette vil vi si at den beste modellen for å predikere kundene som kommer til å si opp lånet er XGBoost.

Ved å kombinere løsningene på de to delprobleme kan en forbedre kundebevaringen ved å identifisere kunder med risiko for å si opp lånet før de har inngått en avtale med en konkurrent. Innsikten vi har oppnådd gjennom studien kan brukes til å utforme strategier både til hvordan en håndterer enkelt kunder, men også gruppene vi har funnet som mer sannsynlig til å slutte. I og med at presisjonen på predikeringen er rundt 77% innebærer dette at modellen predikerer feil på nesten en fjerdedel av kundene. Dette betyr at enkelte kunder som er sannsynlig at slutter vil bli klassifisert som en kunde som ikke er sannsynlig at slutter og motsatt. Det vil derfor være viktig for banken å kalkulere de økonomiske gevinstene av en strategi brukt sammen med

modellen. En kunde som har liten sannsynlighet for å slutte, men som blir klassifisert som en som er sannsynlig at slutter vil bli en unødvendig kostnad for banken om en investerer penger for å beholde denne kunden.

Helt til slutt vil vi understreke viktigheten av å se på data mining som en helhetlig investering. Analysen og predikeringen blir aldri bedre enn dataene man har tilgjengelig. Vi har brukt ekstremt mye tid på å prosessere dataene, noe som kunne vært unngått om man var bevisst på at dataene skulle brukes til data mining når man designet databasene. Ved å se på det som en helhetlig investering kan man både se til at man lagrer data som anses som viktig for å løse framtidige forretningsproblem, samtidig som man kan lagre det i et format som effektivt kan brukes i data mining.

12. Referanser

Analytics Vidhya (2016) *A Complete Tutorial on Tree Based Modeling from Scratch (in R & Pyton)* [Internett] Tilgjengelig fra: <https://www.analyticsvidhya.com/blog/2016/04/complete-tutorial-tree-based-modeling-scratch-in-python/> [Lest: 10. Mars 2017]

Breiman, L. (1996) *Bagging predictors.* Machine learning 24.2, 123-140

Brynjolfsson, E., Hitt, L.M. og Kim, H.H. (2011) *Strength in Numbers: How Does Data-Driven Decisionmaking Affect Firm Performanc*e? SSRN

Clapp, B. (2009) *Retention and attrition: Getting it right... Finally!* ABA Bank Marketing, Vol.41 (2), 48 (1)

Finans Norge/Kantar TNS (2017) *Forbruker og finanstrender 2017* [Internett] Tilgjengelig fra: <https://www.finansnorge.no/aktuelt/sporreundersokelser/forbruker-og-finanstrender/forbruker--og-finanstrender-2017/bytter-boliglansbank-som-aldri-for/> [Lest 20. april 2017]

Grabczewski, K. (2014) *Meta-Learning in Decision Tree Induction*. Springer

Hackeling, G. (2014) *Mastering Machine Learning With Scikit-Learn*. Packt Publishing

Keramatia, A., Ghaneei, H. og Mirmohammadi, S. (2016) *Developing a prediction model for customer churn from electronic banking services using data mining.* Financial Innovation, Vol.2(1), 1-13

Kristensen, T. (1997) *Nevrale Nettverk, fuzzy logikk og genetiske algoritmer*. Cappelen Akademiske Forlag

Lakshminarayanan, B., Roy, D. M. og Teh, Y. W. (2014) *Mondrian Forests: Efficient Online Random Forests,* Advances in Neural Information Processing Systems 27 (NIPS), s. 3140–3148

LeCun, Y. A., Bottou, L., Orr, G. B., og Müller, K. R. (1998) *Efficient backprop* Neural networks: Tricks of the trade. Springer Berlin Heidelberg, 9-48

Mueller A.C. og Guido S. (2016) *Introduction to Machine Learning with Python*. O´Reilley Media Inc.

Nielsen, M. A. (2015) *Neural Networks and Deep Learning*. Determination Press

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., og Vanderplas, J. (2011) *Scikit-learn: Machine learning in Python*. Journal of Machine Learning Research, 2825-2830

Provost, F. og Fawcett, T. (2013) *Data Science for Business*, 1. utg. O´Reilley Media Inc.

