

Classification models for movies

Yutong He

1 Objective

The main objective of this project is to predict whether a movie would win any Oscar and Golden Globes award. This could provide predictive insights to the movie industry prior to the announcement of the annual award. Some anticipated sub-tasks are as follows:

1. Data cleaning and preliminary analysis. This entails dealing with missing entries or entries with invalid values, encoding non-numeric values to numeric ones, scaling the numeric values such that all features have the same input range for the model. To reach the objective, we will also need to identify the target column and the feature columns, and separate them.
2. The dataset should be checked for distribution skewness and, if extremely skewed, needs to be re-balanced for better model performance and more reliable classification results. Some techniques that could come handy when managing unbalanced datasets are class re-weighting, as well as over- and under-sampling to achieve parity between different classes.
3. We could visually inspect the correlations between different input features to gain a first-step intuitive understanding of the dataset.

The analysis is done by developing machine learning (ML) models trained on a labeled dataset containing movie features. The project makes use of the Kaggle dataset [Movies \(IMDb, Earnings and more\)](#), and the documented model development processes are available at this [Jupyter Notebook](#).

2 Data description and preparation

2.1 Data features and correlation

The dataset contains 3974 movies (rows) and 16 features (columns) initially. The features include a mix of object variables (movie names, director, actor(s), genre), and numeric variables (running time, budget, box office, IMDb score, etc.). The header and the first 5 rows are shown in figure 1. The basic information of the columns, such as their data types and the number of non-null content, is shown in figure 2. The descriptive statistical information about the numeric features, such as the mean, standard deviation, and quartiles, is shown in figure 3. We also visualize feature correlation in figure 4.

	Movie	Director	Running time	Actor 1	Actor 2	Actor 3	Genre	Budget	Box Office	Actors Box Office %	Director Box Office %	Earnings	Oscar and Golden Globes nominations	Oscar and Golden Globes awards	Release year	IMDb score
3471	The Broadway Melody	Harry Beaumont	100	Anita Page	Bessie Love	Charles King	Romance	379000	40000000	100.0	100.0	39621000	4	3.0	1929	6.3
2654	Hell's Angels	Howard Hughes	96	Jean Harlow	Marian Marsh	James Hall	Drama	3950000	8500000	100.0	100.0	4550000	1	0.0	1930	7.8
3272	She Done Him Wrong	Lowell Sherman	66	Mae West	Gilbert Roland	Louise Beavers	Comedy	200000	2000000	100.0	100.0	1800000	1	0.0	1933	6.5
2000	42nd Street	Lloyd Bacon	89	Ginger Rogers	Dick Powell	George Brent	Comedy	439000	2500000	100.0	100.0	2061000	2	2.0	1933	7.7
1843	It Happened One Night	Frank Capra	65	Claudette Colbert	Alan Hale	Walter Connolly	Comedy	325000	3500000	100.0	60.0	3175000	5	5.0	1934	8.2

Figure 1: The header and first 5 rows of the dataset.

```

<class 'pandas.core.frame.DataFrame'>
Int64Index: 3974 entries, 3471 to 0
Data columns (total 16 columns):
#   Column                                     Non-Null Count  Dtype
---  -
0   Movie                                     3974 non-null   object
1   Director                                3974 non-null   object
2   Running time                             3974 non-null   int64
3   Actor 1                                  3974 non-null   object
4   Actor 2                                  3974 non-null   object
5   Actor 3                                  3972 non-null   object
6   Genre                                    3974 non-null   object
7   Budget                                  3974 non-null   int64
8   Box Office                              3974 non-null   int64
9   Actors Box Office %                     3974 non-null   float64
10  Director Box Office %                   3974 non-null   float64
11  Earnings                                3974 non-null   int64
12  Oscar and Golden Globes nominations    3974 non-null   int64
13  Oscar and Golden Globes awards         3971 non-null   float64
14  Release year                           3974 non-null   int64
15  IMDb score                             3974 non-null   float64
dtypes: float64(4), int64(6), object(6)

```

Figure 2: Dataset information.

	Running time	Budget	Box Office	Actors Box Office %	Director Box Office %	Earnings	Oscar and Golden Globes nominations	Oscar and Golden Globes awards	Release year	IMDb score
count	3974.000000	3.974000e+03	3.974000e+03	3.974000e+03	3974.000000	3.974000e+03	3974.000000	3971.000000	3974.000000	3974.000000
mean	109.967036	3.690639e+07	1.086770e+08	2.984678e+09	52.780695	7.177060e+07	1.106694	0.321330	2002.043785	6.467866
std	22.507658	4.270431e+07	1.798354e+08	1.344684e+11	35.360199	1.526573e+08	2.724019	1.209324	12.127027	1.072514
min	20.000000	1.100000e+03	5.000000e+04	0.000000e+00	0.000000	-3.231000e+08	0.000000	0.000000	1929.000000	1.600000
25%	95.000000	9.000000e+06	1.200000e+07	3.333000e+01	25.000000	0.000000e+00	0.000000	0.000000	1998.000000	5.900000
50%	106.000000	2.200000e+07	4.300000e+07	5.714000e+01	50.000000	1.850000e+07	0.000000	0.000000	2005.000000	6.600000
75%	120.000000	5.000000e+07	1.250000e+08	8.333000e+01	80.000000	8.100000e+07	1.000000	0.000000	2010.000000	7.200000
max	330.000000	3.900000e+08	2.923000e+09	6.805556e+12	100.000000	2.686000e+09	22.000000	13.000000	2016.000000	9.300000

Figure 3: Dataset description.

