ML Oblig 2

10/10/22 - Gruppe 6 Sander Braastad, Ole Anders Nødland, Stian Aas Trohaug og Alfred Rangul Ryvarden

BESKRIV PROBLEMET

SCOPE

I dette prosjektet skal vi lage et maskinlærings-program som forutser hvor mye en film kommer til å tjene på Box Office basert på data som inneholder informasjon om filmer som er hentet fra IMDB. Hva er det som gjør at filmer tjener mest? Klarer et maskinlærings-program å forutse hvor mye en film kommer til tjene bedre enn mennesket kun basert på informasjonen om filmen? Dersom programmet klarer dette, kan det være mulig for filmselskap og produsenter å bruke et slikt program for å avgjøre om en film burde bli laget eller ikke. De kan også finne ut eventuelle endringer de burde gjøre med filmprosjektet for at filmen skal gjøre det bedre ved slipp.

Prosjektet blir gjennomført av 4 studenter på fritiden. Workspacet består av GitHub, Google Colab, Google docs og Alfred sin personlige server. Vi bruker Discord til å kommunisere og avtale fysiske gruppemøter HVL campus Bergen.

Tidslinje med milepæler

Uke 1 (10.10 - 16.10): Bli enige om valgt oppgave og sette opp workspace. Satt opp GitHub rep og Google Colab notebook som alle kan jobbe på felles samtidig.
Uke 2 (17.10 - 23.10): Hente dataen og inspecte data.
Uke 3 (24.10 - 30.10): Mer inspecting av data
Uke 4 (31.10 - 6.11): Gjøre klar dataen til trening og starte på trening
Uke 5 (7.11 - 13.11): Finne ut hvilken model vi vil bruke og trene dataen på den. Finetune dataen. Jobbe med deploying av data.
Uke 6 (14.11 - 18.11): Finalize og deploye programmet, fullføre rapport og levere prosjekt.

METRIKKER

Et minimumskrav for modellen bør være å gå over ytelsen til nåverande metoder som oppnår det samme.
For å måle den beste ytelsen ser en på kor bra R-Square (R2) resultat en får. R-Square bør komme nermest muligt 100%, vist modellen har for lav R-Square verdi vil det føre til at brukeren ikke kan stole på resultatet, og da heller ikke ta en avgjørelse ut fra den.

DATA

Traindataen som blir brukt kommer fra et IMDB-datasheet som inneholder id, belongs to collection, genres, budget, homepage, imdb-id, spoken language, original title, overview, popularity, poster path, production companies, production country, release-date, runtime, status, tagline, keywords, cast, crew og revenue. Dataen blir hentet fra et GitHub repository. Vi valgte å droppe følgende data: imdb-id, id, original title, title, overview, poster path, status og tagline. Begrunnelsen til dette er fordi imdb-id, id, original title, title bare er referanse til hvilken film det er og ikke gir oss noe mer å arbeide med. Dropper overview fordi det kun er en tekst som beskriver filmen. Dropper poster-path ettersom at den kun henviser til jpg av filmplakaten til filmen, og ikke gir oss noe å gå videre med. Droppet status fordi i kaggle står det at status på alle filmer skal være “Realeased”, og kun forteller oss at filmen er ute. Dropper tagline fordi 20% av dataen mangler tagline og består kun av lite tekst om filmen.