Reichheld, F.F. (1993) *Loyalty-based management*. Harvard Business Review, Vol. 71, 64-73

Scikit-Learn [1] *AdaBoost* [Internett] Tilgjengelig fra: <http://scikit-learn.org/stable/modules/ensemble.html#adaboost> [Lest: 2. Mars 2017]

Scikit-Learn [2] *Decision trees* [Internett] Tilgjengelig fra: <http://scikit-learn.org/stable/modules/tree.html> [Lest 19. Mars 2017]

Scikit-Learn [3] *RandomForestClassifier* [Internett] Tilgjengelig fra: <http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html> [Lest: 25. Februar 2017]

Scikit-Learn [4] *AdaBoostClassifier* [Internett] Tilgjengelig fra: <http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html> [Lest: 01. Mars 2017]

Skandiabanken (2015) *Årsrapport 2015* [Internett] Tilgjengelig fra: <http://hugin.info/169938/R/2001838/738659.pdf> [Lest 5. Mars 2017]

Tan P., Steinbach M. og Kumar V. (2005) *Introduction to Data Mining,* 1. utg. Pearson

Tensorflow [Internett] *Tensorflow* Tilgjengelig fra: <https://www.tensorflow.org/> [Lest: 09. Februar 2017]

Van Del Poel, D. og Larivière, B. (2004) *Customer attrition analysis for financial services using proportional hazard models.* European Journal of Operational Research, Vol.157(1), 196-217

Witten, I. H., Frank, E., Hall, M. A. (2011) *Data Mining – Practical Machine Learning – Tools and Techniques*, 3. utg. Elsevier Inc.

Zeiler, M. D., Ranzato, M., Monga, R., Mao, M., Yang, K., Le, Q. V. og Hinton, G. E. (2013) *On rectified linear units for speech processing*. I Acoustics, Speech and Signal Processing (ICASSP), IEEE International Conference, 3517-3521

Zoric, Alisa B. (2016) *Predicting Customer Churn in Banking Industry using Neural Networks.* Interdisciplinary Description of Complex Systems, Vol.14(2), 116-124

XGBoost [Internett] *Introduction to Boosted Trees* Tilgjengelig fra: <http://xgboost.readthedocs.io/en/latest/python/python_api.html#module-xgboost.sklearn> [Lest: 23. Mars 2017]

13. Vedlegg

13.1. Produktbeholdning

Attributt	Beskrivelse
KundeId	Unik nøkkel som identifiserer kunde
År	År 2010-2014
Måned	12
Postnummer	Postnummer
KalenderId	ÅÅÅÅ1231
PositivSaldoAIE	Positiv saldo på brukskontoer
NegativSaldoAIE	Negativ saldo på brukskontoer
PositivAntallAIE	Antall brukskontoer med positiv saldo
NegativAntallAIE	Antall brukskontoer med negativ saldo
PositivSaldoKontokreditt	Positiv saldo kontokreditt
NegativSaldoKontokreditt	Negativ saldo kontokreditt
PositivAntallKontokreditt	Antall kontokreditt med positiv saldo
NegativAntallKontokreditt	Antall kontokreditt med negativ saldo
RammeKontokreditt	Ramme for kontokredittavtale
PositivSaldoBoligkreditt	Positiv saldo boligkreditt
NegativSaldoBoligkreditt	Negativ saldo boligkreditt
PositivAntallBoligkreditt	Antall boligkreditt med positiv saldo
NegativAntallBoligkreditt	Antall boligkreditt med negativ saldo
RammeBoligkreditt	Ramme boligkreditt
PositivSaldoBSU	Positiv saldo BSU
NegativSaldoBSU	Negativ saldo BSU
PositivAntallBSU	Antall BSU med positiv saldo
NegativAntallBSU	Antall BSU med negativ saldo
PositivSaldoHøyrente	Positiv saldo høyrente
NegativSaldoHøyrente	Negativ saldo høyrente
PositivAntallHøyrente	Antall høyrentekonto med positiv saldo
NegativAntallHøyrente	Antall høyrentekonto med negativ saldo
PositivSaldoKredittkort	Positiv saldo kredittkort

