Challenge Documentation

Predicting Movie Revenue

Thomas van der Molen
S4-AI41

Project Information	
Project members	Thomas van der Molen (4168003)
Project name	Challenge Documentation
Version	1.1

Table of Contents

Version History

Version	Date	Change
1.0	04-05-2022	Created document
1.1	05-05-2022	Fixed Grammar

Prerequisite

This documentation will go over the steps taken during the Challenge: “Predicting Movie Earnings” this challenge will be referenced many times in this document and can be found online or if possible locally.

Challenge Introduction

The goal of this challenge was to try and predict the revenue of a movie using publicly available data. By being able to predict a movie’s revenue before its release, the financial risk that comes along with producing movies can be greatly reduced.

Furthermore, by showing that it is possible to create a good predictor using only publicly available data, it can create more interest into this process and allow larger movie companies to apply and adjust the predictor to their own needs.

Because this project has the goal to predict revenue of movies, the target variable that will be used during predictions will be a movie’s revenue.

From the results of this challenge it was found that it indeed is possible to predict the revenue of movies with a good enough accuracy to be used as a risk lowering tool.
However, it should not be ignored that the predictor does not work flawlessly and can still make errors in its prediction. But on a large enough scale or with extra information to base its predictions on, it can be a good asset to be introduced into the movie production workflow as a risk mitigation strategy.

Project Plan

To get to a functional predictor a couple steps had to be taken, namely: gathering data, observing/cleaning the data and creating a model, these 3 steps will be discussed individually below.

Gathering Data

To gather data for movies, two different publicly available sources have been used to supply data:

TMDb, a community sourced database of movies, series and individuals in this industry (e.g. actors, directors, etc.).
TMDb can be used to gather most information, such as the movie title, budget, revenue, cast, crew, release date and much more, for a detailed list of all data supplied by TMDb, you can go to their developers page.

The-numbers, a news website specialised in the movie industry. The-numbers can supply us with very accurate budget and revenue data on movies. Using a second resource for this will allow the data to be cross referenced, not only increasing the data size but also creating a more accurate and trusted dataset.

The TMDb data can be accessed via their API, in this challenge a custom C# bot is used that records all movies and their actors into two respective csv files. The bot used can be found on GitFront and can be run using its executable (this process can take up to 24 hours).

For the-numbers, a web scraper chrome extension was used, this extension can be added here. The web scraper extension makes use of site-maps to store and share scraper scripts, the one used for this project can be found on Pastebin.

Observing / Cleaning

This phase has been completely documented in the Challenge document referenced during the Prerequisite.

During this part of the process, the data was extensively explored, discovering that most numerical features of the data had a log-normal distribution, which first has to be converted to a normal distribution to see the correlation between a lot of the features.

The features with the best correlation with the target variables are: Budget, Cast/Crew-size, Average Cast popularity, Director popularity and if the movie was part of a Collection/Franchise. Features that did not have a good correlation with revenue however are: Release date, Genre and Runtime.

As for cleaning the data, great caution was taken as to not skew any data seen as movies can have very large ranges in its features such as revenue or budget.
One of the main steps taken while preparing the data, was to cross check the movie entries with known good sources. Because TMDb is community driven, invalid/wrong movies can be added to the dataset and have to be filtered out again.
The dataset will by default also contain documentaries, which is not in the domain of this project.

As for merging the the-numbers dataset with the TMDb dataset, a process of cross checking is used where movies from both datasets will be merged if they contain the same title and the same release date (with a margin of error of a month), if these both match and a field is missing by one of the datasets it will be filled by the other, or will be averaged if both already contained a value.

Modelling

Seen as a numerical labeled value will be predicted using the movies dataset, a form of Regression will be used.

After using multiple different models, (Polynomial) Linear Regression and Support Vector Regression performed the best, with the latter being preferred.
Support Vector Regression allows for a lot of freedom into how the model operates, from the kernel used to represent the data, to how aggresively errors or datapoints will be considered during training.

K-fold Crossvalidation is also used during the modelling phase as to reduce the amount of unexpected varience and allow for better conclusions to be made.

Process

During this process certain aspects were specifically prioritized or considered, some of these have been explicitly explained below.

Data Features

When predicting a movie's revenue, there are many aspects to be considered, such as the corporate influences with Budget, the publics excitement based on marketing and actors and the scale of a production with its crew.

When considering so many factors it is easy to forget the main goal, which is predicting a movie's revenue **before** its release, this means that factors such as reviews, ticket sales, movie exposure and other aspects that mainly grow after the release of a movie should not be considered for the prediction.
Furthermore, great care should also be taken when considering if a feature increases a movie's revenue, or if the movie's revenue has influenced the feature. This can be considered when looking at features such as actor popularity. While it is observed that movies with popular actors sell more tickets, actors also become more popular when they are in popular movies.

Models

When predicting revenue or success, there are many ways to approach this problem, you could for example predict if a movie is successful or not, this would allow for a classification model to be used. However, not every movie or production studio has the same definition of a successful movie. A small studio could make its first movie that gets a million dollars in revenue and be very successful, while most blockbuster movies earn this no matter if it's 'successful'.

Not all movies are also trying to earn as much money as possible, for example a movie that is setting up a larger franchise might not instantly care about the revenue of the one movie, but cares more about the exposure gained for their franchise or company.

Goal

The goal of this project was to show that even with only publicly available data, a model can be created that can reduce the financial risk of producing a movie.

This is done with the goal that perhaps larger production studios see this and would want to explore this further, adding their own private information such as marketing methods or more accurate profit numbers such as per region or even city.
All of these processess could culminate into a system that can be systematically introduced into the movie production workflow as a risk mitigation tool.

Deployment Recommendations

There are many ways to introduce this model into already existing systems, it could be used to show more KPI's to studio leads as a way of greenlighting projects, or it can be introduced into a monitoring system to see how certain marketing techniques or events in the world are possibly affecting an in progress movie.

My recommendation is to host this model as a cloud service, allowing access to it via an API.
I have already taken the first steps in this by deploying the model as a docker container, allowing it to be hosted on any cloud service of choice (such as DigitalOcean, Azure, AWS, etc.) and by adding API endpoints having it be added into already existing monitoring or administration systems used internally.

If there however is no technical team, or there is no budget to deploy such a system, I have also supplied a jupyter notebook, that can train the model using the same data as the challenge and allow for manual predictions to be done.