The Movie Database(TMDb) Analysis and Comparison of Predictive Models

A Preprocessing, Analytical and Modeling Case Study using Supervised ML Models

A Project Presentation

Link for the Colab Notebook:
https://colab.research.google.com/drive/1UTrU1NvRp34KTtOeugscTieXeD9aks07?usp=sharing

Team Members:

Indrashis Paul - 19MIM10046
Om Paras Rajani - 19MIM10099
Aniket Bandyopadhyay - 19MIM10115
Aman Jain - 19MIM10089

Project Guide:

Mr. Ashish Kumar Sahu,
Assistant Professor,
Division of AI and ML,
School of Computer Science and Engineering

Introduction: Motivation and Aim

The movie industry has grown immensely over the past few decades generating approximately $10 billion of revenue for the stakeholders annually. A Movie's gross revenue prediction is a very important problem in the film industry because it determines all the financial decisions made by producers and investors. Furthermore, a prediction system to assess the success of new movies with the help of the predicted revenue can help the movie producers and directors take proper decisions when making the movie in order to increase the chance of profitability and success. Usually, these types of predictions are made using basic statistical techniques that are described below. While these methods are pretty common, they often only provide a very rough and not a specific estimate of revenue and profitability prediction before a film has been released. The goal of this project is to analyze the data and compare two models to check the best computational model through evaluating metrics based on public data for movies extracted from a popular online movie database called The Movie Database (TMDb).

We are using the Linear / Logistic Regression as a baseline model and Random Forest as a comparison check against them.

Although a wide range of supervised learning algorithms are available, each with its strengths and weaknesses, there is no single learning algorithm that works best on all supervised learning problems. So, we are limiting it to only two of them.

Existing Work

Performed by various Data Scientists and Analysts

There has been work done related to certain projects which have referenced the analysis of movie based data sets and made not only predictive models but analyzed these data sets and made a complete analytical model based on their knowledge of Artificial Intelligence & Machine Learning.

Yueming Zhang in 2016 worked on a **Data Mining Algorithm** to analyze which movies are successful and made a **Predictive Model.** Taking TMDB scores as response variable and focus on operating predictions by analyzing the rest of variables in the TMDB 5000 movie data.

GSD in 2017 worked on a model and Explored the data set taken from **TMDb Database** which consisted of movies and in the model calculated what highest popularity as well as highest budget movies were based on the algorithms and calculation only after cleaning the data set and made an analysis based on it.

Implementation : Tools

We will try to solve the Challenges mentioned previously with the help of **Python Programming** Language in a **Jupyter Notebook**.

Python is very popular for **AI, ML and Data Science** projects because of the availability of huge number of importable, open-source libraries pertaining to it.
Jupyter Notebooks is a popular application that enables us to **edit, run and share Python code into a web view**. It allows us to modify and re-execute parts of our code in a very flexible way. That's why Jupyter is a great tool to test and prototype programs, and to easily and interactively visualize our Data.

We will also try to see exactly which **Machine Learning models** we are going to use for our Case Study and examine in detail why one of them works better than the others for a given Dataset.

Methodology: Dataset and Proposed Work

The Dataset (already provided)

The movie dataset on which this case study is based is a database of 5000 movies catalogued by **The Movie Database (TMDb)**.
The information available about each movie is its budget, revenue, rating, actors and actresses, etc.

However, we will be using an unclean dataset for our project.

The Case Study(Problem Statement)

In this case study, we will use this dataset to **analyse** and determine whether any information about a movie can **predict** the **total revenue** of a movie. We will also attempt to predict whether a movie's revenue will exceed its budget (profitability). Also, we will test **two different models** (mentioned previously) for each prediction to check which predicts our target variable better.

Methodology: Module Walkthrough

Novelty of the Project

The novelty of this case study lies in the following factors:

- **Data Wrangling and Mining** techniques are combined with numerous **Visualizations** to gain deeper insight on the dataset.

- **Extensive Comparison Analysis** is done on both the Machine Learning models we are using in order to visualize and understand which of the models work better and the reason for it. To make the comparison more specific and detailed, we stuck to using only three models, unlike ensembling modeling, where multiple models are used but focus is only kept on the Final Score of the model and not a deeper comparison of them.

Real Time Usage of our Case Study

- The **data cleaning and transformation** will help the Movie Analytics Websites to get rid of unnecessary noise in the data they have and help to reproduce a cleaned form of it from which valuable **insights** can be gained.
- The **analysis** and **formation of patterns** in the data will help those Clients to provide **better Movie Analysis** that are usually provided in their various analytics websites. We can also provide with **vivid Visualisations** of all the Movies present in the dataset and present **accurate observations** regarding the movies.
- The **comparison modeling** will help us understand which model works better for a given dataset and also the reason for that. This will help The Film Production Firms, with all the required data present, to **predict** an estimate of the **Revenue** they can generate from a certain movie and **determine** if it will be **Profitable** for them to proceed with the Movie Production or not.

