Finding the best prediction model for movies' revenue for TMDB

Omid Ghamiloo – 1859134

Master of Data Science

In the real world, what the factors are important on the revenue of a movie? How can we estimate the box office revenue?

This project's challenge is to estimate the revenue of movies by construction of a predictive model by using several data mining algorithms and regressions, like linear regression, random forest, lighting GBM and … on some data that we have all factors plus revenue about them. So, by making some relationship between factors, we can guess the revenue of the movies which do not have the revenue and then for evaluating the models, first choose 10 percent of our data that we have its revenue as test set to compare our models' prediction to the real values of revenue. For being sure, I'm not satisfied to choose 10 percent one time as test set, but I choose 10 time as random 10 percent of the data as test set by cross validation.

Dataset

the dataset which is used in this project, is contained 3000 movies in train and 5000 movies on test and variety of metadata obtained from The Movie Database (TMDB) which are downloaded from Kaggle website. Movies are labeled with some variables that you can see their description below.

Id: The Id number of each movie that is unique and its type is integer.

Cast: The name of the casts and their genders are the data that are included in this variable in a dictionary.

Crew: this variable type is as same as the cast but is contain the name of the crew in all sections like music, directing, production, visual effects, dressing, makeup and … with their genders and the specific jobs.

keywords of movies that are related words to the movies as string.

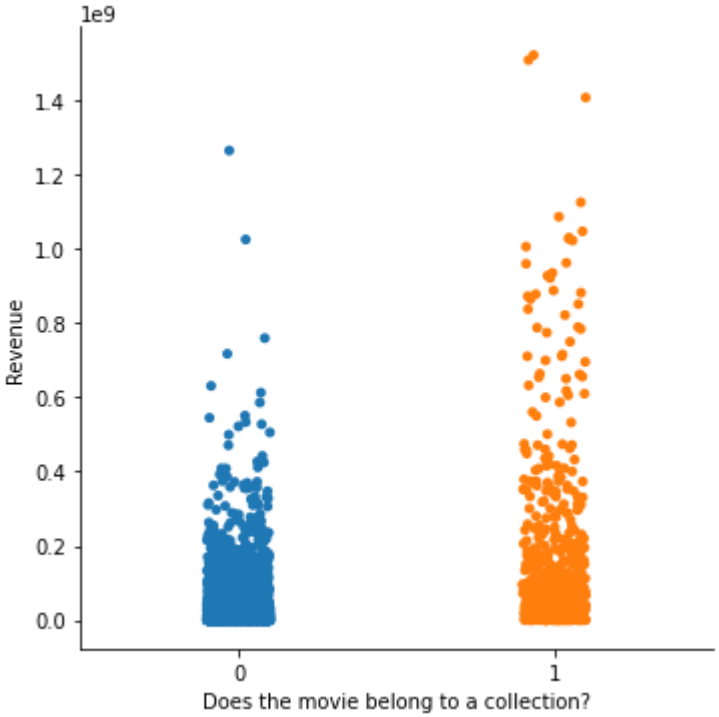
Budget: budget is total money that are spent for production of the movies. Its type is float.

Posters: poster variable is just a web link of the movie's posters.

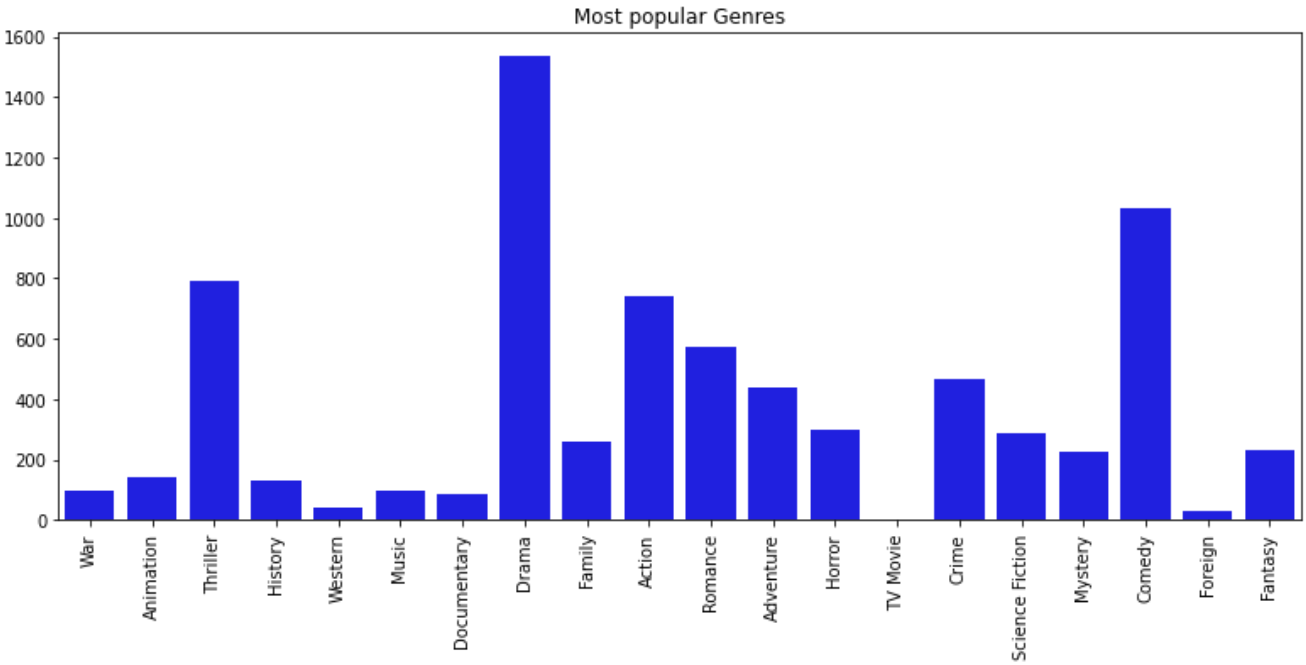
release dates: that is the exact date (day/month/year) of the publishing time.