Vi lager en funksjon som omgjør en bestemt kategori til True eller False bestemt på om det finnes elementer eller ikke i den kategorien. Denne funksjonen bruker vi på homepage og belongs-to-collection for å vise om filmene har elementer i kategoriene.
Lager en funksjon som fjerner filmer med under x i budsjett. Vi har valgt å droppe filmer med under 50,000$ ettersom at dette er en minimum for å lage en film som kan ha kommersiell suksess.
Vi har laget en funksjon som teller antall tilfeller i en kategori for hver film. Denne funksjonen blir brukt på cast, crew, keywords og production-companies for å sjekke hvor mange instanser det er i hver av de kategoriene.
Lager funksjonen oneHotEncode som lager kategorier av valgte verdier og gir verdien 1 eller 0. Eksempel på dette er med sjangere der vi kan gi filmer som har flere sjangere og gi de verdien 1 eller 0.
Vi kom over et problem i dataen der årstallet i filmene kun inneholder de siste to tallene i året, som gjorde at filmer som kom ut før 1969 ble satt som at de kom ut i f.eks. 2068. Dette problemet fikset vi med å lage en funksjon som trakk fra 100 år på filmer som programmet mener kom ut etter 2019. Datasettet er fra 2019 så det er ingen filmer som skal ha høyere årstall enn 2019.
Alle disse funksjonene blir brukt i en funksjon som gjør om fra json data til tallverdier som gjør at modellen vår kan lese verdiene våre.

MODELLERING

Prosjektet bruke “PyCaret” for å trene ulike type modeller for så å sammenligne hvem som er best. Noen viktige modeller å prøva ut er: Random Forest Regressor, Gradient Boosting Regressor og Linear Regression. Et eksempel på *“human level performance”* for inntekten til filmen kan bli skap ved å lage en lineær graf i geogebra som ser på sammenhengen mellom inntekt og budsjett, og ser kor bra denne treffer til dataen.

Finne utav om det er visse parameter som går igjen som skaper dårlige prediksjoner. Grunnen til at det skjer kan ver at det er for få filmer med den verdier som gjer at modellen ikke har funne mønsteret. For å finne ut hvor viktige de ulike kategoriene er, bruker en korrelasjonen mellom hver kategoriene og inntekten (revenue)

DEPLOYMENT

Planen var å deploye og drifte modellen på hjemmeserveren til Alfred, men etter flere dagers arbeid fikk vi ikke dette til å fungere innenfor tidsrammen.

Deploymenten skulle kjøre på en VM hostet på Ubuntu LTS 20.4 med 32gb RAM, 32gb SSD lagringsplass og 32 kjerner (de ikke ble brukt til noe annet, veldig overkill). Men problemene begynte tidlig etter å ha installert junyper på VM’en og importert notebooken vår fra colab. Enviormentet fra DAT158 githuben var ikke egnet til å hoste vårt projekt, noe som tok lang tid å komme frem til. Pycaret var ikke oppdatert for å fungere på python 3.10 som var det som ble satt opp i det enviromentet. Så jeg forsøkte å sette opp et envíorment på python 3.9.10, men dette fungerte heller ikke idet numba pakken ikke ville installeres på denne versjonen.

Etter et nytt oppsett fra skratch med python 3.8.* funket fortsatt ikke pycaret, og etter mye feilsøking kom jeg frem til at pycaret fungerte på 3.8, men trengte en veldig spesifikk versjon av numpy, som trengte en veldig spesiell versjon av numba, som kræsjet med installerte versjonen av scikit. Etter mye reinstallasjoner både i pip via notebooken og via conda enviormentet kom det fortsatt errors, men den lot meg kjøre koden videre til neste steg etter en endring av setup funksjonen fra simple til iterativ imputation.