Hardware Requirements

A Desktop or a Laptop with:

- RAM - At least 8GB
- Secondary Memory - At least 1.5GB of Disk Space and 700MB of Virtual Memory
- CPU - Quad core Intel Core i7 Skylake or higher
- Premium graphics cards, for instance GTX 980 or 980MS

Software Requirements

The softwares include:

- Python 3.6 or higher
- Anaconda Jupyter Notebooks for scripting purposes
- Libraries to be imported:
  - Numpy
  - Pandas
  - Matplotlib
  - Seaborn
  - Scikit-learn

The Overall System Architecture

Literature Review

There has been work done related to certain projects which have referenced the analysis of movie-based data sets and made not only predictive models but analyzed these data sets and made a complete analytical model based on their knowledge of Artificial Intelligence & Machine Learning.

Ibtesam Ahmed in May 2020 made a model relating to a **Movie Recommendation System** using data from multiple sources. Recommendation System is a type of information filtering systems as they improve the quality of search results and provides items that are more relevant to the search item.~(Kaggle ID:Ibtesama)

GSD in 2017 worked on a model and Explored the data set taken from **TMDb Database** which consisted of movies and in the model calculated what highest popularity as well as highest budget movies were based on the algorithms and calculation only after cleaning the data set and made an elaborate analysis based on it.~(Kaggle ID:gsdeeppakkumar)

Yueming Zhang in 2016 worked on a **Data Mining Algorithm** to analyze which movies are successful and made a **Predictive Model.** Taking IMDB scores as response variable and focus on operating predictions by analyzing the rest of variables in the IMDB 5000 movie data. The results can help film companies to understand the secret of generating a commercial success movie.~(Kaggle ID:carolezhangdc)

Simonoff ,J. S. and Sparrow, I. R. in 2000 made a model of predicting movie grosses. They took data from the Internet Movie Database (www.imdb.com). Movies that opened on a limited time in 1997 and then released globally in 1998 are included in the sample as well. This yields a total of 311 films. A shortened version of their research paper appeared in *Chance* , 13(3), 15-24 (Summer 2000).

A research by Timothy King about the effects of criticism affecting box office ratings in 2003. In his findings he found out that two of the highest grossing movies(The Lord Of The Rings: Return Of The King and Finding Nemo) also received the highest score of any movie released in 2003. His analysis made use of **Statistical Graphs and Representation** to easily grasp his research. The research paper is available at Research Gate under the title “Does Film Criticism Affect Box Office Ratings”.

A. Chen used **Regression** concepts in his Report and proved Predicting the box-office revenue of a movie before its theatrical release is an important but challenging problem that requires a high level of Artificial Intelligence. Social media has shown its predictive power in various domains, which motivates us to exploit social media content to predict box-office revenues. He employed both Linear and Logistic Regression. ~University of Washington, Seattle, June 2002

Module 1 - Data Preprocessing

Data preprocessing is an important step in the data mining process. The phrase "garbage in, garbage out" is particularly applicable to data mining and machine learning projects. Data-gathering methods are often loosely controlled, resulting in out-of-range values (e.g., Income: −100), impossible data combinations (e.g., Sex: Male, Pregnant: Yes), and missing values, etc. Analyzing data that has not been carefully screened for such problems can produce misleading results. Thus, the representation and quality of data is first and foremost before running any analysis. Often, data preprocessing is the most important phase of a machine learning project, especially in computational biology.

If there is much irrelevant and redundant information present or noisy and unreliable data, then knowledge discovery during the training phase is more difficult. Data preparation and filtering steps can take considerable amount of processing time. Data preprocessing includes cleaning, Instance selection, normalization, transformation, feature extraction and selection, etc. The product of data preprocessing is the final training set.

Data pre-processing may affect the way in which outcomes of the final data processing can be interpreted. This aspect should be carefully considered when interpretation of the results is a key point, such in the multivariate processing of chemical data (chemometrics).

Data Preprocessing - Workflow

A minimized form of our Dataset

	budget	genres	homepage	id	keywords	original_language	original_title	overview	popularity	production_companies	production_countries	release_date	revenue	run
0	237000000	Action, Adventure, Fantasy, Science Fiction	http://www.avatarmovie.com/	19995	culture clash, future, space war, space colony...	en	Avatar	In the 22nd century, a paraplegic Marine is di...	150.437577	Ingenious Film Partners, Twentieth Century Fox...	United States of America, United Kingdom	2009-12-10	2787965087	
1	300000000	Adventure, Fantasy, Action	http://disney.go.com/disneypictures/pirates/	285	ocean, drug abuse, exotic island, east india t...	en	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	139.082615	Walt Disney Pictures, Jerry Bruckheimer Films,...	United States of America	2007-05-19	961000000	
2	245000000	Action, Adventure, Crime	http://www.sonypictures.com/movies/spectre/	206647	spy, based on novel, secret agent, sequel, mi6...	en	Spectre	A cryptic message from Bond's past sends him o...	107.376788	Columbia Pictures, Danjaq, B24	United Kingdom, United States of America	2015-10-26	880674609	
3	250000000	Action, Crime, Drama, Thriller	http://www.thedarkknightrises.com/	49026	dc comics, crime fighter, terrorist, secret id...	en	The Dark Knight Rises	Following the death of District Attorney Harve...	112.312950	Legendary Pictures, Warner Bros., DC Entertain...	United States of America	2012-07-16	1084939099	
4	260000000	Action, Adventure, Science Fiction	http://movies.disney.com/john-carter	49529	based on novel, mars, medallion, space travel,...	en	John Carter	John Carter is a war-weary, former military ca...	43.926995	Walt Disney Pictures	United States of America	2012-03-07	284139100	

Defining Target Variables

Now, we will define the regression and classification outcomes. Specifically, we will use the **revenue** column as the target for regression. For classification, we will construct an indicator of **profitability** for each movie.