original languages and spoken language: these are about the languages which are used in the movies. Some movies used more than one language. So, all the languages for each movie are gathered in a dictionary.

belong to collection: this is the other variable that is about the movie is in a series or not. This variable contains the collection name and its Id.



Genres: the movie genres that is categorical variable with values like Family, Horror, Sci fi, Romance and …. Like language variable, some movies might have more than one genre and they gathered in a dictionary. You can see the number of each genres in this dataset in the below bar plot.



production companies that are the companies which produce the specific movie.

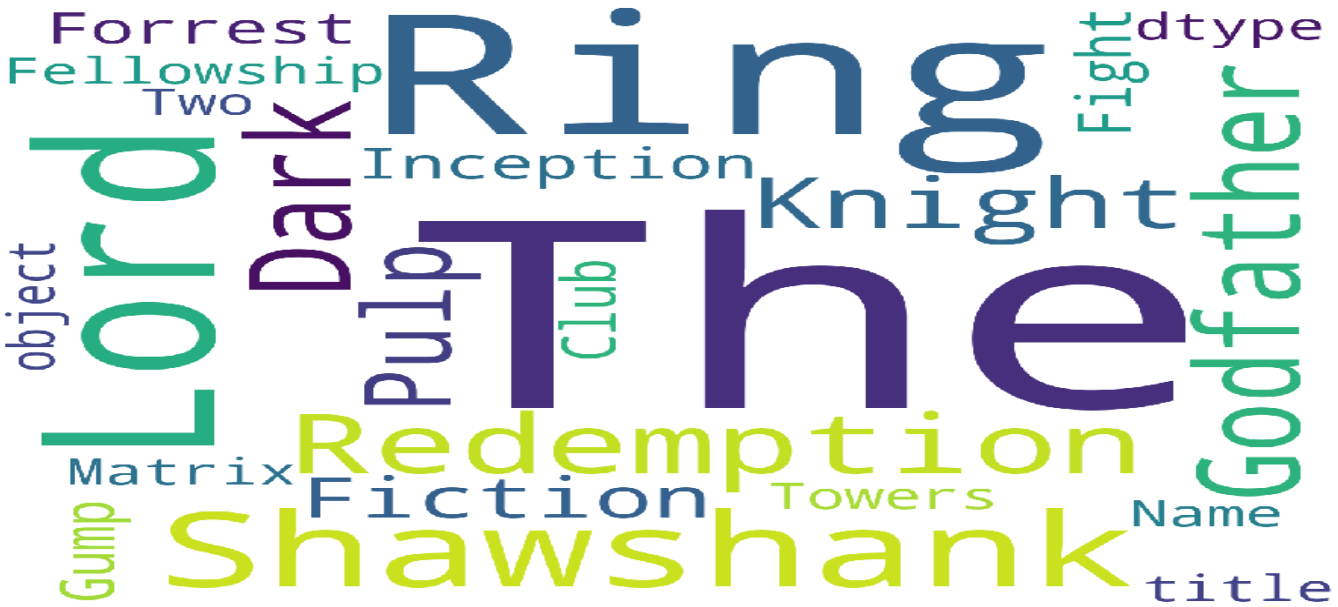
IMDB movie id that are used for data gathering from IMDB website.

runtime that is the duration of movies. Its type is integer.

Status is categorical variable and has two factors, published or not.