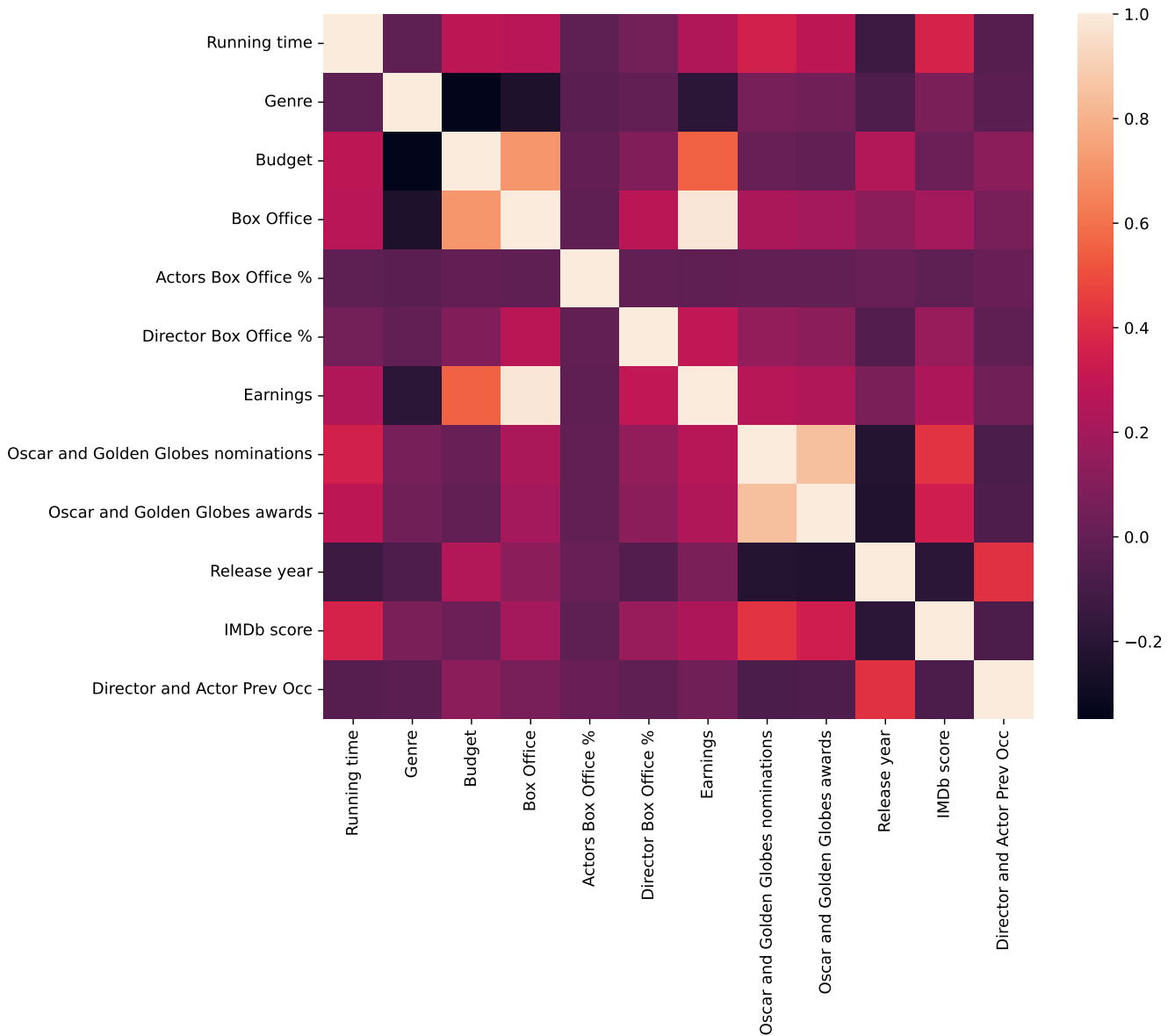


Figure 4: Heatmap showing the dataset features and their correlations, where the color gradient indicates correlation strength.

We also found out the distribution of the dataset was extremely skewed (see figure 5), as the vast majority of movies in the dataset won no award. Even combining all award winning movies into one category would not produce a significantly more balanced dataset (see figure 6). This suggests that care should be taken to balance the dataset before applying ML classifiers.

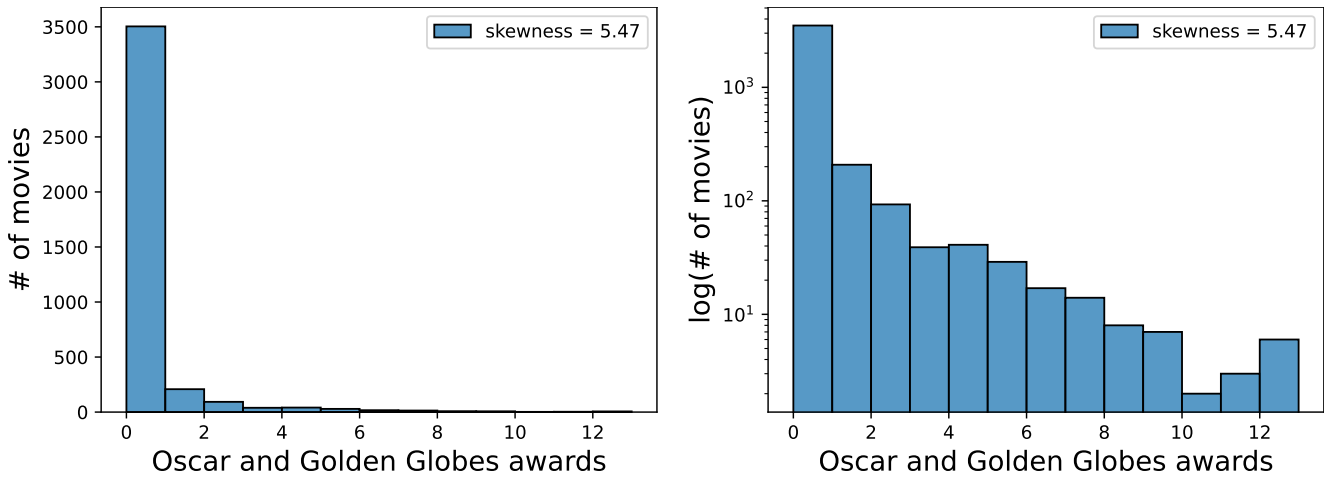


Figure 5: Skewed distribution of the original dataset in terms of the target variable on the horizontal axis. Linear (left) and logarithmic scales (right) are shown on the vertical axis.