Steps

- We will create a new column in our DataFrame called **profitable**, defined as 1 if the movie **revenue** is greater than the movie **budget**, and 0 otherwise.
- Next, we will define and store the outcomes we will use for regression and classification as such
  - **regression_target** as the string 'revenue'.
  - **classification_target** as the string 'profitable'

Removing null, unwanted or infinite values

For simplicity, we will proceed by analyzing only the rows without any missing data. Here, we will first **remove** all unimportant features to reduce high data dimensionality and then remove all the rows with any infinite or missing values.

Steps:

- We check the number of null values or **numpy.nan** values by using **DataFrame.isnull().sum().**
- We use **DataFrame.drop()** to remove any columns that are unimportant.
- We check if the important columns contain value = 0 and replace those values of the budget column with **numpy.nan** since the budget of a movie cannot be 0 even if the revenue might be.
- We create a column which shows the **return** gained or lost from the movie production.
- We use **DataFrame.replace()** to replace any cells from the resultant dataset with type **numpy.inf** or **-numpy.inf** with **numpy.nan**.
- We drop all rows with any **numpy.nan** values in that row using **DataFrame.dropna()**.
- We use the **inplace=True** argument to immediately reflect the changes in the current dataset.

BEFORE

budget	0
genres	28
homepage	3091
id	0
keywords	412
original_language	0
original_title	0
overview	3
popularity	0
production_companies	351
production_countries	174
release_date	1
revenue	0
runtime	2
spoken_languages	87
status	0
tagline	844
title	0
vote_average	0
vote_count	0
movie_id	0
cast	43
profitability	0

dtype: int64

AFTER

budget	0
genres	0
id	0
keywords	0
original_language	0
overview	0
popularity	0
release_date	0
revenue	0
runtime	0
status	0
tagline	0
title	0
vote_average	0
vote_count	0
movie_id	0
cast	0
profitable	0
return	0

dtype: int64

Feature Engineering

Encoding of Genres Column

Many of the variables in our dataframe contain the names of **genre**, **actors/actresses**, and **keywords**. Here, we add indicator columns for each **genre**.

Steps

- We determine all the **genres** in the genre column and make use of the **strip()** function on each genre to remove trailing characters.
- Next, we include each listed genre as a new column in the dataframe. Each element of these genre columns is 1 if the movie belongs to that particular genre, and 0 otherwise, keeping in mind, a movie may belong to several genres at once.
- This process is also called as **One-Hot Encoding** but here it is applied on each data of each row of the Column.

The Encoded Genres Column

	Action	Adventure	Fantasy	Science Fiction	Crime	Drama	Thriller	Animation	Family	Western	Comedy	Romance	Horror	Mystery	History	War	Mus
0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	
1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	
2	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	
3	1	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	
4	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
4788	0	0	0	0	1	0	0	0	0	0	1	0	1	0	0	0	
4791	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	
4792	0	0	0	0	1	0	1	0	0	0	0	0	1	1	0	0	
4796	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0	
4798	1	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	

3266 rows × 20 columns

Creating the Year column for analysis

Some variables in the dataset can be used to create new and important columns which can be further used for better analysis of the dataset

Steps

- Create a new column, **year** where we extract and store the year of production of a movie from the given format of **release_date.**
- After creating the column, we extract the year from the **release_date** using the **pd.to_datetime** function.

```
0    2009-12-10
1    2007-05-19
2    2015-10-26
3    2012-07-16
4    2012-03-07
```

Feature Selection

Some variables in the dataset are already **numeric** and perhaps useful for **regression** and **classification**. Here, we will store the names of these variables for future use. We will also take a look at some of the **continuous variables** and outcomes by plotting each pair in a **scatter plot**. Finally, we will evaluate the **skew** of each variable.

Steps

- We call **plt.show()** to observe the plot shown below to find which of the covariates and/or outcomes are correlated with each other.
- We call **skew()** on the columns **outcomes_and_continuous_covariates** in the dataset and check the features for which the skew is above 1.

Feature Transformation

Here, we will transform these variables to **eliminate** this skewness. Specifically, we will use the **numpy.log10()** method. Because some of these variable values are exactly 0, we will add a <u>small positive value</u> to each to ensure it is defined; this is necessary because log(0) is negative infinity.

Steps

- For each above-mentioned variable in the dataset, we will transform the value x into **numpy.log10(1+x)**.

Storing the Transformed Dataset

We will now save our dataset.

Steps

- We use **DataFrame.to_csv()** to save the DataFrame object as **“movies_clean.csv”**.