Overview: A brief summary of a completed screenplay's core concept, major plot points, and main character acts. Its type is string.

original title: the original name of the movie. You can see the name of the top 10 movies in the below word cloud.



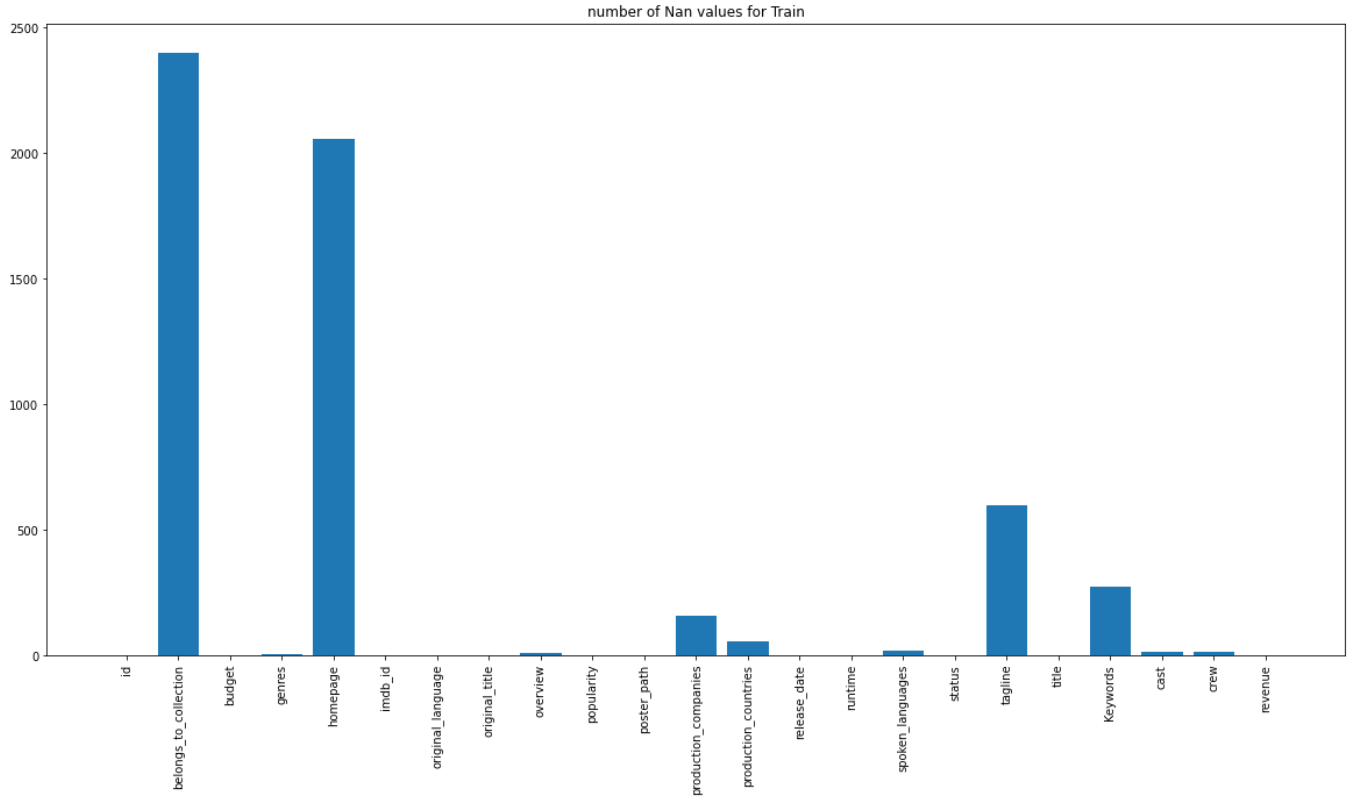
Revenue: the dependence variable that is float and its unit is USD.

Popularity: That is a numerical interval between 10 to 100.

Homepage: the link of the movie homepage.

countries: The countries that the movie are made in.

Now it’s the time for checking the missing values. You can see the plot of Nan values in the below bar chart.



I will talk about fixing missing values in the rest of the report.

Two other factors (number of votes and rate) are excist that I will talk about them in extra variables section.

Libraries

Some libraries are important for data analysis like Numpy, Pandas and matplotlib which are imported first. The others are imported when I needed them like scikit-learn for modeling and beautiful soup for web scraping.

Extra variables

The IMDB rate and number of votes could be useful for prediction, so added them to the trainset from IMDB website by the movies' IMDB id by web scraping. First find the page of each movies on IMDB, then find the content of the movies IMDB page and then find the class that is related to the rate and number of votes. Then put them into the dataset.

