Boligverdimodell

[Torje Gloppholm Sylta / ML-gruppe 19], [17.11.2022]

# **BESKRIV PROBLEMET**

## **SCOPE**

Målet med prosjekter er å lage en modell som kan gjette prisen på et boligsalg i Ames, Iowa og deretter opprette en nettside hvor en bruker kan gi denne modellen en del inputs og få returnert en pris. Dette prosjektet kan bidra til å gjøre det lettere og raskere for både privatpersoner og meglere i Ames, Iowa, å finne ut cirka hvilken pris en bolig er verdt. Hvis en privatperson vurderer å selge boligen sin kan de bruke nettsiden til å finne ut prisen på boligen sin før de kontakter en megler. I Norge f.eks. gir noen meglere gratis prisantydning av boligen din, men noen andre tar betalt for det. Denne prisen ligger som regel mellom 2000 til 6000 kroner og en privatperson vil derfor kunne fått spart en del penger ved å bruke nettsiden istedenfor. En annen måte en privatperson kunne ha bruk for det er at de kan bruke nettsiden til å finne ut hvor mye drømmeboligen deres ville koste. En megler vil få bruk for nettsiden ved at det kan redusere arbeidsmengden og kanskje føre til en mer objektiv prisantydning. Folk som bygger boliger, kan også få bruk for nettsiden ved at de kan finne ut hvordan en bolig bør være for å få størst mulig salgspris.

Fant noen lignende nettsider for boligsalg i Norge. To av disse er DNB Eiendoms sin nettside og den andre er Sol eiendom sin. Med DNB Eiendom sin fikk jeg problemer til å gi noen pris og Sol eiendom ga bare salg over tidligere solgte boliger i nærheten. Hvis man hadde hatt data om boligsalg i Norge hadde det vært gode muligheter for å opprette en modell og deretter lage en nettside som kunne gi en bedre prisantydning en det de eksisterende nettsidene gjør i dag.

Hvis maskinlæringsmodellen skulle vært en del av et større system som ble brukt av folk, ville det vært noen bruk for noen ulike komponenter:

1. Data om boligsalgene i Ames, Iowa. Denne dataen trengs også å jevnlig oppdateres ved å fjerne data som er for gammel og legge til ny data om de seneste boligsalgene. Dette er viktig for å få et mer nøyaktig estimat av prisen etter som boligmarkedet over tid alltid går oppover. F.eks. har boligprisene i Norge steget med 27% de siste 5 årene
2. En maskinlæringsmodell for estimat av pris for en bolig. Trengs å oppdateres jevnlig ettersom vi får ny data inn og gammel fjernes. Hvilken type maskinlæringsmodell som brukes kan også oppdateres underveis.
3. Utviklerne skal utvikle en maskinlæringsmodell som skal ta inn dataen og gi et estimat over hva prisen kan være. De har ansvar for å oppdatere maskinlæringsmodellen med ny data underveis og de må holde seg oppdater om ny teknologi og kunnskap innenfor maskinlæring slik at de kan oppdatere og forbedre modellen underveis. Trenger også utviklerne til å utvikle, oppdatere og forbedre nettsiden.
4. En nettside hvor folk kan bruke modellen til å få et estimat for prisen til en bolig. Over tid må nettsiden blir oppdatert ettersom modellen kanskje fjerner eller legger til nye attributt. Nettsiden må også oppdatere hvilken versjon av modellen den bruker etter hvert som det blir fjerner og lagt til ny data.
5. Folk til å bruke nettsiden. Dette vil være privatpersoner eller meglere som ønsker å finne ut prisen for en bolig. Nettsiden kan også oppdateres over tid slik at den blir mer brukervennlig.

En tentativ tidslinje for prosjektet:

1. Finne ut hvilket prosjekt jeg vil gjøre
2. Lese meg opp på prosjektet. Endte opp med å bruke dataen fra «Housing Prices Competition for Kaggle Learn Users». Går da gjennom informasjonen om konkurransen og går gjennom noen av prosjektene som andre har gjort
3. Opprette en rapport over prosjektet og skriver på denne underveis gjennom hele prosjektet
4. Oppretter en notebook i jupyter og henter dataen for prosjektet
5. Ser på dataen for å finne ut hvilket attributt det har, om noen av disse har rekker som mangler data og hva slags type de er
6. Utforsker attributtene ved hjelp av grafer og ser etter sammenhenger.
7. Fjerner attributt som har altfor mange rekker som mangler data
8. Legger til data på attributt som har få rekker som mangler data. Brukte den hyppigste verdien til attributtet.
9. Endrer attributt av typen «object» til å bli av typen «int32». Delte originalt attributtene opp i to deler der jeg brukte one hot encoding på den første delen og ordinal encoding på den andre, men endte opp å endre det til å gjøre ordinal på alle ettersom det gjorde jobben på distribusjon delen mindre og enklere.
10. Utforske ulike modeller og bruke grid search til å finne de beste parameterne.
11. Velger den modellen som gir best resultat. Endte opp med å bruke XGBoost.
12. Bruker testsettet på modellen og laster opp resultatet på konkurransen på kaggle. Endte opp med en score på 15207.66545. Dette tilsvarer plasseringen 2307 av totalt 53289 konkurrenter.
13. Lærte om Pycaret på forelesning og prøvde meg på det, men rett før jeg ble ferdig fikk jeg en del problemer som jeg trur skylder hvilken versjon av Python jeg bruker og hvilken versjon noen av bibliotekene er. Prøvde å fikse det, men fikk det ikke til.
14. Utvikle en nettside hvor en bruker kan skrive en del informasjon om en bolig og deretter få hvilken pris modellen tror boligen er verdt. Endte opp med å bruke flask som vi lærte om i en forelesning.

**METRIKKER**

For at prosjektet skal være en suksess, er det to mål som må bli oppnådd:

* Resultatet jeg får når jeg gir modellen noen inputs skal ikke være alt for langt i fra den originale prisen ettersom det ville være dumt å f.eks. få en prisantydning på 100,000$ når boligen faktisk er verdt 200,000$. Bruker konkurransen på Kaggle for å få en realistisk forestilling over hvor godt modellen min kan og bør være. Ser man på ledertavlen på konkurransen ser man at de beste resultatene ligger på rundt 11616.96532 (hvis jeg ser bort ifra de med en score på 0). Går jeg til de som ligger midt på tabellen, ser jeg at scoren ligger rundt 16644. For at resultatet skal være en suksess, bør jeg helst ligge en plass mellom disse to scorene.
* Nettsiden må være brukervennlig å bruke. Det skal være enkelt å legge til informasjon om en bolig og det bør ikke ta lang tid for modellen å returnere resultatet.

Når jeg skal teste de ulike modellene kommer jeg til å bruke «Root-mean-square deviation», for å sjekke hvor god de er. Grunnen til at jeg bruker RMSE er fordi det er konkurransen på Kaggle bruker for å gi score.

# **DATA**

Datasettet som blir brukt under prosjektet er hentet fra Kaggle konkurransen «Housing Prices Competition for Kaggle Learn Users» og det inneholder til sammen 81 attributter. En av disse er en Id attributt, men de resterende 80 inneholder informasjon om en bolig og salget av det. Attributtene beskriver område der boligen er, hvordan tomten er, de ulike materialene som ble brukt på boligen og hvilken kvalitet det er, størrelse på slik som garasje, kjeller og bolig, antall rom, når og hvordan boligen ble solgt, osv. Datasettet inneholder til sammen 1460 boligsalg i fra Ames, Iowa i fra perioden 2006 til 2010.

Noen av disse boligsalgene mangler informasjon på noen av attributtene. Hvis det er et attributt der det er alt for mange boligsalg som mangler informasjon om det, vil jeg fjerne attributtet. Hvis det er et attributt som bare har noen få hussalg som mangler det, vil jeg enten fylle på med den hyppigste verdien, gjennomsnittverdien til attributtet eller rett og slett bare fjerne hussalget fra datasettet. Har også muligheten for å bruke maskinlæringsmodeller som aksepterer at data mangler.

Noen av attributtene er av typen «object». Må her endre typen slik at de er av typen int. Bruker encoding til å gjøre dette. Er to forskjellige typer encoding jeg kan gjøre. Den første er «one hot encoding» og den blir som regel brukt på attributt der de ulike verdiene ikke er større/bedre eller mindre/dårligere en hverandre. Den andre måten er «ordinal encoding» og denne blir som regel brukt på data hvor det er en sammenheng mellom verdiene. Endte til slutt opp med å bruke «ordinal encoding» på alle «object» attributtene selv om ikke alle disse attributtene hadde verdier som hadde en sammenheng mellom hverandre. Grunnen til dette er fordi «one hot encoding» oppretter en kolonne for hver verdi i et attributt, noe som fører til veldig mange kolonner og gjør hele prosessen ved å implementere modellen på en nettside vanskeligere.