Module 2 - Exploratory Data Analysis

In statistics, **exploratory data analysis** is an approach to analyzing data sets to summarize their main characteristics, often with visual methods. A statistical model can be used or not, but primarily EDA is for seeing what the data can tell us beyond the formal modeling or hypothesis testing task. Exploratory data analysis was promoted by John Tukey to encourage statisticians to explore the data, and possibly formulate hypotheses that could lead to new data collection and experiments. EDA is different from initial data analysis (IDA), which focuses more narrowly on checking assumptions required for model fitting and hypothesis testing, and handling missing values and making transformations of variables as needed.

Tukey defined data analysis in 1961 as:

"Procedures for analyzing data, techniques for interpreting the results of such procedures, ways of planning the gathering of data to make its analysis easier, more precise or more accurate, and all the machinery and results of (mathematical) statistics which apply to analyzing data."

EDA - Module Section Workflow

Statistical Analysis

Trend Analysis of **vote_average** of movies

Correlation Analysis

Genre Feature Analysis

Regression Target Analysis

General Feature Trend Analysis

Analysis of Movie Trends over the Year

A minimized form of the Transformed Dataset

	budget	genres	id	keywords	original_language	overview	popularity	release_date	revenue	runtime	status	tagline	title	vote_average	vote_count	movie_id	cast	profitable	return	Action	Adventure
0	8.374748	Action, Adventure, Fantasy, Science Fiction	19995	culture clash, future, space war, space colony...	en	In the 22nd century, a paraplegic Marine is di...	2.180234	2009-12-10	9.445287	2.212188	Released	Enter the World of Pandora.	Avatar	7.2	4.071919	19995	Sam Worthington, Zoe Saldana, Sigourney Weaver...	1	11.763566	1	1
1	8.477121	Adventure, Fantasy, Action	285	ocean, drug abuse, exotic island, east india t...	en	Captain Barbossa, long believed to be dead, ha...	2.146384	2007-05-19	8.982723	2.230449	Released	At the end of the world, the adventure begins.	Pirates of the Caribbean: At World's End	6.9	3.653309	285	Johnny Depp, Orlando Bloom, Keira Knightley, S...	1	3.203333	1	1
2	8.389166	Action, Adventure, Crime	206647	spy, based on novel, secret agent, sequel, mi6...	en	A cryptic message from Bond's past sends him o...	2.034936	2015-10-26	8.944815	2.173186	Released	A Plan No One Escapes	Spectre	6.3	3.650016	206647	Daniel Craig, Christoph Waltz, Léa Seydoux, Ra...	1	3.594590	1	1
3	8.397940	Action, Crime, Drama, Thriller	49026	dc comics, crime fighter, terrorist, secret id...	en	Following the death of District Attorney Harve...	2.054280	2012-07-16	9.035405	2.220108	Released	The Legend Ends	The Dark Knight Rises	7.6	3.959375	49026	Christian Bale, Michael Caine, Gary Oldman, An...	1	4.339756	1	0
4	8.414973	Action, Adventure, Science Fiction	49529	based on novel, mars, medallion, space travel,...	en	John Carter is a war-weary, former military ca...	1.652507	2012-03-07	8.453531	2.123852	Released	Lost in our world, found in another.	John Carter	6.1	3.327359	49529	Taylor Kitsch, Lynn Collins, Samantha Morton, ...	1	1.092843	1	1

Statistical Analysis

Analyzing all the features form a Statistical point of view

	budget	id	popularity	revenue	runtime	vote_average	vote_count	movie_id	profitable
count	3266.000000	3266.000000	3266.000000	3266.000000	3266.000000	3266.000000	3266.000000	3266.000000	3266.000000
mean	7.278334	45488.042254	1.292131	6.855850	2.038060	6.271341	2.604175	45488.042254	0.694121
std	0.740971	77053.912258	0.405928	2.486905	0.082333	0.894677	0.631083	77053.912258	0.460849
min	0.301030	5.000000	0.000688	0.000000	0.000000	0.000000	0.000000	5.000000	0.000000
25%	7.000000	5307.500000	1.041590	7.030228	1.986772	5.700000	2.204120	5307.500000	0.000000
50%	7.397940	11317.000000	1.317127	7.695402	2.029384	6.300000	2.645913	11317.000000	1.000000
75%	7.740363	44698.250000	1.575985	8.146561	2.082785	6.900000	3.042477	44698.250000	1.000000
max	8.579784	417859.000000	2.942792	9.445287	2.530200	8.500000	4.138397	417859.000000	1.000000

Observations:

- It shows the count of all the features as **3266** confirming that the data present in the dataset is constant and doesn't contain any discrepancy.
- It also shows the mean of all the features and we can clearly see that all the numerical features have been normalized to their **log10** counterparts.
- It also shows us the maximum and minimum values present in each feature.

Correlation Analysis

Plotting a heatmap for the correlation of all the features of our Dataset

Observations:

- Most of the features have Pearson's Correlation Coefficient between **0.3** and **0.7** meaning

  that they have some relation between them.