Nan values

The nan values of the dataset are shown below. Some variables (Genre, Overview, Keywords, Production companies) which are possible, are replaced by the correct values from IMDB website by web scraping. poster path and homepage are not important so we can remove them. 2996 numbers of the movies are released and just 4 of them are rumored on train and on test dataset there are 2 Nan values that are replaced by the released which are the most value.
Genres, overviews, keywords and production companies' Nan values are replaced by web scraping like number of votes and rates collection from IMDB website.

About tagline variable, its nan values are replaced by nothing as string type and spoken language variable's nan values are filled by original language variable, because the spoken language is as same as original language in many rows.

production countries' nan values are replaced by the maximum value of it, because it's categorical variable and we can fix the nan values by this way.

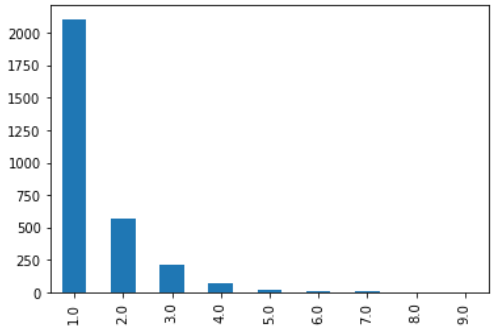
Runtime has zero in its values that are converted to nan values and then mean of runtime is the value which is replaced instead of nan values.

Cast and crew are nested dictionary as string that have some valuable variable which are extracted and fixed by some written function, for example a function that convert the string nested dictionary to dictionary or a function that convert list to string or vice versa.

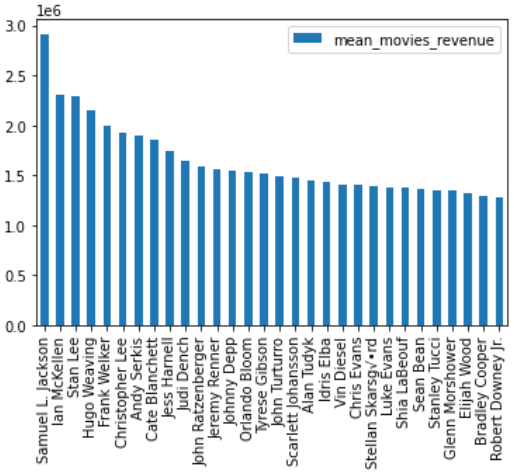
Budget has zero in some rows that are fixed by lighting GBM model that predict the zero values.

Data extraction

Some movies have more than one genre. Maybe the number of genres could be useful for the prediction of revenue. The same thing has been done on companies, countries and spoken language for revenue prediction. The bar plot is about number of spoken languages.



About the cast, the total number of total casts, number of female casts, number of male casts and number of casts who does not have gender and the name of cast are extracted from cast variable.
Another variable about the casts is created that is about the top casts by the mean revenue of them. The information of this variable is about the number of top casts who are in the movie.



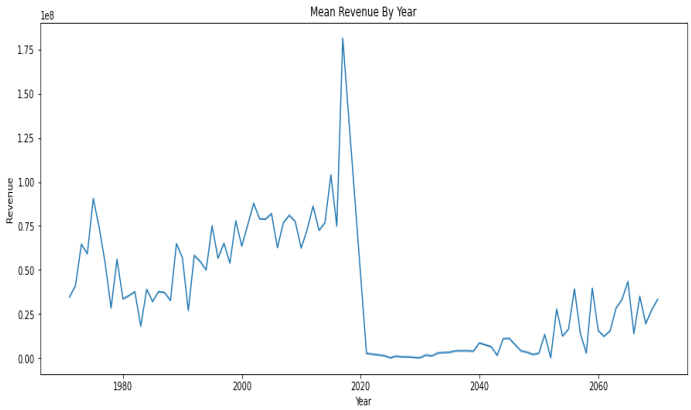
The crew variable has some information too. The director, writer, producer and music composer names, total number of them and the number of each gender are extracted and their nan values are filled by their mean value. Some new variables about top crews are created that are as same as top cast.

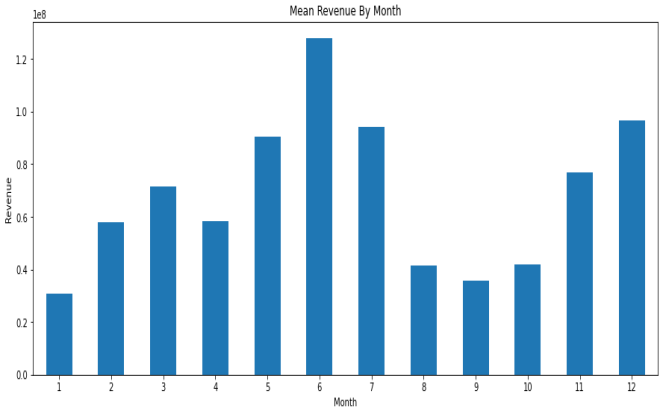
Has-collection variable is created by belong-to-collection variable and its values are 1 if the movie belongs to a collection or 0 if the movie does not belong to a collection.