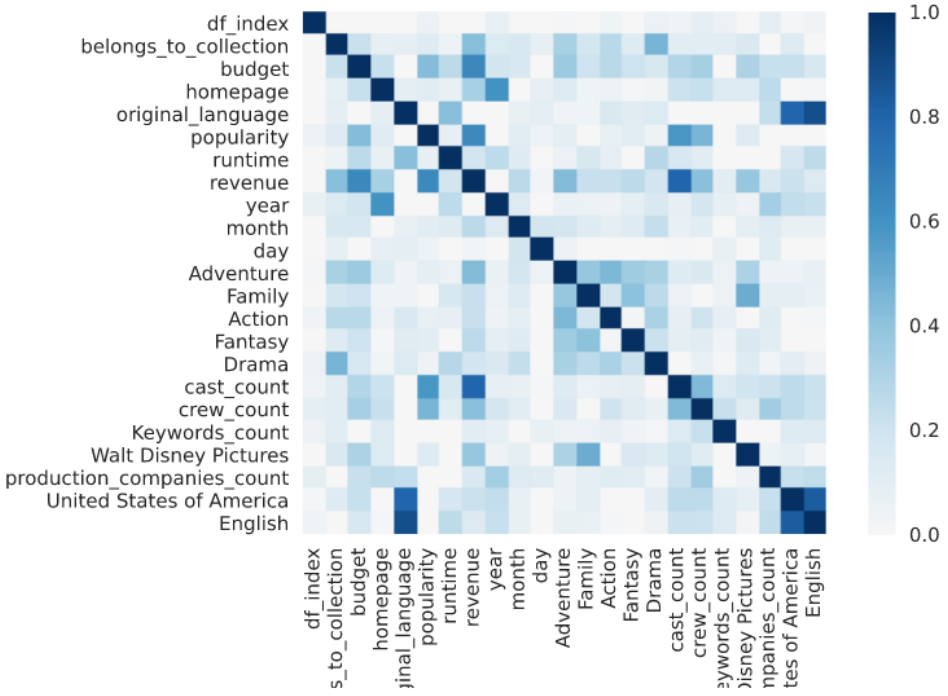
I begynnelsen forsøkte jeg å bruke flask for å sette opp deployment av modellen, men på grunn av ulike import, depencency og kompileringsfeil fokuserte jeg først og fremst på å få til en simpel side via jupyter notebooken. Derfor fokuserte jeg sånn på å få notebooken til å kjøre. Og etter alt det arbeidet over to dager og det er på tide å teste modellen via gradio, så funker det ikke.

```
[79]: import pickle
      pickle.dump(final_model, open('final_model.pkl','wb'))

[80]: #!pip install gradio
      from gradio import *

[93]: #best.get_all_params()
      #best.predict(test_data)

[92]: create_app(final_model)

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In [92], line 1
----> 1 create_app(final_model)

NameError: name 'create_app' is not defined
```

På dette tidspunktet måtte jeg gi opp på notebooken og gå tilbake til flask. Etter å ha laget enda et environment, laste ned ny dependencies og fjernet andre bugs har vi en nesten-ferdig flask webserver med modellen som lastes inn. Men rekkefølgen på input dataen blir feil, og jeg har ikke klart å finne ut av hvordan man finner riktig input rekkefølge. Modellen jeg bruker for å teste med er en catboost modell, og funksjonene som skal returnere forventet input typer fungerer ikke. Så etter to dager med 34 timers arbeid til 7 om morgenen og lite søvn må jeg nesten kaste inn håndkleet. Har gjort så godt jeg kunne men denne kom jeg ikke i mål med dessverre.

KONKLUSJON

Etter å har laget ML-modellen ser en at det er: budsjett og popularitet, har stor betydning for inntekten på en film. Med hjelp av budsjett, popularitet og andre verdier så ende modellen sin R2 skår på litt over 70%. 70%er for dårleg skår til å kunne stole på prediksjonen som modellen gir, men kan vere med å gi en pekepinne på mye en film kommer til å tjene.



Som correlation matrisen viser er det stor sammenheng mellom budsjett og inntekt, som betyr at høyre budsjett bør gi høyere inntekt. Matrisen viser også at det er sammenheng mellom inntekter og popularitet. Det er også en stor sammenheng mellom inntekt og antall skuespillere, men for å ha flere skuespillere må man ha høyere budsjett. Så i bunn og grunn er det budsjettet og populariteten som gir oss mest info om hvor mye filmen kommer til å tjene.

REFERANSER

https://www.wrapbook.com/blog/film-budget-tiers