- return feature is solely created for visualization and so is irrelevant to all the features.

Regression Target Analysis

-Plotting the correlation graph between revenue and budget

Observations:

- We can definitely see strong correlation between **budget** and **revenue**
- Quite a few number of outliers are present
- The relation is mostly **linear**

-Plotting the correlation graph between revenue and popularity

Observations:

- Again as expected, we see some correlation between **revenue** and **popularity**
- Here, too, quite a few number of outliers are present
- The relation is mostly **linear**

-Plotting the correlation graph between revenue and vote_count

Observations:

- Strong correlation between **revenue** and **vote_count** is observed.
- The movies that have high revenue necessarily has a high vote_count.
- There are outliers present, although not too many.
- The relation is strongly **linear**.

Movie Return trends over the years

-Sorting out the title, vote_count and year to check the movies with highest vote_counts ever

	title	vote_count	year
96	Inception	4.138397	2010
65	The Dark Knight	4.079290	2008
0	Avatar	4.071919	2009
16	The Avengers	4.071035	2012
788	Deadpool	4.041235	2016
95	Interstellar	4.036150	2014
287	Django Unchained	4.004321	2012
94	Guardians of the Galaxy	3.988693	2014
426	The Hunger Games	3.975707	2012
127	Mad Max: Fury Road	3.974420	2015

	title	vote_count	year	revenue	return
96	Inception	4.138397	2010	8.916734	5.159580
65	The Dark Knight	4.079290	2008	9.001975	5.430046
0	Avatar	4.071919	2009	9.445287	11.763566
16	The Avengers	4.071035	2012	9.181717	6.907081
788	Deadpool	4.041235	2016	8.893824	13.501948
95	Interstellar	4.036150	2014	8.829381	4.091636
287	Django Unchained	4.004321	2012	8.628765	4.253682
94	Guardians of the Galaxy	3.988693	2014	8.888364	4.548992
426	The Hunger Games	3.975707	2012	8.839610	9.216143
127	Mad Max: Fury Road	3.974420	2015	8.578477	2.525722

-Storing the same part of the Dataframe with **revenue** and **return** in a variable

-Storing only the movies with a very high revenue, specifically

revenue > 1,00,00,00,000 or 1 billion

-Plotting High-revenue and high-vote_count movies according to their release year

Observations:

- After Titanic (1997) which was the first movie to reach **1 billion** dollar mark, 2010, 2011, 2012, 2013 and 2015 had many movies crossing the billion dollar mark significantly.
- There has been an increase in the revenue of movies filmed till 2012 which showed the peak revenue generated, and although it decreased sharply for 2013 and 2014 yet, it again increased in 2015

Trend Analysis of vote_average of movies

-Extracting and printing the sorted sub-dataset of title, vote_average, vote_count and year of the movies for which vote_count is greater than 3000

	title	vote_average	vote_count	year
1881	The Shawshank Redemption	8.5	3.914132	1994
3337	The Godfather	8.4	3.770410	1972
3865	Whiplash	8.3	3.628900	2014
2294	Spirited Away	8.3	3.584444	2001
1818	Schindler's List	8.3	3.636488	1993
3232	Pulp Fiction	8.3	3.925776	1994
662	Fight Club	8.3	3.973774	1999
2731	The Godfather: Part II	8.3	3.523616	1974
809	Forrest Gump	8.2	3.899164	1994
690	The Green Mile	8.2	3.607348	1999

Observations:

- These are the 1st 10 movies in rank considering the **vote_average** feature
- We observe that **The Shawshank Redemption (1994)** is the movie with the highest vote_average ever.

-Plotting the correlation graph between vote_average and popularity

Observations:

- Some correlation between **vote_average** and **popularity** is observed.
- The movies that have high vote_average doesn't necessarily have to be highly popular.
- There are outliers present, although not too many.
- The relation is mostly linear.

Genre Feature Analysis

-Extracting the number of of each genre used throughout the total dataset

-Creating a new dataframe to store the genres and their count in a sorted order

	genre	movies
0	Drama	1449
1	Comedy	1140
2	Thriller	984
3	Action	919
4	Adventure	666
5	Romance	576
6	Crime	533
7	Science Fiction	442
8	Horror	378
9	Family	362
10	Fantasy	347
11	Mystery	278
12	Animation	178
13	History	132
14	Music	112
15	War	108
16	Western	57
17	Documentary	30
18	Foreign	4
19	TV Movie	2

-Plotting the first 15 genres and their movie count

Observations:

- Drama is the most used genre in the Film Industry.
- Comedy and Thriller genres follow after it.
- Foreign and TV Movie genres has the lowest movie count (excluded in the graph).