About string variables like title and overview, the length of string of them maybe could be useful.

Many of movies are in English, so a variable is created about the movies are English or not.

From the release date, year, month, day, day of week, week of year and season are extracted. These are the plots about the mean revenue by year and month.





Top keywords, top genres and top companies are created as same as top cast too.

Dummies technic could be used for genres variable, because of categorical variable which is used on this dataset.

Feature engineering

weighted rate (The Bayesian Average): Bayesian Average computes the mean of a population by not only using the data residing in the population but also considering some outside information.

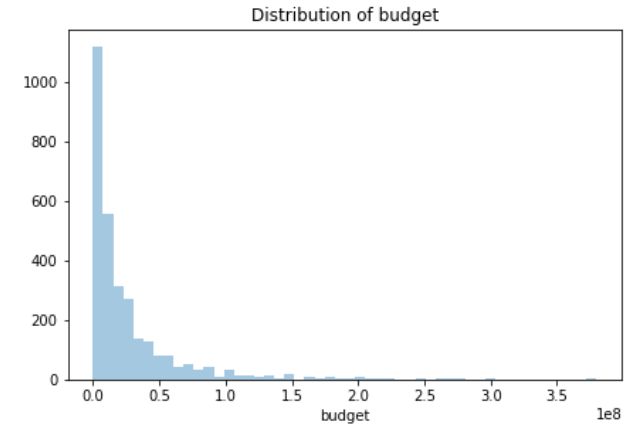
rating = ((v / (v + m)) * r) + ((m / (v + m)) * c)

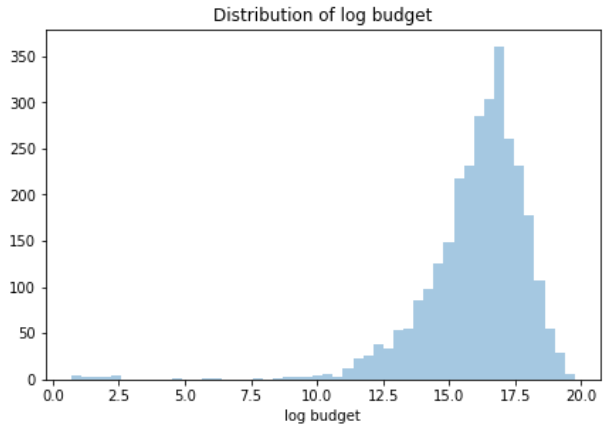
r is the average of rates by its number of votes, v is the number of votes, c is average if all rates and m is the minimum vote across the whole report (currently 25000).

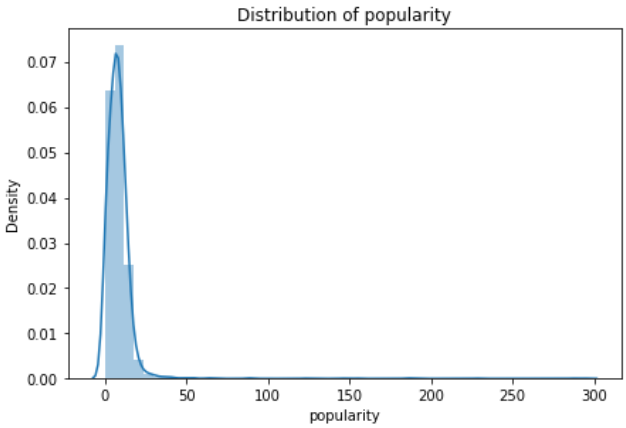
Some additional features are calculated like ratio of budget on year or budget on popularity and so on.