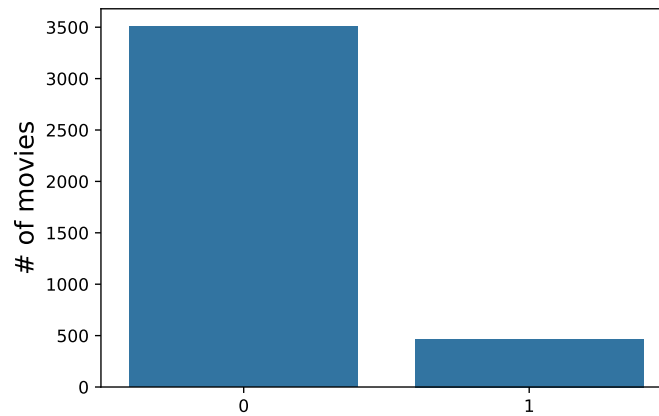


Figure 6: Clumping all award-winning movies in one category encoded with number 1, irrespective of the number of awards, would still not match category 0, i.e., movies that won no award.

2.2 Data cleaning

At the preprocessing stage, we performed the following tasks.

1. We dropped the column containing movie names. This assumed the names would not have effects on movies' award-winning chance as much as other factors in the dataset. A practical reason of dropping this column is if we were to keep it, then encoding the names uniquely to numerical values would create too many extra features to model.
2. We combined four columns in the dataset, ['Director', 'Actor 1', 'Actor 2', 'Actor 3'], into one, and encoded their impact in terms of the number of times any of the crew members won awards from previous years. Apart from the assumed correlation between the past success of a movie crew and the performance of a movie in the future from the same crew, a practical reason of doing so is also to avoid creating too many extra columns.
3. The movie ['Genre'] column was uniquely encoded to integers.
4. Rows with missing values were dropped.

2.3 Dataset re-balance

We apply three different methods to balance the originally skewed dataset and check their effects via a simple logistic regression model. The three methods are:

1. Class-reweighting. This introduces a weight on one class over another but the overall amount of samples in both classes remain the same. In this analysis, class 1 is assigned 0.8 and class 0 has 0.2.
2. Synthetic Minority Oversampling Technique (SMOTE). This method synthetically produces more samples in class 1 to match the samples in class 0 (see left panel of figure 7).
3. Random undersampling. This achieves the opposite of method 2 above and randomly choose initially more populated class 0 to match class 1 (see right panel of figure 7).

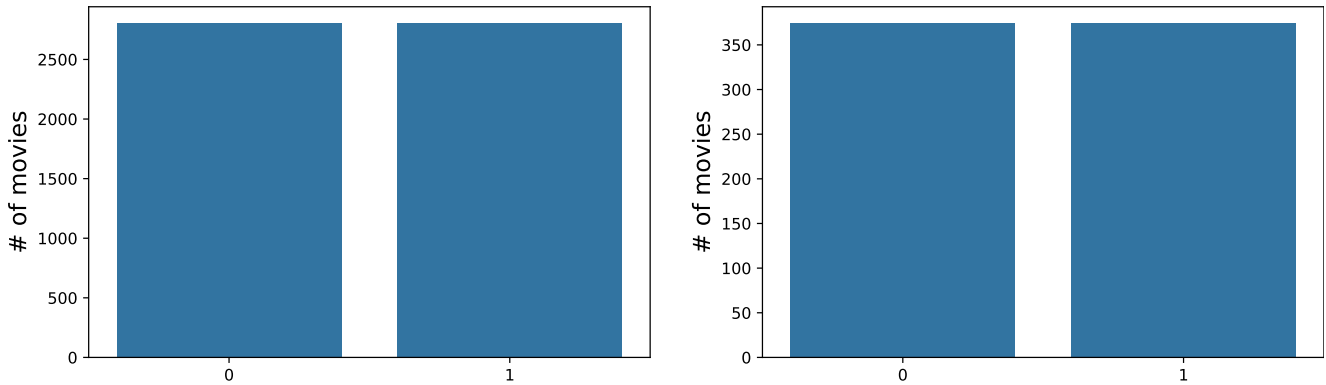


Figure 7: Using methods SMOTE (left) and undersampling (right), parity is achieved between two initially unbalanced classes.

Measuring the accuracy, precision, recall, f score, and auc from the three methods relative to those of the unbalanced dataset (see figure 8), we choose class-reweighting as the method to implement throughout the project. The confusion matrices corresponding to the three methods and for the original dataset are shown in figure 9 to illustrate the effect and importance of balancing the dataset.