General Movie Trend Analysis

-Checking the most expensive movie ever made

	title	budget	revenue	return	year
17	Pirates of the Caribbean: On Stranger Tides	8.579784	9.019413	2.751878	2011
1	Pirates of the Caribbean: At World's End	8.477121	8.982723	3.203333	2007
7	Avengers: Age of Ultron	8.447158	9.147801	5.019299	2015
4	John Carter	8.414973	8.453531	1.092843	2012
6	Tangled	8.414973	8.772171	2.276134	2010
5	Spider-Man 3	8.411620	8.949815	3.452991	2007
13	The Lone Ranger	8.406540	7.950802	0.350157	2013
19	The Hobbit: The Battle of the Five Armies	8.397940	8.980467	3.824079	2014
3	The Dark Knight Rises	8.397940	9.035405	4.339756	2012
8	Harry Potter and the Half-Blood Prince	8.397940	8.970328	3.735837	2009

Observations:

- **Pirates of the Caribbean: On Stranger Tides**, the Johnny Depp starrer is the most expensive movie ever made! The budget was **3.8 Billion** which is still the highest!

-Checking the highest grossing movies of all time

	title	budget	revenue	year
0	Avatar	8.374748	9.445287	2009
25	Titanic	8.301030	9.266004	1997
16	The Avengers	8.342423	9.181717	2012
28	Jurassic World	8.176091	9.179991	2015
44	Furious 7	8.278754	9.177897	2015
7	Avengers: Age of Ultron	8.447158	9.147801	2015
124	Frozen	8.176091	9.105244	2013
31	Iron Man 3	8.301030	9.084734	2013
546	Minions	7.869232	9.063232	2015
26	Captain America: Civil War	8.397940	9.061944	2016

Observations:

- The mighty **Avatar** was the highest grossing movie of all time, since it generated the highest revenue ever, which was surpassed recently by **The Avengers: End Game**.

-Checking the trend of revenue over the years

Observations:

- We can see that the revenue has been steadily increasing over the years

Module 3- Predictive and Comparison Modelling

Predictive modeling uses statistics to predict outcomes. Most often the event one wants to predict is in the future, but predictive modelling can be applied to any type of unknown event, regardless of when it occurred. For example, predictive models are often used to detect crimes and identify suspects, after the crime has taken place.

Similar to the study conducted in various Papers, we used k-fold cross-validation rather than running a single experiment, specifically 10-fold cross-validation.
As mentioned in further slides, we used two fundamentally different types of predictive models – Basic Statistical Model, Linear and Logistic Regression and Tree-based Ensemble Model, Random Forest Model. Also, we used Correlation (r2_score) for the Regression task and Accuracy (accuracy_score) for the Classification task.

Predictive and Comparison modelling - Workflow

Defining Variables and Instantiating Predictive Models

In this segment, we will instantiate regression and classification models. Code provided prepares the covariates and outcomes we will use for model analysis.

Steps

- Instantiate LinearRegression(), LogisticRegression(), RandomForestRegressor(), and RandomForestClassifier() objects, and assign them to linear_regression, logistic_regression, forest_regression, and forest_classifier, respectively.
- For the random forests models, specify max_depth=4 and random_state=0 initially.

Defining Metrics for scoring our Models

In this segment, we will create two functions that compute a model's score. For regression models, we will use correlation as the score. For classification models, we will use accuracy as the score.

Steps

- Define a function called correlation with arguments estimator, X, and y. The function should compute the correlation between the observed outcome y and the outcome predicted by the model.
  - To obtain predictions, the function should first use the fit method of estimator and then use the predict method from the fitted object.
  - The function should return the first argument from r2_score comparing predictions and y.
- Define a function called accuracy with the same arguments and code, and computing the accuracy_score for comparing predictions and y.

Performance of Baseline Regression Models

In this segment, we will compute the cross-validated performance for the linear and random forest regression models.

Steps

- Call cross_val_score using linear_regression and forest_regression as models. Store the output as linear_regression_scores and forest_regression_scores, respectively.
    - Set the parameters cv=10 to use 10-fold cross-validation and scoring=correlation to use our correlation function defined in the previous segment.
- Plotting code has been provided to compare the performance of the two models. Use plt.show() to plot the correlation between **actual** and **predicted** revenue for each cross-validation fold using the linear and random forest regression models.
- We observe that forest_regression clearly shows a better performance than linear_regression

Performance of Baseline Classification Models

In this segment, we will compute cross-validated performance for the logistic and random forest classification models.

Steps

- Call cross_val_score using logistic_regression and forest_classifier as models. Store the output as logistic_regression_scores and forest_classification_scores, respectively.
    - Set the parameters cv=10 to use 10-fold cross-validation and scoring=accuracy to use our accuracy function defined in the previous exercise.
- Plotting code has been provided to compare the performance of the two models. Use plt.show() to plot the accuracy of predicted profitability for each cross-validation fold using the logistic and random forest classification models.
- We observe that both models performs good for the classification but forest_classifier works a bit better.

Improving Performance of Predictive Models

In the Baseline Regression and Classification Models, we saw that predicting revenue was only moderately successful while predicting profitability was really successful. It might be the case that predicting movies that generated precisely no revenue is difficult. In the next three segments, we will exclude these movies, and rerun the analyses to determine if the fits **improve**. In this segment, we will rerun the regression analysis for this subsetted dataset. Although our target is to improve the regression models, we will still re-train our classification models too for an even better prediction than before.