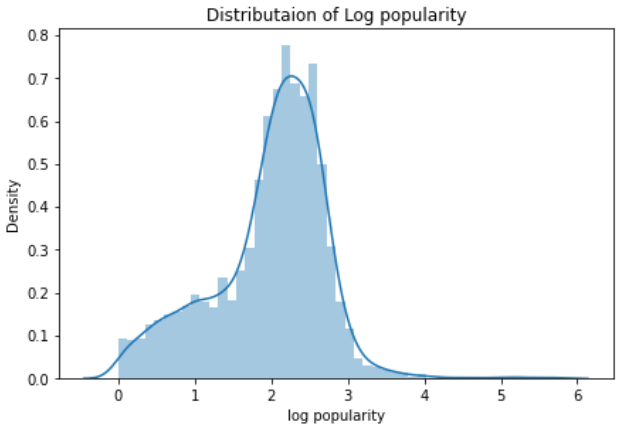
Normalization

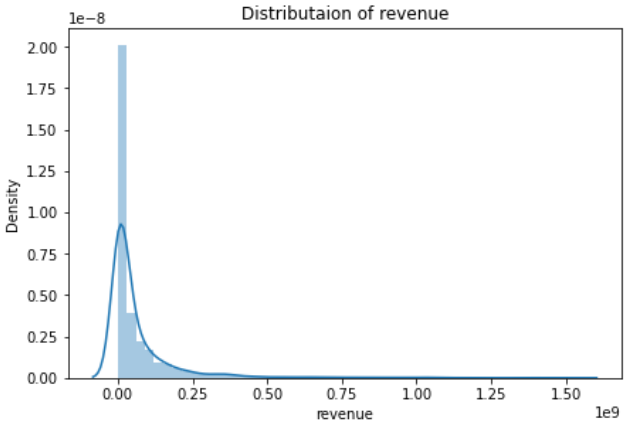
After checking the budget and popularity distribution and their logarithm distribution, I decided to use their logarithm in the models. The distribution plots are shown below.