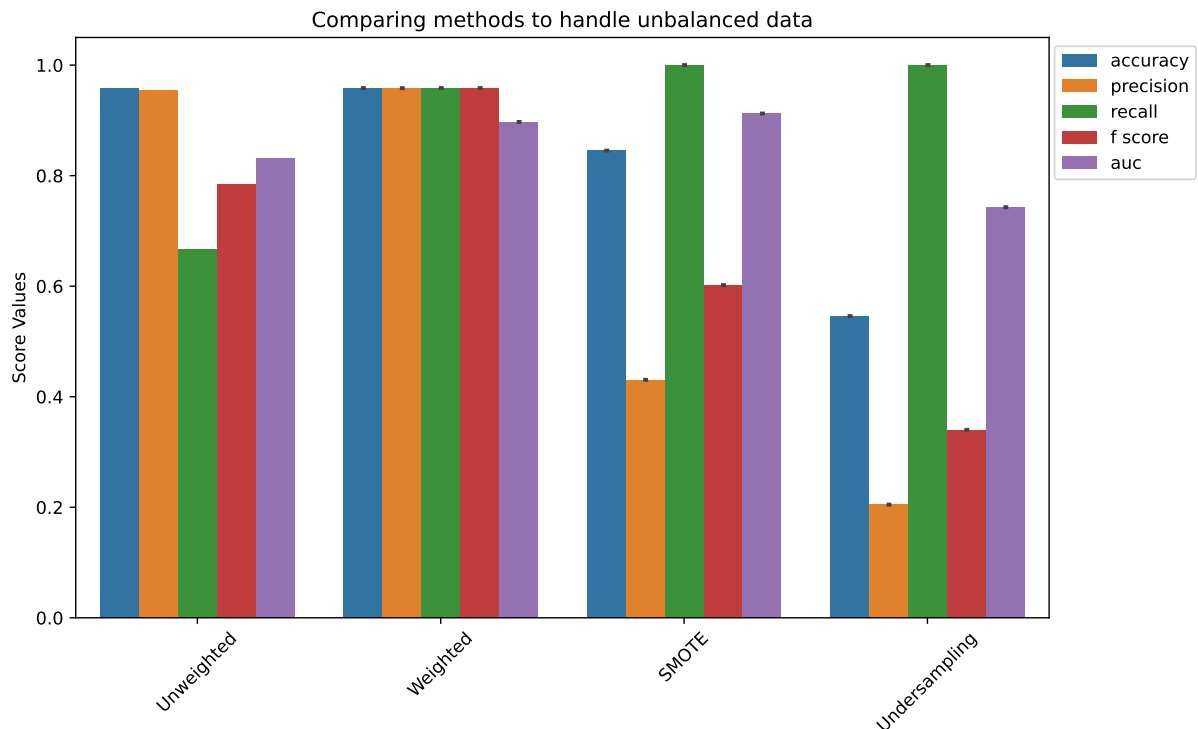


Figure 8: Comparing the effect of different methods to handle unbalanced dataset.

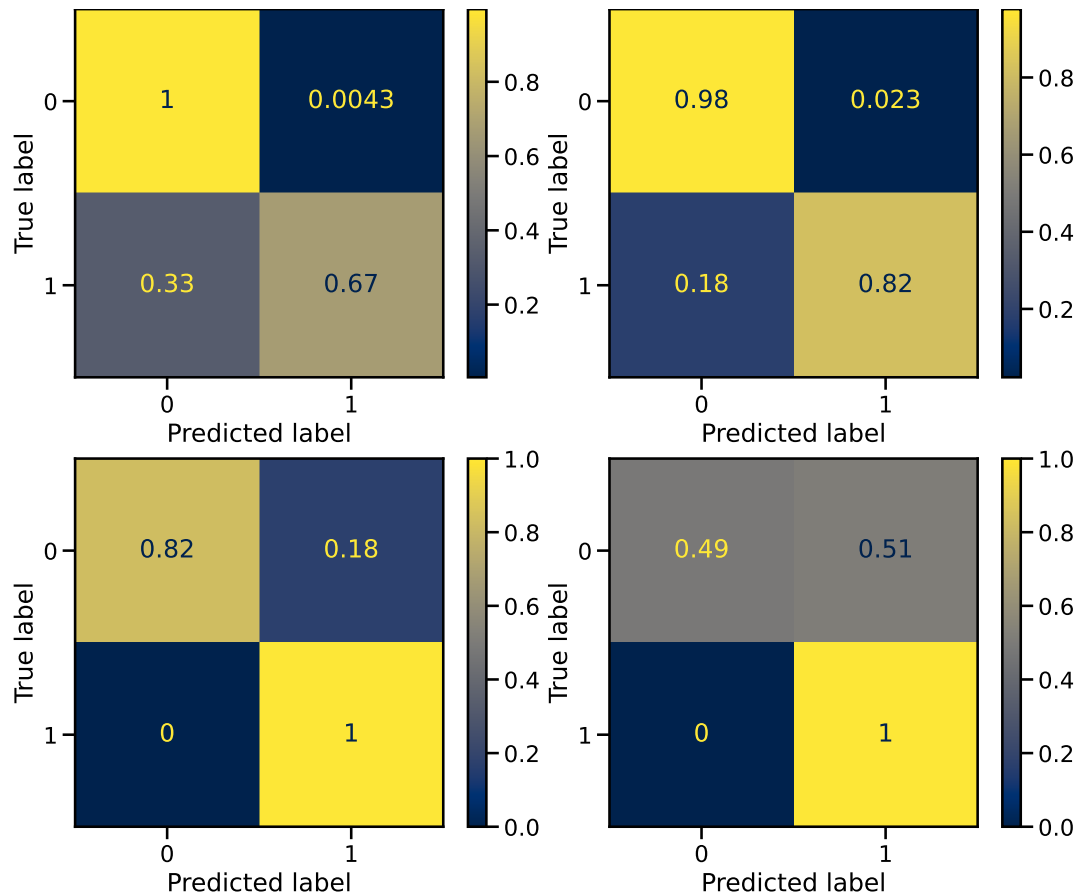


Figure 9: Confusion matrices of a logistic regression model applied to the original dataset without weight (upper left) and with class re-weighting (upper right), and to the SMOTE (lower left) and randomly undersampled dataset (lower right).

3 Classification models

Five classifiers were developed for this project: logistic regression with L1 and elastic-net penalties respectively, support vector machine (SVM), decision tree, and random forest. Different sets of model parameters corresponding to the relevant models were searched through for tuning purposes. The final results were obtained from each model using optimally tuned parameters. See the [Jupyter Notebook](#) for detailed model development and tuning process.

3.1 Classification model 1: logistic regression with L1 penalty

We used the input parameters shown in figure 10 for a logistic regression classifier and obtained the performance results indicated in table 1. Note that the choice of L1 penalty is a result of parameter optimization via a grid search between both L1 and L2 penalties.

```
LogisticRegression
LogisticRegression(class_weight={0: 0.2, 1: 0.7}, max_iter=500,
                    multi_class='auto', penalty='l1', random_state=42,
                    solver='saga')
```

Figure 10: Model parameters for the logistic regression classifier with L1 penalty.