Steps

- Define positive_revenue_df as the subset of movies in df with revenue greater than zero.
- Code is provided below that creates new instances of model objects. Change the max_depth parameter of RandomForest models from 4 to 20
- to increase the accuracy of our model. Replace all instances of df with positive_revenue_df.

Performance of Optimized Regression Models

In this segment, we will compute the cross-validated performance for the optimized linear and random forest regression models for positive revenue movies only.

Steps

- Call cross_val_score using linear_regression and forest_regression as models. Store the output as linear_regression_scores and forest_regression_scores, respectively.
    - Set the parameters cv=10 to use 10-fold cross-validation and scoring=correlation to use our correlation function defined in the previous exercise.
- Plotting code has been provided to compare the performance of the two models. Use plt.show() to plot the correlation between **actual** and **predicted** revenue for each cross-validation fold using the optimized linear and random forest regression models.
- Here, too, the forest_regression model works better than the linear_regression model. We also find substantial improvement in the performance of our regression models.
- Code is provided that prints the importance of each covariate in predicting revenue using the random forests regressor.
    - We observe that vote_count, budget and popularity are the most important features in **predicting revenue**

Performance of Optimized Classification Models

In this segment, we will compute cross-validated performance for the logistic and random forest classification models for positive revenue movies only.

Steps

- Call cross_val_score using logistic_regression and forest classifier as models. Store the output as logistic_regression_scores and forest_classification_scores, respectively.
    - Set the parameters cv=10 to use 10-fold cross-validation and scoring=accuracy to use our accuracy function defined in the previous segment.
- Plotting code has been provided to compare the performance of the two models. Use plt.show() to plot the accuracy between actual and predicted profitability for each cross-validation fold using the logistic and random forest classification models.
- forest classifier obviously performs far better than logistic_regression. In fact, logistic_regression doesn't improve at all and is stuck at the same range but forest classifier improves to a perfection.
- Code is provided that prints the importance of each covariate in predicting profitability using the random forests classifier.
    - We observe that here too vote_count, popularity and budget are the most important features but the importance of popularity and budget are reversed

RESULTS

Type of Model	Task	Model	Evaluation Metric	Score (Mean)
BASELINE MODEL	1. Regression	Linear Regression	Correlation	0.278
		Random Forest Regressor		0.567
	2. Classification	Logistic Regression	Accuracy	0.85
		Random Forest Classifier		0.932
OPTIMIZED MODEL	1. Regression	Linear Regression	Correlation	0.527
		Random Forest Regressor		0.911
	2. Classification	Logistic Regression	Accuracy	0.85
		Random Forest Classifier		1

DEMO VIDEO

TMDb Movie Analysis and Comp
https://colab.research.google.com/drive/1UTrU1NvRp34KTtOeugscTieXeD9aks07?usp=sharing#scrollTo=...
Learn Intro to Mach...
All You Need to Kn...
Analytics Vidhya De...
Made With ML
Machine Learning P...
Dashboard | Hacker...
Professional ML En...
Machine Learning C...
Verzeo/Machine Le...
TMDb Movie Analysis and Comparative Prediction _ Sem 3 Project .ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
Initializing
Editing
The Movie Database (TMDb) Analysis and Comparison of Predictive Models
The movie dataset on which this case study is based is a database of 5000 movies catalogued by The Movie Database (TMDb). The information available about each movie is its budget, revenue, rating, actors and actresses, etc. In this case study, we will use this dataset to determine whether any information about a movie can predict the total revenue of a movie. We will also attempt to predict whether a movie's revenue will exceed its budget.
Part 1: Data Preprocessing
We will inspect, clean, and transform the data.
Importing Libraries
First, we will import several libraries. scikit-learn (sklearn) contains helpful statistical models, and we'll use the matplotlib.pyplot and seaborn library for visualizations. Of course, we will use numpy and pandas for data manipulation throughout.
[ ] # Data Preprocessing
import pandas as pd
import numpy as np
# Data Visualisation
import matplotlib.pyplot as plt
import seaborn as sns
# Data Modeling

Limitation / Constraints of our System

Although our pipeline resulted in a pretty high score with Random Forest Ensemble model, we still have quite a few limitations that needs to be attended:

- The Dataset is too small in comparison, so, fitting a model with such limited data might fetch us a good result for now but it won't be able to for larger datasets.
- The method we used for treating missing values was by dropping those rows. Although this would be good for datasets with less noisy data but one with a high amount will provide inaccurate results.
- While optimizing our predictive models, we found out that none of the models are able to properly train for those movies with revenue generated as 0 while that is not unnatural in the real world.

Future Enhancements

In accordance to the limitations mentioned above, there can be various enhancements to be made in building our model. Some of them are:

- Using a very large Dataset supposedly provided by either Netflix or any other large database where all TV Series and other shows will also be considered since they are becoming quite popular nowadays.
- Explore and use various other feature engineering techniques to come up with a way for countering missing data. Also, since it's a large dataset, outliers should also be accounted for and taken care of.
- Using a two-layer neural network to segregate movie revenue into different categories, where, revenue is modelled as a discrete variable instead of as a continuous one that we used in our case.

