1 Do Wo(men) Talk Too Much In Films?
Project in Machine Learning

Abstract

This document contains a report of data analysis and classification models used to analyse a machine learning problem. The problem here is to classify the gender of the lead character in a movie based on the available data set of details of screenplays of around 1000 films.

2 Introduction

Hollywood or the movie industry in general is believed to be a male dominant industry since the inception of cinema. Though over the years, there have been changes in the relevance of female characters, it is still hard to say the change has made the industry gender-neutral. Here we are analysing screenplay data of around 1000 Hollywood movies, where we have records of the number of words spoken by male and female actors, the gender of lead actors and the movies' revenue from the box office. The objective of this report is to analyse the data and make predictions based on classification models whether the lead of the movie is a male or female character.

The data set we are using primarily consists of screenplay data in which we have fields such as "Lead" where it is assumed to be the person speaking most words and we have to consider co-lead as the opposite gender of Lead. And we have "Year" which is the year film was released and "Gross" which is the profit made by the movie. We also have "Number of female actors" and "Number of male actors". And in terms of words, we have fields such as "Total words", "Number of words male", "Number of words female", "Number of words lead" and "Difference in words lead and co-lead". Now in terms of age, we have fields such as "Lead Age" and "Co-lead Age" which is the age of lead and co-lead characters respectively. We also have the average age of males and females in the "Mean Age Male" and "Mean Age Female" fields.

3 Data Analysis

3.1 Do men or women dominate speaking roles in Hollywood movies?

Based on the data provided, we found that men have a dominant position on the screen when it comes to speaking roles in Hollywood movies.

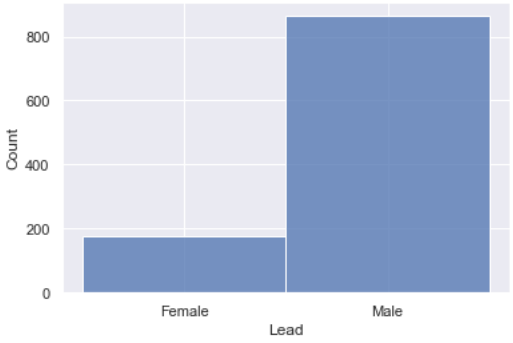

Figure 1: Male vs Female

As you can see from this chart, men dominate speaking roles over women. The number of movies in which male characters speak more than female characters exceeds 800. The number of movies with female characters speaking more words than male characters is just about 200.
In the given dataset the number of words spoken by male actors amounts to 71% of the total. According to these statistics, men dominate speaking roles by a wide margin.

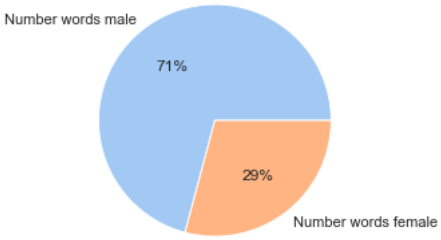

Figure 2: Male vs Female Words Comparison

3.2 Has gender balance in speaking roles changed over time (i.e. years)?

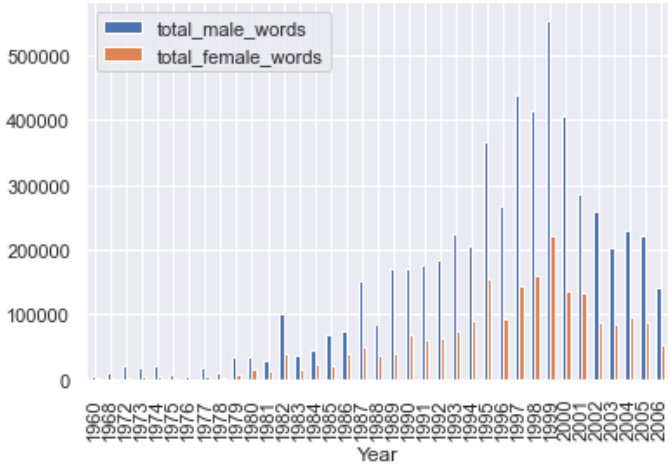

Figure 3: Gender Comparison

It was very rare for female characters in early 1960s movies to speak many words. A large majority of the dialogue was spoken by male characters. As far as speaking was concerned, female characters played a negligible role from 1960 to 1980. The early 1980s marked the beginning of a change in this trend. Female characters started taking more in Hollywood movies. Compared to the number of words spoken by male characters, their contribution was small.