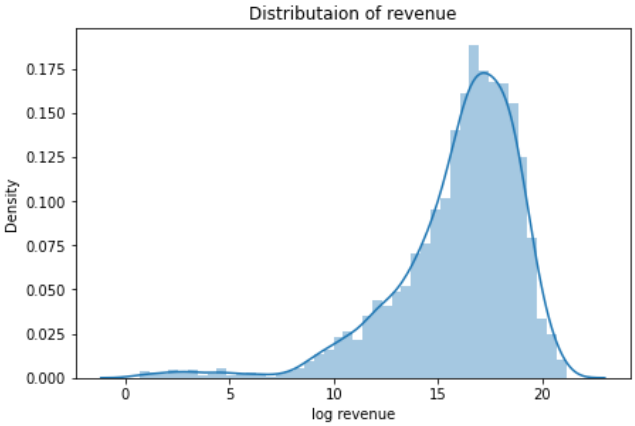




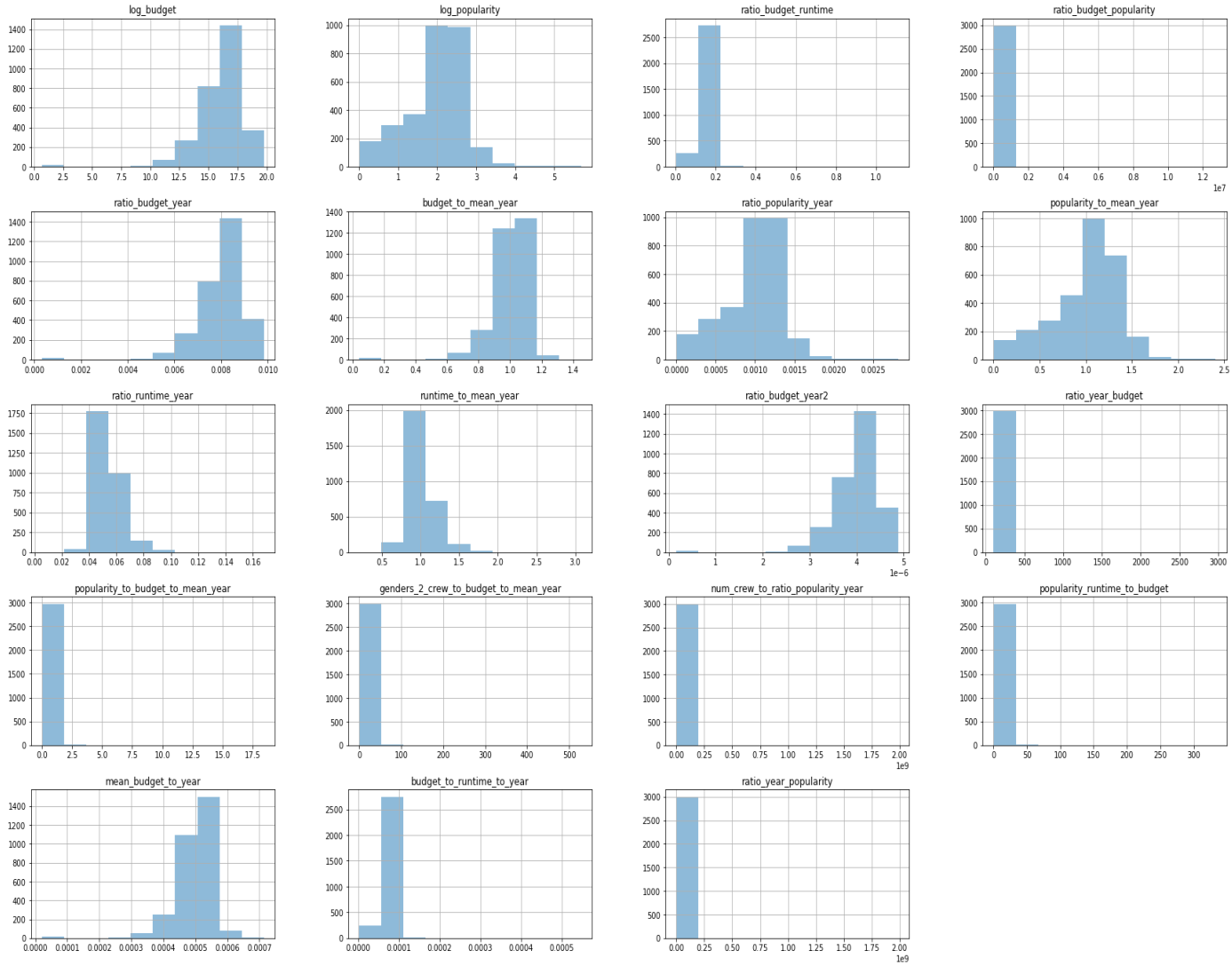








Then the new variables which are made in feature engineering section are checked for knowing that they should be scaled or not.



Then the variables which are needed, are scaled by scikit learn min max scaler library like runtime, length of string variables and number of votes.

Train and test split

The logarithm of revenue is selected as dependence variable and the other numeric ones as independence variable. For creating the models, 0.1 of our train set are chosen as test and 0.9 of it as train.

Models

Seven models are trained on this project. Linear regression, Lasso, Ridge, Elastic Net, Random forest, Lighting GBM and Cat boost.

Each of them is tuned by some of its parameters. By using randomized search cross validation and repeated K folds for finding the best values for each parameter.

Lasso: alpha is the parameter for the Lasso algorithm. I search the best parameter for alpha between 0 and 1 by the randomized search cross validation and repeated k fold. After that the alpha is chosen base on the minimum MAE. {'alpha': 0.0704}.and the model accuracy increased by one percent.

Ridge: alpha is the parameter for Ridge as same as Lasso. And after randomized search and repeated k fold and after all {'alpha': 19.998} is chosen. The accuracy increased just 0.1 percent.

ElasticNet: alpha is one of the parameters that is the mixing parameter between ridge (α = 0) and lasso (α = 1). The other parameter is l1_ratio. After randomized search and repeated k fold {'l1_ratio': 0.1297, 'alpha': 0.233} are chosen. The accuracy increased one precent in comparing with the ElasticNet without using hyper parameters.

Random forest: n_estimators, max_depth, min_samples_split, min_samples_leaf are the parameters that are used in hyperparameters section in Random forest algorithms. The number of estimators is selected from 200 to 1000 by 50 steps. Max depth are chosen from 10 to 100. Min samples split is from 1 to 10 and the min sample leaf is from 1 to 5. after all {'n_estimators': 346, 'min_samples_split': 4, 'min_samples_leaf': 3, 'max_depth': 76} are selected and increase the accuracy just 0.2 percent.

LightGBM: is another model that is used. The parameters of this model are learning_rate, max_depth, lambda_l1, bagging_freq, num_leaves, max_bin, feature_fraction, bagging_fraction and subsample. After all proccesing on randomized search {'subsample': 27, 'num_leaves': 10, 'max_depth': 43, 'max_bin': 406, 'learning_rate': 0.051, 'lambda_l1': 6.51, 'feature_fraction': 0.633, 'bagging_freq': 12, 'bagging_fraction': 1, 'num_boost_round': 520} these parameters are selected. The accuracy increased just 0.2 percent.

Cat boost: The last model that is used is Cat boost. The parameters of this model are chosen {'learning_rate': 0.03, 'l2_leaf_reg': 9, 'depth': 6}. And the accuracy unfortunately decreased 0.1 percent.

Evaluation

The evaluation parameters on this project are root mean squared error, mean absolute error, accuracy and cross validation on mean absolute error. The accuracy is calculated by 100 multiples on the mean absolute percentage error minus 100. You can see the difference value between base models and the tuned one.

