

Kaggle Competition

Anna, Marcel, Matthew, Omar, Rishi





Initial Preprocessing

Removed features:

- imdb_id
- original_title
- poster_path
- status
- title

Transformed into boolean values:

- homepage
- belongs_to_collection
- original_language (english or not)

Other

- released_data --> transformed into release year, season, weekday
- runtime,budget --> missing values were added
- overview, tagline --> counting nbr of words
- genres --> multiclass hot encoding

Initial Modelling and Testing

Using Pycaret and Sklearn we ran several regression models to see which one was the most effective. After using the model with the highest R2 value we used that against the testing data. However this led to a result of 2.7!

	Model	MAE	MSE	RMSE	R2	RMSLE	MAPE
et	Extra Trees Regressor	40368419.5255	6348277073671206.0000	78846495.9100	0.8403	2.6808	33724.8964
rf	Random Forest Regressor	40464389.0058	6370684632735439.0000	78687119.7748	0.6386	2.6783	35384.5950
lightgbm	Light Gradient Boosting Machine	40573202.8891	6406044008808770.0000	78897989.0491	0.6369	2.6849	26582.5138
gbr	Gradient Boosting Regressor	40510681.9354	6581332782741144.0000	80066180.6416	0.6247	2.7267	26745.6225
lr	Linear Regression	47719073.3857	7007339523465278.0000	83154086.6867	0.5919	3.0742	89363.3710
lasso	Lasso Regression	47719070.0351	7007339410139310.0000	83154086.0422	0.5919	3.0742	89363.3156
ridge	Ridge Regression	47678028.5948	7005845584420441.0000	83145649.3636	0.5919	3.0747	88962.5832
llar	Lasso Least Angle Regression	47719070.0288	7007339423893557.0000	83154086.1248	0.5919	3.0742	89363.3266
en	Elastic Net	45065859.5544	7406747661059545.0000	85553517.1525	0.5667	2.8911	50749.3383
omp	Orthogonal Matching Pursuit	45468106.4857	7945082771729379.0000	88672186.1940	0.5329	2.8020	27866.8545
br	Bayesian Ridge	45414786.3783	8287460119541504.0000	90630269.5883	0.5116	2.8038	25019.5681
huber	Huber Regressor	42391838.2919	9045066974337708.0000	94253529.5678	0.4803	7.1596	4822.1448
knn	K Neighbors Regressor	50765711.0842	9946610663439126.0000	99264359.3609	0.4151	2.8797	34841.6702
par	Passive Aggressive Regressor	47201438.8758	1097281675919284.0000	103109847.9408	0.3545	2.6983	15900.3738
dt	Decision Tree Regressor	54217606.8776	12102026125779846.0000	109647824.6040	0.2481	3.0755	76291.6203
ada	AdaBoost Regressor	102294745.0899	13754038195547520.0000	116843462.6109	0.1839	4.0556	327196.2532
dummy	Dummy Regressor	76719259.2866	17704917074429304.0000	131560859.6130	-0.0043	3.7361	197425.2864
lar	Least Angle Regression	5774252033611.7627	406757362349181445269880832.0000	8462151059369.2686	-34209048778.4355	8.0010	6723702749.9296

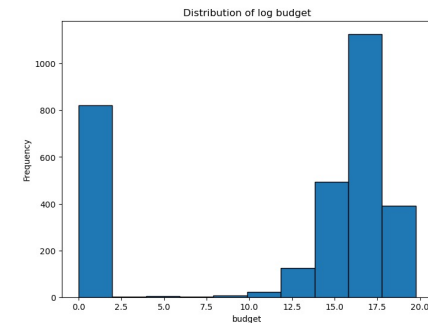
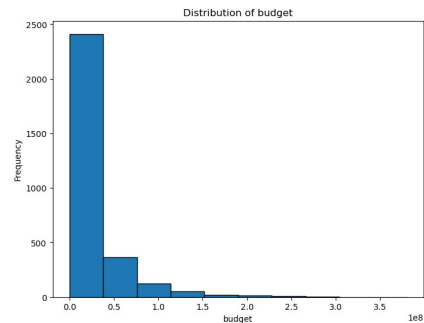
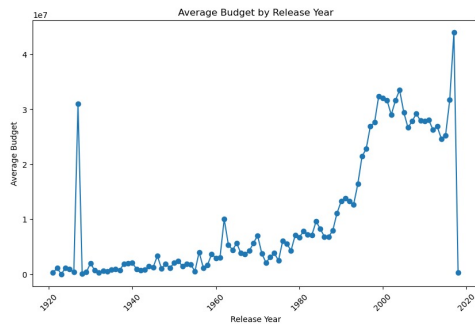
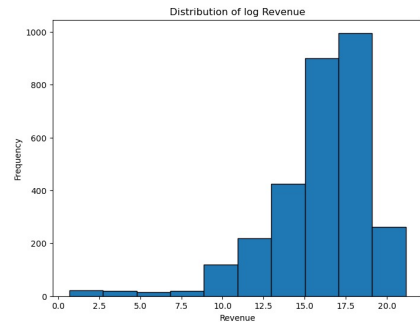
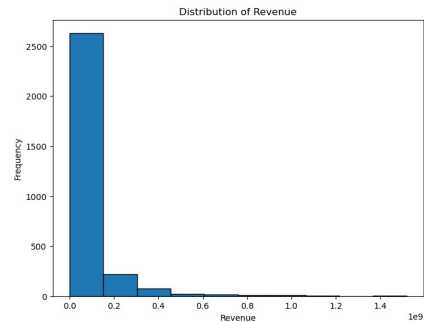