However, as time went on, the number of words spoken by female characters has also been increasing, in spite of men speaking more words. This suggests that over the years there was progression to make gender roles more neutral, as seen in the figure below which plots the ratio of words spoken by male and female characters.

3.3 Do films in which men do more speaking make a lot more money than films in which women speak more?

Yes, The above graph clearly shows that the movies in which male characters speak more words than female characters collect more money than the movies in which female characters speak more.

4 Implementation of Classification Methods

In all the models, we have used a splitting ratio of 70:30 for training and testing, by using a function called train_test_split [15].

The Naive Classifier was used by assuming every prediction to be 'Male' as a benchmark to compare and analyse the results from different models.

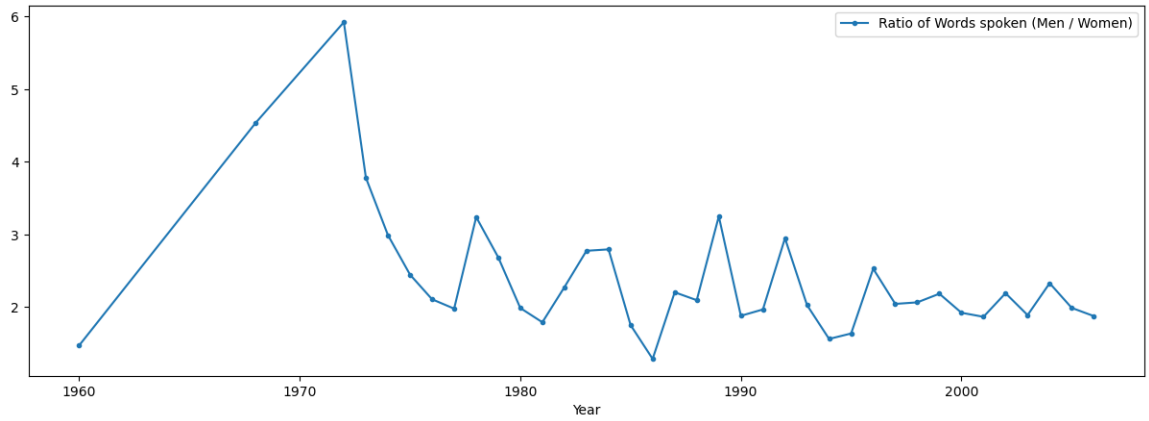

Figure 4: Ratio of words spoken (Male/Female)

Figure 5: Gross COmparison

4.1 Logistic Regression

As we know, Logistic regression is a classification algorithm which can be designed to predict categorical target labels, and in this case, we are trying to predict labels as whether the lead is male or female. Basically, Logistics regression comes from the need to transform a Linear regression model into a classification model based on the use of the logistic function which is also referred to as sigmoid function [16] which can be represented as h(z) = $\frac{1}{1+e^{-z}}$. Logistic regression can be implemented in python using sklearn.linear_model.LogisticRegression() [9] function. Here the regularized logistic regression is implemented using different solvers such 'liblinear', 'lbfgs', 'newton-cg', 'sag' and 'saga' solvers.

4.1.1 Feature Selection

Now coming to features used, we have used 'Number of words lead', 'Number of male actors', 'Number of female actors', 'Gross' and 'Year' from the data set. And additionally, we have added one field namely *"Number of words co-lead"* which is the difference between 'Number of words lead' and 'Difference in words lead and co-lead' which we feel can quantify the difference between lead and co-lead better than 'Difference in words lead and co-lead field specified in train data set. Over the years, the prominence of female characters in movies has been on the rise and we assume the gross revenue of the movies is affected in terms of the gender of the lead character. And the number of female and male actors could also provide a wide perspective for the analysis.

4.1.2 Modelling

For this classification method, as stated above we are splitting the data set into train and test data using sklearn.model_selection.train_test_split() [15] function in the ratio 70:30. Additionally we have used random_state as 50 to provide a random seed to the split generator. Also, we have used StandardScaler [14] method from the sklearn.preprocessing package to scale and transform the train data for producing better results.