Model	Accuracy	Precision	Recall	F score	AUC
Logistic regression (l1)	0.960	0.960	0.960	0.960	0.907

Table 1: Performance indicator of the logistic regression classifier with l1 penalty.

3.2 Classification model 2: logistic regression with elastic-net penalty

We used the input parameters shown in figure 11 for a logistic regression classifier and obtained the performance results indicated in table 2. Since l1 penalty proves to be more optimal than l2, it makes sense that the hybrid penalty leans toward l1, i.e., l1 ratio is tuned to 0.7 in the hybrid elastic-net penalty.

```
LogisticRegression
LogisticRegression(class_weight={0: 0.3, 1: 0.9}, l1_ratio=0.7, max_iter=500,
                    multi_class='auto', penalty='elasticnet', random_state=42,
                    solver='saga')
```

Figure 11: Model parameters for the logistic regression with elastic-net penalty.

Model	Accuracy	Precision	Recall	F score	AUC
Logistic regression (elastic-net)	0.962	0.962	0.962	0.962	0.899

Table 2: Performance indicator of the logistic regression classifier with elastic-net penalty.

Logistic regression provides an intuitive understanding of the factors contributing positively and negatively to the classification result, shown in figure 12. Some of the coefficients contribute negligibly to the outcome and are now shown in the figure. Of the five features that remain important in the trained logistic regressor, four correlate positively – Oscar and Golden Globes nominations, IMDb score, Budget, Director Box Office %, in descending order of importance. And one correlates negatively – Release year.

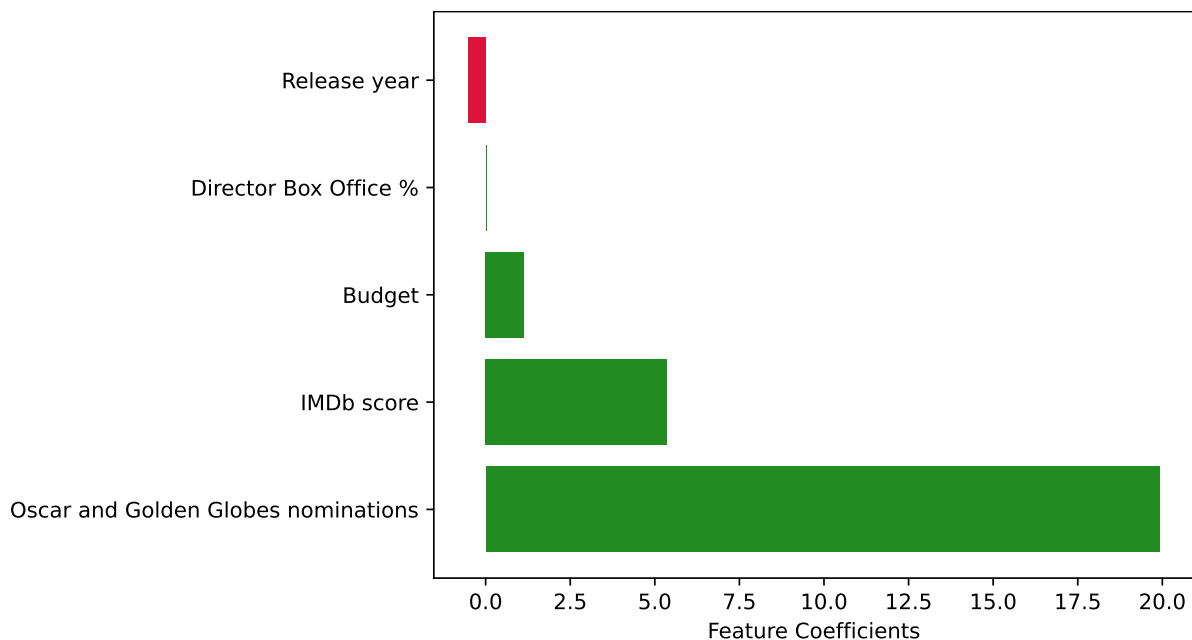
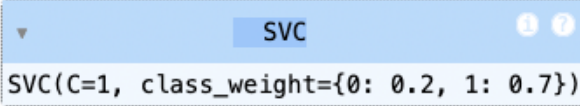


Figure 12: Coefficients of important features contributing positively (green) and negatively (red) to the classification output from the logistic regression model.

3.3 Classification model 3: support vector machine (SVM)

We used the input parameters shown in figure 13 for a SVM classifier and obtained the performance results indicated in table 3.



```
SVC(C=1, class_weight={0: 0.2, 1: 0.7})
```

Figure 13: Model parameters for the SVM classifier.

Model	Accuracy	Precision	Recall	F score	AUC
Support vector machine	0.965	0.965	0.965	0.965	0.915

Table 3: Performance indicator of the SVM classifier.

SVM operates by determining a hyperplane boundary separating the two classes that are otherwise rather difficult to disentangle. To illustrate the effect of SVM visually, we show in figure 14 the decision boundary between the two classes in a reduced case where only two features, Oscar and Golden Globes nominations and IMDb score, are selected. The solid blue curve indicates the boundary maximizing the distances away from support vectors (black circles) in each class 0 (red dots) and 1 (blue dots). As a result, most class 0 items (red dots) and class 1 items (blue dots) reside on either side of the decision boundary. This reduced SVM as an example also provides a rather straightforward visual illustration of the decision-making reasoning. But overall, SVM operating on the full dataset in higher dimensions does not output an easily visualizable boundary.