New features and model

- `budget_runtime_ratio` : `budget/popularity`
- `budget_year_ratio` - `budget/release_year`
- `production_companies_count` - count the nbr of production companies
- Added: `popularity_2`, `rating` and `totalVotes` from another database
- Filled budget for the missing values (some)
- Log transformation for budget and revenue
- Filtered the most common `production_companies` and `production_countries`
- The crew was broken down into different categories and counted how many participated in the movie
- Updated model to use `xgboost`, `lightgbm` and `catboost`
- Catboost was best

New features and model

- Exponential distribution for revenue and budget
- Close to normal distribution after log transformation
- Budget year ratio can make the budget feature more stationary

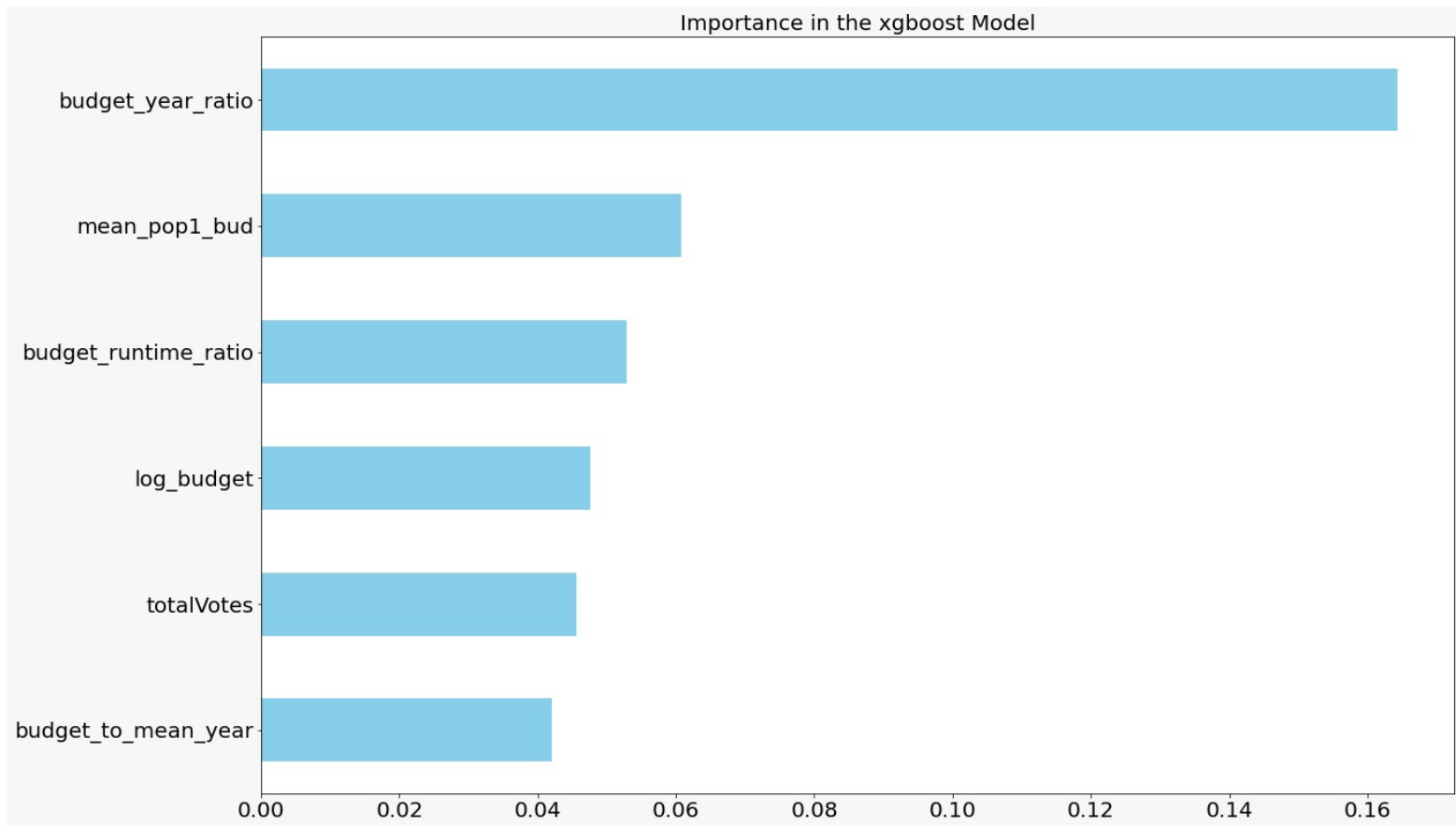


Final score = 1.97

(Root mean squared logarithmic error)



Most relevant features



**Thank you for
listening!**

Q&A