In addition to this, for tuning the model, we have used GridSearchCV [2] function for various combinations of hyperparameters such as solvers, penalty and C value parameters. In the parameter grid, we have tried solvers such as 'lbfgs', 'liblinear', 'sag', 'saga' and 'newton-cg' with penalties as L1 and L2. Also, we have added logarithmic spaced C values and max_iter between the range 100 and 5000 to the parameter grid to find the optimum parameters. Based on the GridSearchCV function, we have found the best parameters to be solver as liblinear, penalty as L1, max_iter as 100 and C as 2.7825594022071245.

4.1.3 Evaluation

Now coming to the evaluation part, our initial model without hyperparameter tuning, we are getting an accuracy of 84% for the prediction model. And after hyperparameter tuning using the best parameters, the optimum accuracy was observed to be 87.5 %. And while applying the naive classifier for a model that always predicts males as lead actors, the accuracy is observed as 81.1%. The cross-validation score for this model is observed as 85% with a standard deviation of 3.7%.

4.2 Discriminant Analysis

Discriminant analysis is a multivariate technique used for separating two or more groups of observations based on variables noted on each experimental unit and computing the impact of each parameter in dividing the groups. [1]

The features were chosen by estimating the Chi-Square value between each feature and the target variable, using the function "SelectKBest" [13]. Chi-square is a statistical test used to calculate the independence of two events and when two features are unrelated to each other a small chi-square value is expected.

4.2.1 LDA Linear Discriminant Analysis

Linear discriminant analysis [8] is used as a tool for classification, dimension reduction, and data visualization.In this case, the distribution of X can be characterized by its mean (μ) and covariance (Σ), and explicit forms of the above allocation rules can be obtained. Following the Bayesian rule, we classify the data x to class j if it has the highest likelihood among all K classes for i = 1,. . . , K. LDA arises in the case where we assume equal covariance among K classes. That is, instead of one covariance matrix per class, all classes have the same covariance matrix. Then we can obtain the following discriminant function:

$$\delta_k(X) = X^T\Sigma^{-1}\mu_k - \frac{1}{2}\mu_k^T\Sigma^{-1}\mu_k + log\pi_k$$

The function used here was sklearn.linear_model.LinearDiscriminantAnalysis() [7].A classifier with a linear decision boundary, generated by fitting class conditional densities to the data and using Bayes' rule. In LDA, the features were selected using the function "SelectKBest" [13] as it selects the features based on their statistical significance. The number of features was selected by running a model in a loop, which resulted in k=13. By trying out a different number of features we came to the conclusion to use all the features as it gives better accuracy than the rest.

The features have been preprocessed with the help of "MinMaxscaler" [10]. From using the above parameters we got an accuracy of 84%. The cross_validation score for LDA is 84% and a naive classifier that always predicts male as the lead actor accuracy is 69%. Hyperparameter tuning was done using the function GridSearchCV [3] for the model parameter param_solver. In grid[solver] we chose singular value decomposition(SVD). After tuning the method with the k value we got an accuracy rate of 84%. The confusion matrix shows the accuracy of females and males in which the accuracy is 60% and 95% respectively where in males the misclassification is relatively low compared to the female counterpart.

4.2.2 QDA Quadratic Discriminant Analysis

Similar to LDA, Quadratic discriminant analysis is also a classification method based on Gaussian mixture models. But the difference is that since for each class, there is a covariance matrix calculated the decision boundary line ends up being quadratic because the logarithm of the Gaussian probability density function is quadratic. [17]

$$\hat{y}_* = arg\max_m \{ln\hat{\pi}_m + ln\mathcal{N}(x_*|\hat{\mu}_m, \hat{\Sigma}_m)\}$$

In this problem, after the initial Data Analysis, the QDA classifier was tested with different features in the dataset. New features were created as well in order to boost the performance by using the existing features such as "$Number\_words\_female$","$Number\_of\_words\_lead$","Age_Co_Lead", "$Number\_of\_male\_actors$","$Number\_of\_female\_actors$","$Number\_words\_male$", "Age_Lead".