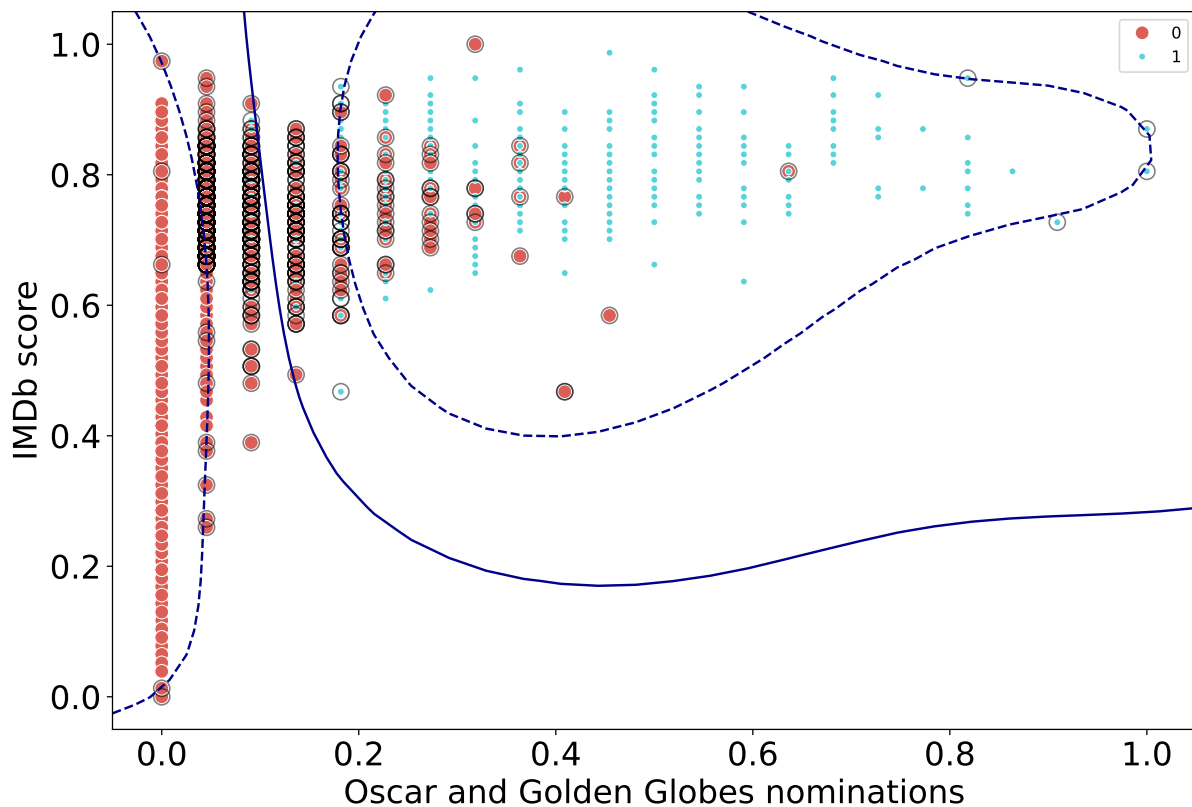
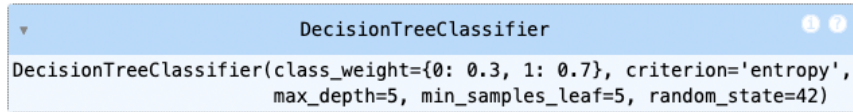


Figure 14: Decision boundary of a reduced SVM with only two features displayed on the horizontal and vertical axes.

3.4 Classification model 4: decision tree

We used the input parameters shown in figure 15 for a decision tree classifier and obtained the performance results indicated in table 4.



```
DecisionTreeClassifier(class_weight={0: 0.3, 1: 0.7}, criterion='entropy',
                        max_depth=5, min_samples_leaf=5, random_state=42)
```

Figure 15: Model parameters for the decision tree classifier.

Model	Accuracy	Precision	Recall	F score	AUC
Decision tree	0.937	0.947	0.937	0.937	0.908

Table 4: Performance indicator of the decision tree classifier.

Decision tree starts from the full dataset and successively splits into two or more branches until the final subsets contain objects within only one of the two classes. The tree is split such that the information purity is increased at every level further from the top. This process is rather logical and easy to follow in principle, although in practice the system becomes quickly complicated as the level goes deeper, such as figure 16 in the Appendix for our example on the movie dataset.

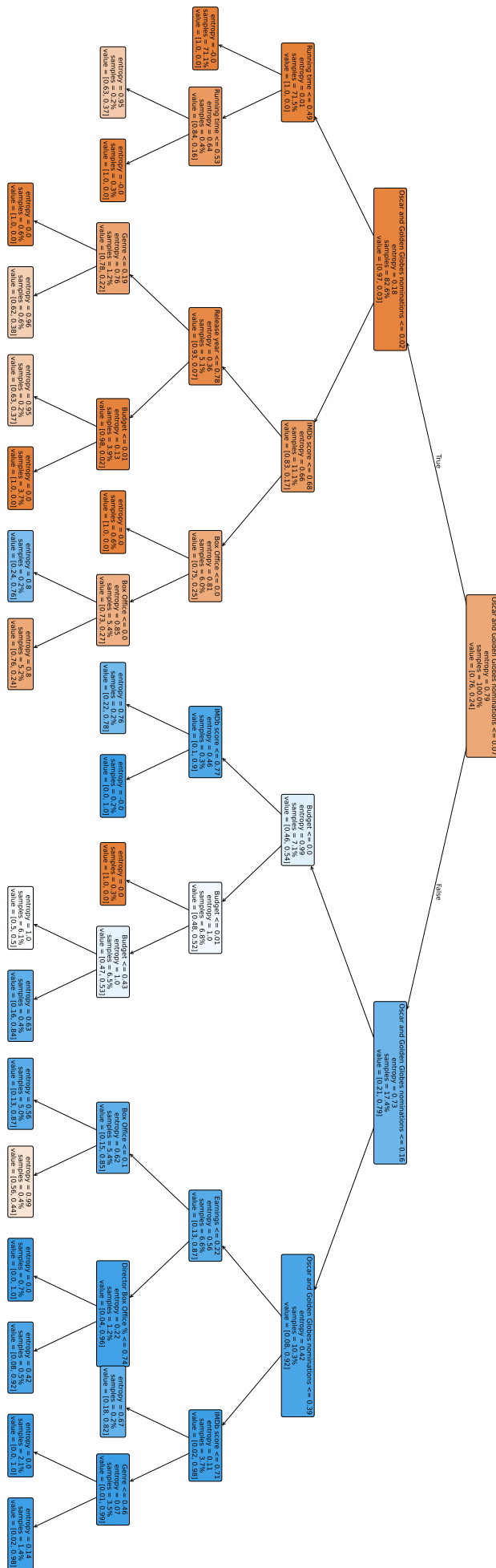


Figure 16: Decision tree visualization.