CONCLUSION

There are various ways to implement a Predictive Model Pipeline for predicting Movie revenue and classifying Movie profitability but here we compared the most popular Ensemble model with the basic Statistical Models and drew an analysis of how we can improve the analysis and prediction score by using various feature engineering techniques. I also demonstrated that Logistic regression performs quite well even with faulty data and that Random Forest can produce very good results in spite of noises and outliers present in the data.

FEW SNAPSHOTS OF THE PROJECT

TMDb Movie Analysis and Comparative Prediction _ Sem 3 Project .ipynb

File Edit View Insert Runtime Tools Help Changes will not be saved

+ Code + Text Copy to Drive

- Most of the features have Pearson's Correlation Coefficient between 0.3 and 0.7 meaning that they have some relation between them.
- `return` feature is solely created for visualization and so is irrelevant to all the features.

Regression Target Analysis

```
# plotting the correlation graph between revenue and budget
sns.jointplot(x='budget', y='revenue', data=df2)
```

```
<seaborn.axisgrid.JointGrid at 0x7fed1d245160>
```

Observations:

- We can definitely see strong corelation between budget and revenue
- Quite a few number of outliers are present
- The relation is mostly linear

```
[ ] # plotting the correlation graph between revenue and popularity
sns.jointplot(x='popularity', y='revenue', data=df2)
```

```
<seaborn.axisgrid.JointGrid at 0x7fed1f830f60>
```

TMDb Movie Analysis and Comparative Prediction _ Sem 3 Project .ipynb

File Edit View Insert Runtime Tools Help Changes will not be saved

+ Code + Text Copy to Drive

- The movies that have good returns are necessary to have a good critical score.
- There are outliers present, although not too many.
- The relation is strongly linear.

This proves that Predicting `revenue` would require Regression model and thus we will be using - "Linear Regression" and "Random Forest Regression"

Movie return Trend analysis over the years

```
# Sorting out the title, vote_count and year to check the movies with highest vote_counts ever
df2[['title', 'vote_count', 'year']].sort_values('vote_count', ascending=False).head(10)
```

	title	vote_count	year
96	Inception	4.138397	2010
65	The Dark Knight	4.079290	2008
0	Avatar	4.071919	2009
16	The Avengers	4.071035	2012
788	Deadpool	4.041235	2016
95	Interstellar	4.036150	2014
287	Django Unchained	4.004321	2012
94	Guardians of the Galaxy	3.988693	2014
426	The Hunger Games	3.975707	2012
127	Mad Max: Fury Road	3.974420	2015

```
[ ] # Storing the same part of the Dataframe with revenue and return in a variable
most_popular_movies_and_return = df2[['title', 'vote_count', 'year','revenue','return']].sort_values('vote_count', ascending=Fals
```

```
[ ] np.log10(1000000000)
```

```
9.0
```

```
[ ] # Storing only the movies with a very high revenue, specifically revenue > 1,00,00,00,000 or 1 billion
most_popular_movies_and_return = most_popular_movies_and_return[most_popular_movies_and_return['revenue']>=9]
most_popular_movies_and_return.head(10)
```

	title	vote_count	year	revenue	return
65	The Dark Knight	4.079290	2008	9.001975	5.430046
0	Avatar	4.071919	2009	9.445287	11.763566
16	The Avengers	4.071035	2012	9.181717	6.907081

TMDb Movie Analysis and Comparative Prediction _ Sem 3 Project .ipynb

File Edit View Insert Runtime Tools Help Changes will not be saved

+ Code + Text Copy to Drive

```
[ ] # Determine the cross-validated correlation for linear and random forest models.
linear_regression_scores = cross_val_score(linear_regression, covariates, regression_outcome, cv=10, scoring=correlation)
forest_regression_scores = cross_val_score(forest_regression, covariates, regression_outcome, cv=10, scoring=correlation)

# Plot Results
plt.figure(figsize=(5,5), facecolor='w', dpi=120)
plt.axes().set_aspect('equal', 'box')
plt.scatter(linear_regression_scores, forest_regression_scores)
plt.plot((0, 1), (0, 1), 'k-')

plt.xlim(0, 1)
plt.ylim(0, 1)
plt.xlabel("Linear Regression Score")
plt.ylabel("Forest Regression Score")

# Show the plot.
plt.show()

# Print the importance of each covariate in the random forest regression.
forest_regression.fit(covariates, regression_outcome)
sorted(list(zip(all_covariates, forest_regression.feature_importances_)), key=lambda tup: tup[1])
```

```
[('TV Movie', 0.0),
 ('Foreign', 0.0001945188684806162 5),
```

References

1. TMDb 5000 Movie Dataset, Metadata on ~5,000 movies from TMDb, Version 2, 2017 <https://www.kaggle.com/tmdb/tmdb-movie-metadata>

2. Data Pre-processing, Wikipedia, October, 2020, < https://en.wikipedia.org/wiki/Data_pre-processing>

3. Exploratory data analysis, Wikipedia, October, 2020 <https://en.wikipedia.org/wiki/Exploratory_data_analysis>

4. Predictive modelling, Wikipedia, October, 2020 <https://en.wikipedia.org/wiki/Predictive_modelling>

THANK YOU