The number of features selected was determined by running the model through a for loop with a varying number of input features and observing the highest accuracy, from our tests we found the ideal number to be 13. The main function used here was sklearn.discriminant_analysis.QuadraticDiscriminantAnalysis(). Hyperparameter tuning and k-fold cross-validation were done using the function GridSearchCV, which is a function that fetches the best parameter values to fit the model [11]. The model parameter *'reg_param'* [12], which is related to the per_class covariance and *'tol'* were the only parameters tuned. The final prediction accuracy score post-12-fold cross-validation was found to be 91.34% as compared to the naive classifier's accuracy of 69%.

4.3 K Nearest Neighbor

k-nearest neighbours algorithm (k-NN) is a non-parametric supervised learning method first developed by Evelyn Fix and Joseph Hodges in 1951 and later expanded by Thomas Cover [5]. K-NN is a distance-based machine learning method. When looking for the output label for a data point x, the K-NN algorithm will look for the nearest k data points and assign the label to which the neighbouring group belongs. Different distance metrics can be used to find the closest data points, such as Manhattan Distance, Euclidean Distance, and Minkowski Distance. K-NN can be used for classification and regression. In this case, we are using the KNN for a classification problem: predicting whether the lead is male or female based on the given parameters. Before beginning the modelling, data preprocessing step is performed to convert the data to a form that is ready for processing.

4.3.1 Feature Selection

Feature selection is an important step in modelling. Because it helps to select essential features from the data set which can give the best result for the modelling. Here we are using "SelectKBest" for feature selection [13]. By trying out the different numbers of features we came to the conclusion that choosing 5 gives the best accuracy. The important features that we are using here are: "Difference in words Lead and Co-Lead", "Number of male actors", "Number of female actors", "Age-Lead" and "Age co-lead".

4.3.2 Modelling

The data set is divided into testing and training data sets according to the train-test split described above. Since KNN is a distance-based algorithm, it is better to scale the features to the same range before modelling. It is done with the help of "MinMaxscaler" [10]. Now the data is ready for modelling. The model is constructed using KNeighborsClassifier available in sklearn [6]. For each value of k, the training score and testing score are calculated for each value of k in the range of 0 to 20. For tuning GridSearchCV method is used [2]. After tuning, the final result is that the model gives the best result when k is 11 and the weights are uniformly distributed. The test accuracy of this model is 81 per cent. As a final step, the performance of the model is validated using the K-fold cross-validation method to see how the model will perform when unseen data is given to the model [4]. The cross-validation score obtained is 79%.

4.4 Tree based methods

4.4.1 Classification trees

Classification trees are a type of Decision tree used for categorical value predictions, where the predictions are based on the density of data points from each categorical value in that area. The splitting criterion is usually misclassification rate (squared loss), entropy/deviance, gini index [17]. The features and the number of features were estimated and used similarly to that in QDA. Hyperparameter tuning was done using the function GridSearchCV [11] like in previous methods. The parameters which were tuned were $split\_criterion$ - criteria used for splitting at each node and in particular, max_depth - maximum depth of each tree, as this was the important parameter that determines the model which doesn't have high bias and variance. The k value for cross-validation was also selected based on fitting the model over a for loop and observing the k value for the best score. The final accuracy was noted to be 80.1% after an 8-fold cross-validation, compared to the naive classifier's accuracy of 69.2%.

4.4.2 Random Forest

Random forests are similar to the bagging methodology where the dataset is split into *'B'* components, trees are constructed for each dataset and are then aggregated in the end by averaging. But in random forests, only some random subset of the input is considered as splitting variables. This helps to maximize variance reduction as the *'B'* trees are being de-correlated in the process while selecting random subsets. Due to this factor, the performance of Random Forests is generally better compared to classification and regression trees (CART)

Similar to QDA, the features and number of features were estimated and used here as well and the hyperparameter tuning was done using the function GridSearchCV [11]. The parameters tuned were *'n_estimators'* - the number of trees, a higher number could lead to better performance,*'max_depth'* - maximum depth of each tree, *'min_samples_split'* - minimum number of samples required to be counted as a leaf node, *'min_samples_leaf'* - the size of the leaf, which is, the end node of the tree. After performing the iteration for the k value for cross-validation, the accuracy was noted to be 81.7% after a 10-fold cross-validation, compared to the naive classifier's accuracy of 69.2%.