3.5 Classification model 5: random forest

We used the input parameters shown in figure 17 for a random forest classifier and obtained the performance results indicated in table 5.

```
RandomForestClassifier(
    class_weight={0: 0.3, 1: 0.7}, max_depth=13,
    max_features='log2', min_samples_split=5,
    n_estimators=16, random_state=42)
```

Figure 17: Model parameters for random forest classifier.

Model	Accuracy	Precision	Recall	F score	AUC
Random forest	0.962	0.961	0.962	0.962	0.890

Table 5: Performance indicator of random forest classifier.

Random forest is an ensemble model based on trees. But unlike the base decision trees, random forest assigns estimators that decorrelate features in the input dataset. The number of estimators is thus a tunable model parameter that does not have to be the same as the number of features. Figure 18 in the Appendix shows the correlation between the 16 random forest estimators selected for this project. This model is less intuitive than others since the estimators are assigned algorithmically and may not always correspond to variables that have real-world meaning. However, random forest can be a powerful classifier precisely because it could construct indicators that are less correlated among themselves.

	estimator 1	estimator 2	estimator 3	estimator 4	estimator 5	estimator 6	estimator 7	estimator 8	estimator 9	estimator 10	estimator 11	estimator 12	estimator 13	estimator 14	estimator 15	estimator 16
estimator 1	1.000000	0.724044	0.265788	0.662112	0.720188	0.404644	0.714982	0.640978	0.688985	0.698876	0.682436	0.630846	0.653398	0.737229	0.723085	0.659966
estimator 2	0.724044	1.000000	0.431390	0.800578	0.800630	0.493041	0.922539	0.723229	0.894243	0.867317	0.822295	0.698487	0.789582	0.927147	0.811437	0.600991
estimator 3	0.265788	0.431390	1.000000	0.378963	0.302593	0.385716	0.387956	0.379682	0.404437	0.379062	0.403318	0.372163	0.364212	0.390541	0.378963	0.485741
estimator 4	0.662112	0.800578	0.378963	1.000000	0.763828	0.507691	0.789720	0.752058	0.749608	0.843042	0.951291	0.805851	0.704257	0.804558	0.853874	0.580991
estimator 5	0.720188	0.800630	0.302593	0.763828	1.000000	0.459695	0.820692	0.693315	0.782557	0.852344	0.752578	0.722623	0.727798	0.814818	0.786327	0.544771
estimator 6	0.404644	0.493041	0.385716	0.507691	0.459695	1.000000	0.505126	0.467317	0.511582	0.531841	0.521243	0.532406	0.377188	0.471568	0.521243	0.469604
estimator 7	0.714982	0.922539	0.387956	0.789720	0.820692	0.505126	1.000000	0.723229	0.875006	0.867317	0.822295	0.710573	0.769368	0.907716	0.811437	0.609484
estimator 8	0.640978	0.723229	0.379682	0.752058	0.693315	0.467317	0.723229	1.000000	0.639014	0.708314	0.739645	0.660722	0.701154	0.715810	0.801708	0.561736
estimator 9	0.688985	0.894243	0.404437	0.749608	0.782557	0.511582	0.875006	0.639014	1.000000	0.878266	0.771182	0.715669	0.731375	0.869816	0.781968	0.568071
estimator 10	0.698876	0.867317	0.379062	0.843042	0.852344	0.531841	0.867317	0.708314	0.878266	1.000000	0.843042	0.765446	0.717866	0.871702	0.854089	0.629366
estimator 11	0.682436	0.822295	0.403318	0.951291	0.752578	0.521243	0.822295	0.739645	0.771182	0.843042	1.000000	0.819404	0.715592	0.837243	0.878228	0.590516
estimator 12	0.630846	0.698487	0.372163	0.805851	0.722623	0.532406	0.710573	0.660722	0.715669	0.765446	0.819404	1.000000	0.629492	0.726213	0.819404	0.512008
estimator 13	0.653398	0.789582	0.364212	0.704257	0.727798	0.377188	0.769368	0.701154	0.731375	0.717866	0.715592	0.629492	1.000000	0.824078	0.738262	0.490649
estimator 14	0.737229	0.927147	0.390541	0.804558	0.814818	0.471568	0.907716	0.715810	0.869816	0.871702	0.837243	0.726213	0.824078	1.000000	0.826348	0.579288
estimator 15	0.723085	0.811437	0.378963	0.853874	0.786327	0.521243	0.811437	0.801708	0.781968	0.854089	0.878228	0.819404	0.738262	0.826348	1.000000	0.600041
estimator 16	0.659966	0.600991	0.485741	0.580991	0.544771	0.469604	0.609484	0.561736	0.568071	0.629366	0.590516	0.512008	0.490649	0.579288	0.600041	1.000000

Figure 18: Random forest estimators and their correlations.

3.6 Best classifier for the objective: SVM

We compare the key findings from the 5 classifiers presented in sections 3.1–3.5 by looking at their accuracy, precision, recall, F score, and AUC summarized in table 6 and visualized in figure 19. Overall SVM has the highest scores across all metric. We also provide the confusion matrix of the SVM in figure 20. Given that SVM also has relatively easy interpretation conceptually, we choose it as the final classifier to predict the award-winning prospect of a movie.