NegativSaldoKredittkort	Negativ saldo kredittkort
PositivAntallKredittkort	Antall kredittkort med positiv saldo
NegativAntallKredittkort	Antall kredittkort med negativ saldo
RammeKredittkort	Ramme kredittkort
PositivSaldoPlasseringskonto	Positiv saldo plasseringskonto
NegativSaldoPlasseringskonto	Negativ saldo plasseringskonto
PositivAntallPlasseringsKonto	Antall plasseringskonto med positiv saldo
NegativAntallPlasseringsKonto	Antall plasseringskonto med negativ saldo
RammePlasseringskonto	Ramme plasseringskonto
PositivMarkedsverdiAksje	Positiv markedsverdi på aksjer
NegativMarkedsverdiAksje	Negativ markedsverdi på aksjer
PositivAntallAksjer	Antall aksjer med positiv markedsverdi
NegativAntallAksjer	Antall aksjer med negativ markedsverdi
PositivMarkedsverdiFond	Positiv markedsverdi på fond
NegativMarkedsverdiFond	Negativ markedsverdi på fond
PositivAntallFond	Antall fond med positiv markedsverdi
NegativAntallFond	Antall fond med positiv markedsverdi
PositivSaldoBoliglån	Positiv saldo boliglån
NegativSaldoBoliglån	Negativ saldo boliglån
AktivAntallBoliglån	Antall aktive boliglån
InAktivAntallBoliglån	Antall inaktive boliglån
PositivSaldoBillån	Positiv saldo billån
NegativSaldoBillån	Negativ saldo billån
AktivAntallBillån	Antall aktive billån
InAktivAntallBillån	Antall inaktive billån
AktivRammeBillån	Ramme på aktive billån
InAktivRammeBillån	Ramme på inaktive billån
PositivSaldoUsikretlån	Positiv saldo usikret lån
NegativSaldoUsikretlån	Negativ saldo usikret lån
AktivAntallUsikretlån	Antall aktive usikret lån
InaktivAntallUsikretlån	Antall inaktive usikret lån
AktivRammeUsikretlån	Ramme på aktiv usikret lån

InaktivRammeUsikretlån	Ramme på inaktiv usikret lån
PositivSaldoBrukslån	Positiv saldo brukslån
NegativSaldoBrukslån	Negativ saldo brukslån
AktivAntallBrukslån	Antall aktive brukslån
InAktivAntallBrukslån	Antall inaktive brukslån
AktivRammeBrukslån	Ramme aktivt brukslån
InAktivRammeBrukslån	Ramme inaktivt brukslån

13.2. Prosessert datasett

KundeId*
Kjonn
AlderIValgtAr
FodselsAr*
ArSomKunde
OpprettetDato*
HarLoggetInnMobil
Ar*
PositivSaldoAIE
PositivAntallAIE
SaldoKontokreditt
RammeKontokreditt
SaldoBoligkreditt
RammeBoligkreditt
PositivSaldoBSU
PositivSaldoHoyrente
NegativSaldoKredittkort
RammeKredittkort
SaldoPlasseringskonto
PositivMarkedsverdiAksje
PositivMarkedsverdiFond
NegativSaldoBoliglan

AktivAntallBoliglan
InAktivAntallBoliglan
NegativSaldoBillan
AktivAntallBillan
InAktivAntallBillan
BankIdEkstern
BankIdMobilEkstern
BankIdMobilSKB
BankIdSKB
Certificate
CodeCard
Sms
NettBank
Trader
MobilBank
Churn
Avtale
Avtalegiro
Iflg. oppgave
Kreditrente
Minibankuttak utenfor eget konsern
NettGiro m/KID med forfallsdato
NettGiro m/melding med forfallsdato
Overf. til annen konto
Overforsel
Varekjop i betalingsterminal (BAX) m/kontantutbetaling
Varer/bensin betjent betalingsterminal m/kvittering
Varer/bensin ubetjent betalingsterminal u/kvittering
VISA kontantuttak
VISA Varekjop
Gebyr
Igangsatt giro
Skatt

e-Faktura NettBank m/KID
Debetrente bruksdel
e-Faktura AvtaleGiro
Omkostninger
Papirbasert reserve-losning bet.terminal
NettGiro m/melding uten forfallsdato
Debetrente ord. bev.
Giro
Overforsel utland
Verdipapirer engangsfullmakt kreditorbetalt (debet)
Finansinstrument (aksjehandel)
Oppdrag
Overforsel fra utlandet
Kontantuttak i betalingsterminal (BAX) (Ikke kombinert med varekjop)
Verdipapirer engangsfullmakt (debet)
Bedriftsterminal overforsel
Oppretting retur
Lonn
Barnetrygd
Pensjon

13.3. Slettede attributter

Navn	**Dominerende verdi**	**Antall**
SmartOTP	0	4789
Ukjent	0	4780
Lettbank	0	4785
Preprod	0	4782
Kreditorbetalt Avtalegiro	0	4748
Innskudd	0	4755
Tinglysning	0	4768
Kontofon	0	4756
Innbetaling	0	4770

Tilbakeførsel	0	4776
Uttak	0	4784
Pris	0	4767
Verdipapir	0	4784
Lån	0	4768
Utbetaling	0	4783
Sjekk Interbankgebyr	0	4781
Trygd/Stønad	0	4786
Bankremisse	0	4784
Postering på blokkert/sperret konto	0	4788
Valuta	0	4787
Landbruk	0	4788
Autogiro, kred.bet.	0	4788