4.5 Boosting

Gradient boosting is a machine learning technique used in regression and classification tasks, among others. Boosting is a technique that uses a random sample of data, fits a model, and then trains a sequence of models, each one attempting to compensate for the weakness of the previous one. Each iteration combines the weak rules from each classifier to create a strong prediction rule.

4.5.1 Feature Selection

In this case, we are using the GradientBoosting algorithm to predict whether the lead is "Male" or "Female" based on the given data. All of the features given in the data set are used for modelling in this case. During the modelling, the model decides which are the best parameters to use. As we tried out different parameters, we found that choosing the number of parameters as ten produces the best accuracy.

4.5.2 Modelling

For modelling the data set is first divided into a training data set and a testing data set using the $train\_test\_split$ method which is mentioned in the introduction section. The model is constructed using "GradientBoostingClassifier" available in scikit-learn library. For getting the best result the model is trained on different learning parameters. The learning parameters that we mainly used are 0.05,0.075,0.1,0.25,0.5,0.75 and 1. The model got the best result when the learning rate was 0.5. The accuracy that we got in this stage is 86.5%. Then the model is tuned using GridSearchCV [2] and the accuracy got improved to 89.2%. Finally, the performance of the model is evaluated using the K-fold cross-validation method.

Table 1: Classification model metrics

Classification	Accuracy	Precision	Sensitivity	Specificity
Logistic Regression	87.5	90.2	94.9	55.9
LDA	84.60	84.4	95.3	60.4
QDA	91.34	89.0	97.6	72.9
K Nearest Neighbor	81.1	83.3	93.3	47.6
Classification Trees	80.12	83.47	88.9	60.4
Random Forests	81.1	80.3	96.7	47.9
Boosting	89.2	85.9	93.0	57.3

5 Comparison of Classification Models

Based on the different classification models and their corresponding parameter sets we have tried to predict the lead character of the film. Table 1 represents the accuracy and other metrics of each model.

While comparing the different classification models which was obtained using various combinations of input features and parameters, it can be observed that the accuracy score of Quadratic Discriminant Analysis QDA is having the highest accuracy rate. It is also pretty clear from the table, that the discriminant-based methods performed better than the tree-based methods with the given input features. But based on one metric alone we are not coming to the conclusion that QDA is the best classification model to predict the lead of a movie.

From the table, it is clear that QDA is having the upper hand in Sensitivity and Specificity where as it is marginally behind Logistic Regression in Precision metrics. Although accuracy could be improved further with Random Forests using a better hyperparameter tuning with more values, because of its computational complexity that wasn't possible. The specificity also is observed to be below 0.80 for all the models, this could be due to the result of the dataset being biased towards Men. But when compared to other models QDA has higher specificity. Due to the reasons stated above, we are going ahead with implementing the QDA model for the test data set, with features and parameters specified during our classification of the training data set.

References

[1] _discriminant_analysis. https://www.statisticssolutions.com/discriminant-analysis/.

[2] Gridsearchcv. https://https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html.

[3] Gridsearchcv function. https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html.

[4] K-fold cross-validation. https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.KFold.html.

[5] K-nearest neighbors. https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm.

[6] Knearestneighbour. https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html.

[7] Linear discriminant analysis. https://scikit-learn.org/stable/modules/generated/sklearn.discriminant_analysis.LinearDiscriminantAnalysis.html#sklearn.discriminant_analysis.LinearDiscriminantAnalysis.

[8] Linear_discriminant_analysis. https://www.geeksforgeeks.org/ml-linear-discriminant-analysis/.

[9] Logistic regression. https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html/.

[10] Minmaxscalar. https://https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html.

[11] Quadratic discriminant analysis. https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html/.

[12] Quadratic discriminant analysis. https://scikit-learn.org/stable/modules/generated/sklearn.discriminant_analysis.QuadraticDiscriminantAnalysis.html/.

[13] Selectkbest. https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.SelectKBest.html.

[14] Standard scalar. https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html.

[15] train_test_split function. https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html/.

[16] Understanding logistic regression. https://https://www.geeksforgeeks.org/understanding-logistic-regression/.

[17] Andreas Lindholm, Niklas Wahlström, Fredrik Lindsten, and Thomas B. Schön. *Machine Learning - A First Course for Engineers and Scientists*. Cambridge University Press, 2022.