Noen maskinlæringsmodeller gjør en bedre jobb på skalert data en på uskalert data. Opprettet derfor en skalert tabell slik at jeg hadde både en uskalert tabell og en skalert tabell. Bruke da den skalerte tabellen på maskinlæringsmodellen «Random forrest regressor» og den uskalerte på «linear regression», «gradient boosting regressor» og «XGBoost regressor».

**MODELLERING**

De første modellene jeg utforsket er «Linear regression»,«Random forrest regressor» og «gradient boosting regressor». Disse modellene brukte jeg i forrige innlevering og hadde allerede erfaring med dem. Etter jeg hadde prøvd ut disse algoritmene viste det seg at det var «gradient boosting regressor» som gav det beste resultatet. Etter dette gikk jeg på konkurransen på Kaggle og så gjennom hva andre hadde brukt. La da merke til at XGBoost var en populær modell. Etter litt Googling fant jeg ut at XGBoost er en modell som veldig ofte vinner konkurranser på Kaggle. Prøvde derfor ut XGBoost og fikk et resultat som var bedre en det jeg hadde fått ved de andre modellene. Forbedret deretter XGBoost ved å bruke grid search for å finne bedre parametere. Kjørte modellen på testsettet og lastet det opp på konkurransen på Kaggle. Fikk en score på 15207.66545 og dette tilsvarer plasseringen 2307 av totalt 53289 konkurrenter.

Prøvde meg også på å bruke Pycaret for å prøve noe nytt. Brukte Pycaret til å finne de beste modellene, deretter tok jeg de fem beste og gjorde hyperparameter tuning av dem. Prøvde å kombinere alle modellene og prøvde å lage en blender. Til slutt sammenlignet jeg alle de ulike modellene og de viste seg at den beste modellen var den beste av de modellene som ikke var innstilt eller kombinert. Evaluerte til slutt denne modellen ved å se på ulike grafer. Fikk ikke prøvd ut modellen på testsettet ettersom at Pycaret sluttet å funke på grunn av hvilken versjon jeg hadde av Python og de ulike bibliotekene. Fikk ikke fikset problemet så endte opp med å bruke XGBoost modellen på nettsiden.

# **DEPLOYMENT**

Modellen skal settes i drift ved bruk av en nettside. En bruker skal kunne komme inn på nettsiden, fylle inn alle input feltene og deretter få ut igjen en estimert pris. For å lage nettsiden brukte jeg flask og gjorde det ganske likt som de vi hadde fått vist på eksemplene vi hadde fått i undervisning.

For å forbedre systemet ville jeg nok sitt på hvilken av de ulike attributtene som er viktige og uviktige. Akkurat nå må brukeren fylle inn 63 attributt, noe som er en del. Hvis jeg da hadde vist hvilken attributter som ikke var nødvendige å ha med så kunne jeg ha fjernet disse.

En annen forbedring ville vært å bruke ny data. Dataen som er brukt under dette prosjektet er hentet fra boligsalg mellom 2006 og 2010. Hvis en person i dag skulle brukt nettsiden for å finne ut prisen på boligen sin ville de fått en langt mindre pris en hva boligen faktisk er verdt. I Norge f.eks. har boligmarkedet steget med 130% siden 2006.

Hvis nettsiden skulle vært i bruk i dag ville det vært viktig at modellen blir jevnlig trent opp på ny data der det blir lagt til nye boligsalg og de eldste boligsalgene blir fjernet. Maskinlæring er et voksende felt så det kan også være nyttig å holde seg oppdatert og prøve ut nye modeller i tilfelle man finner noe som gir bedre resultat.

# **REFERANSER**

* https://www.kaggle.com/competitions/home-data-for-ml-course
* <https://www.meglersmart.no/pris/verdivurdering#:~:text=Gratis%20eller%20lokketilbud%3F,ligger%20langt%20unna%20et%20meglerkontor>.
* <https://www.smartepenger.no/kalkulatorer/873-boligprisutvikling>
* Boligkostnadkalkulator til DNB eiendom: <https://www.samsolgt.no/verdiestimat?utm_source=altombolig&utm_medium=lenke&utm_campaign=verdiestimat>
* Boligkostnadkalkulator til Sol eiendom: <https://sol.no/eiendom>
* Encoding:<https://machinelearningmastery.com/one-hot-encoding-for-categorical-data/>
* <https://machinelearningmastery.com/xgboost-for-regression/>