	RMSEbase	RMSE tuned	RMSE Diff	MAE base	MAE tuned	MAE Diff	Accurbase	Accur tuned	Accur Diff	CV base	CV tuned	CV Diff
Linear Reg	1.854	1.854	0.000	1.341	1.341	0.000	89.020	89.020	0.000	1.730	1.730	0.000
Lasso	1.948	1.868	0.080	1.440	1.343	0.0967	88.514	89.270	0.756	1.636	1.450	0.186
Ridge	1.848	1.847	0.001	1.345	1.351	-0.005	89.067	89.115	0.0485	1.3973	1.402	-0.005
Elastic	1.947	1.871	0.076	1.434	1.343	0.090	88.619	89.263	0.643	1.613	1.450	0.163
Rand Forest	1.682	1.685	-0.003	1.150	1.148	0.001	90.463	90.436	-0.027	1.2367	1.2363	0.0004
Cat boost	1.667	1.658	0.008	1.122	1.129	-0.006	90.522	90.414	-0.108	1.196	1.198	-0.001
LGBM	1.721	1.683	0.037	1.138	1.119	0.018	90.426	90.647	0.221	1.212	1.207	0.005

The best model Considering the RMSE is Cat Tuned one with 1.65881283422061 RMSE

The best model Considering the MAE is LGBM Tuned one with 1.1198599722006688 MAE

The best model Considering the CV is Cat Base one with 1.1965434932087693 CV

The best model Considering the Accuracy is LGBM Tuned one with 90.64770979755164 Accuracy

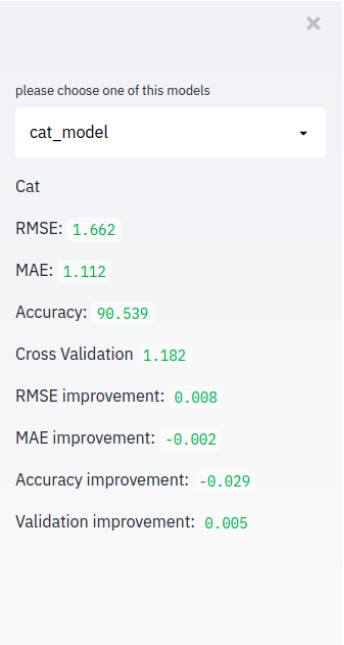
combination of the best of the models means Cat boost, Lighting GBM and Random forest with 10% of Cat boost, 60% of Lighting GBM and 30% of random forest, give the better model with accuracy 90.68%, RMSE 1.654 and MAE 1.110.

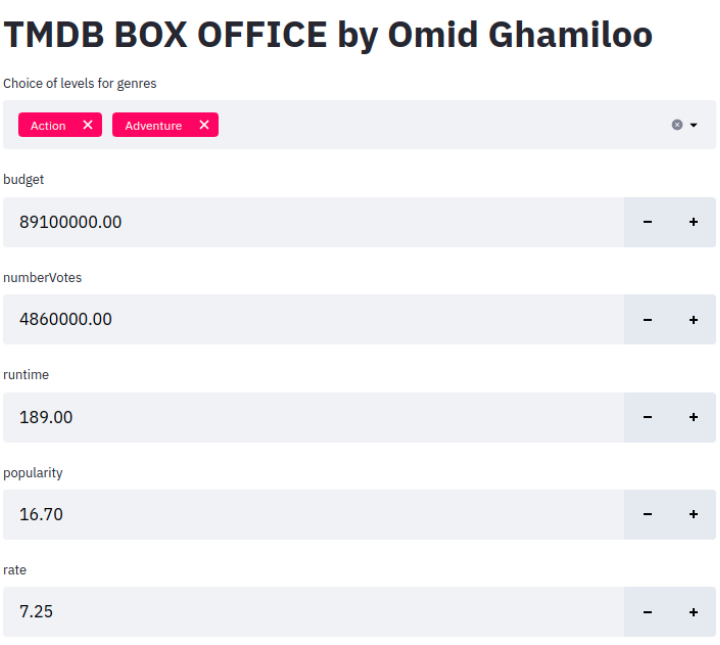
Dashboard Creating

For making the project more effective, I tried the Streamlit that you can find the details of codes in another file which is called Stream.

The dashboard contains two part. First part that is the main is about estimating the revenue of movies by your data. For example, you want to know the revenue of a movie which is made in April of 2019 in Action and Comedy genres with 2 million dollars as budget and so on, you can use this dashboard. In the main page you must select your values and after that click on Predict bottom and see the revenue which is estimated. For this prediction, you can choose your model too. The models are exported from the main python file of this project.

The second part of the dashboard is in sidebar. Some information about the evaluation of the model that you chose, will be shown in the sidebar.





please choose one of this models

cat_model

Cat

RMSE: 1.662

MAE: 1.112

Accuracy: 90.539

Cross Validation 1.182

RMSE improvement: 0.008

MAE improvement: -0.002

Accuracy improvement: -0.029

Validation improvement: 0.005

which one of these Directors, directed the movie?

Ron Howard

which one of these writers, wrote this movie?

Choose an option

which one of these Prodicers, produced this movie?

Tim Bevan

which one of these musicion, created the music of this movie?

Hans Zimmer

	has_collection
0	1

	is_eng
0	0

Predict

[1.9072821e+08]