Model	Accuracy	Precision	Recall	F score	AUC
Logistic regression (l1)	0.960	0.960	0.960	0.960	0.907
Logistic regression (elastic-net)	0.962	0.962	0.962	0.962	0.899
Support vector machine (SVM)	0.965	0.965	0.965	0.965	0.915
Decision tree	0.937	0.947	0.937	0.937	0.908
Random forest	0.962	0.961	0.962	0.962	0.890

Table 6: Performance indicator of classification models, where SVM ranks on top across all metric.

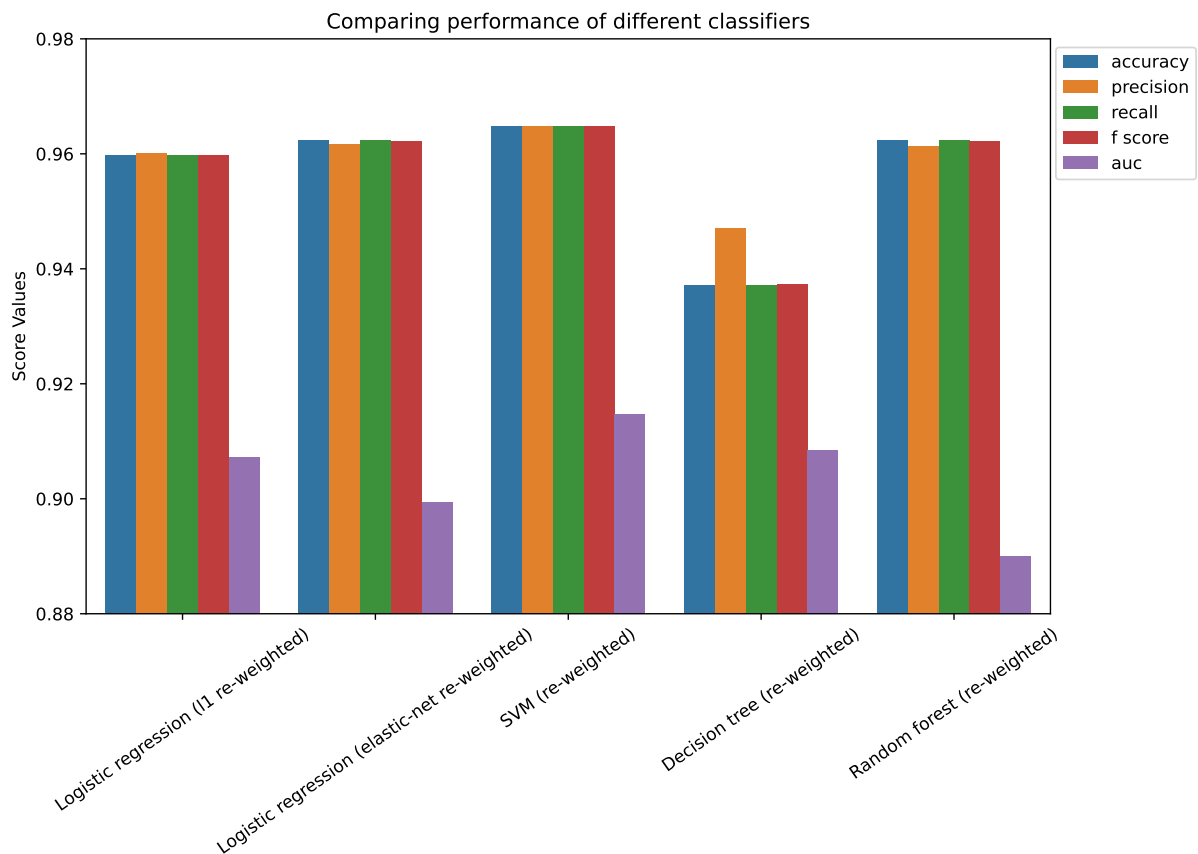


Figure 19: Performance comparison of the 5 classification models presented in sections 3.1–3.5.

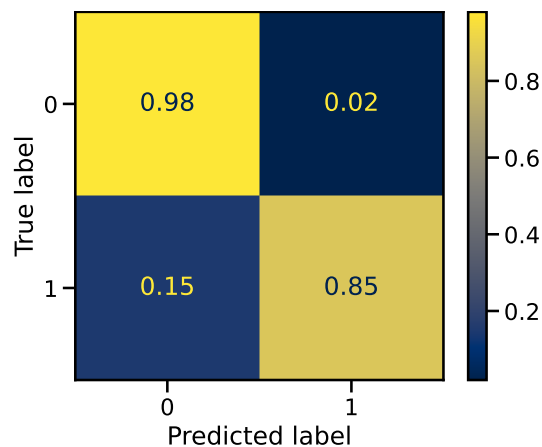


Figure 20: Confusion matrix of SVM, the best-performing classifier for the project objective.

4 Insights and key findings

Over the course of the project, we summarize the following key insights and findings:

1. Building ML models is made straightforward as many functionalities are readily available in python packages such as sklearn. But data preprocessing and exploratory analysis, especially balancing the dataset, presented in section 2 takes up the bulk of the project in terms of time spent.
2. ML models are project specific and need to be customized to address the individual questions and datasets. This is why we trained 5 different classifiers and determined SVM was the best for the particular task.
3. Different ML models have different degrees of interpretability. Logistic regression and decision tree classifiers made it the easiest to trace the step-by-step process from the input dataset to the final classification. However, they ultimately did not produce the best performing results compared to SVM. In this case, SVM provides some level of explanation, especially in the example of reduced input features, and is in any case a more explainable classifier than random forest.

5 Next steps

A number of aspects could be improved in the future, in term of both the technical model development and the data aspects.

1. In terms of the modeling, the next step would be to sweep through a larger parameter space to fine-tune the models for better performance.
2. A related aspect is to decide a different measure for optimal model parameters. In this project, the weighted f score is chosen as the metric for the grid search but perhaps a different measure would produce better model parameters.
3. A multiple class predictor would be an even better deliverable than a binary classifier but the input dataset would be extremely skewed and tricky to handle.
4. In terms of the dataset, a larger dataset containing awards other than Oscar and Golden Globes could perhaps compensate for the imbalance to some extent.