PROJECT 2

Vidhi Kokel (20241891)

Electrical & Computer Engineering
STAT 457/857 Statistical Learning

1. Introduction

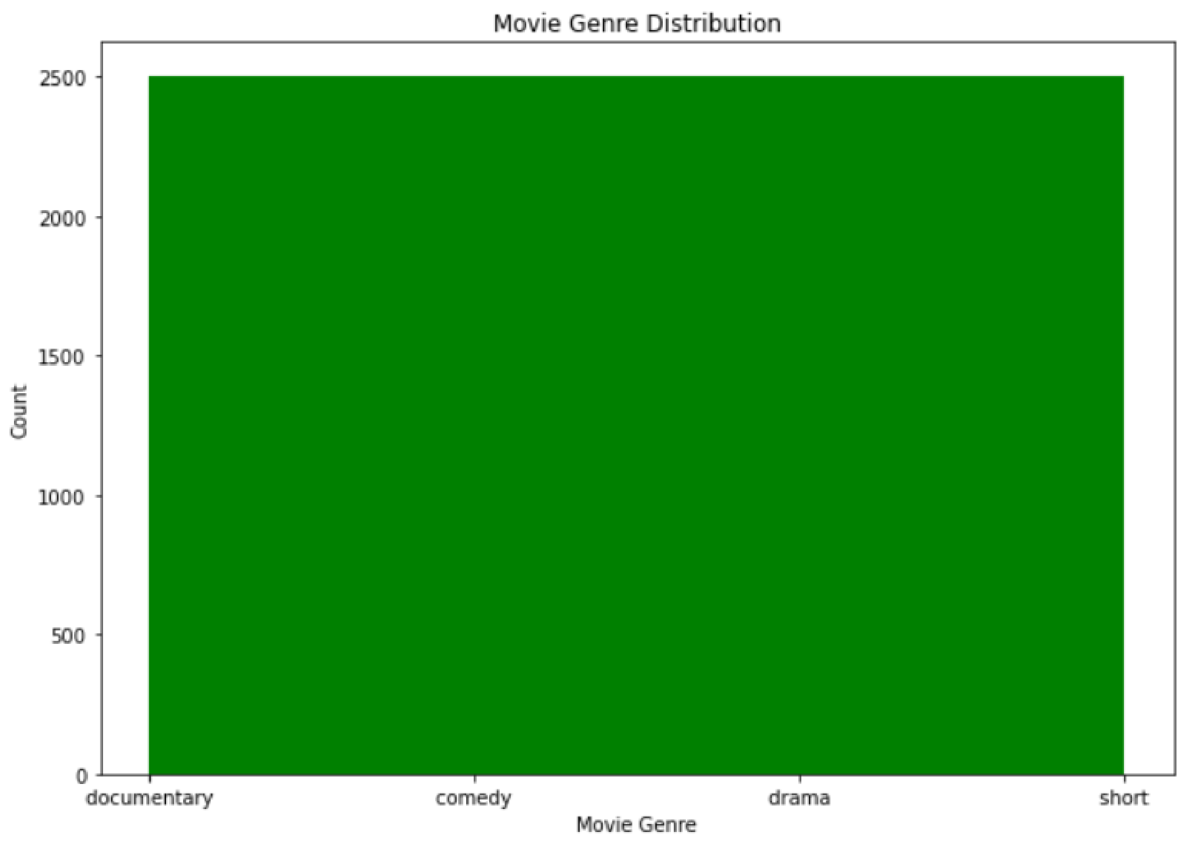
The “Genre Classification IMDb” dataset contains the description and genres of movies with an identifier id. The details of the attributes is as follows and the problem statement is to predict the genre of the movie from the provided movie descriptions given as part of test data.

- id - a unique identifier for each movie, in integer format
- genre - a specific genre that each movie belongs to, in text format
- description - a brief description for each movie, in text format

There is total 10,000 and 6,000 entries in the training and test dataset respectively. Since we are required to predict the genre of the movie, which is a categorical value, this is a classification problem.

2. Exploratory Data Analysis & Data Pre-Processing

- First, the given data is in CSV format. So, we need to read it from the file and convert it into a dataframe in python using “read_csv” function.
- After analyzing the training and test datasets, it is evident that there are no missing values to be handled.
- Moreover, on analysing the train data more closely, it was found that it contains equal number of records for each class to be classified as shown in below figure. Hence, there is no need to balance the instances of each class to get better performance.



- Now since we have textual data, we need to perform some pre-processing steps in order to make the data more interpretable and to improve the performance of the classification models.

3. Data Pre-processing

To clean the data we perform following pre-processing techniques.

- **Convert to lower-case:** Here we convert the description to lower-case for better interpretation.
- **Remove bad symbols**: Here we remove any bad symbols like “#”, “+” etc. from the descriptions.
- **Remove digits**: Here we remove the digits if any from the description since having numbers will not help better classify the genre of a movie.
- **Remove punctuations:** Here we remove all the punctuations from the descriptions as they are irrelevant for identifying the genre.
- **Remove stop words:** Here we remove commonly used words in English language from the descriptions like “the”, “a” etc. in order to make the model focus on important words.
- **Stemming:** Here we convert all the words in the description to it’s base word and remove their respective affixes from them. For example, the stem of the words eating, eats, and eaten is eat.
- **TF-IDF Vectorization:** Here we convert the textual descriptions into a matrix of features using Term Frequency – Inverse Document Frequency algorithm.

Thus, we pass on this pre-processed data to various classification models listed in the next section for the classification process. Following are some pre-processing techniques that we experimented with but these did not improve the model performance, but rather deteriorated it.

- **Lemmatization:** This is similar to Stemming but in this case it makes sure that the base words obtained as results are meaningful. But this technique did not work for our dataset.
- **LDA Vectorization:** Latent Dirichlet Allocation (LDA) is a classical way to do topic modeling. Topic modeling is unsupervised learning and the goal is to group different documents to the same “topic”. But even this technique did not work for our dataset.

4. Prediction Models

All the prediction models that are used for this project are supervised learning models/algorithms. Once the data is pre-processed, it is provided to the respective models and the results are predicting for the test dataset individually for each model.

One-vs-all Classifier has been used for all the below mentioned models. In this technique, the classifier breaks down a multiclass classification problem into small binary classification problems for each possible combination across all the classes and eventually combines the results for each one of them to solve the multiclass classification problem. This technique significantly improves the performance of all the models. Moreover, the models are fine-tuned using some techniques which are discussed below.

i. Stochastic Gradient Boosting (SGD)

- SGD classifier is used to implement regularized linear models with Stochastic Gradient Descent.
- As we know, Gradient Descent is an optimization algorithm that tries to minimize the prediction error by finding optimal weights. It uses whole training data to change the weights, but SGD considers only one random point at a time for that. Because of this, SGD is much faster than gradient descent. So, for large datasets, SGD is more preferred.
- Here in this project, to choose the loss and alpha that maximizes the accuracy, “Grid Search Cross Validation” technique with 3 folds is utilised.
- Thus, here ‘log’ and ‘0.0001’ values are chosen respectively for loss and alpha parameters of the model.
- For the above parameters, the accuracy score of SGD prediction model on Kaggle is **0.65850.**

ii. Logistic Regression

- Logistic regression is a statistical analysis method to predict a binary outcome, such as yes or no, based on prior observations of a data set. It can also be used for multiclass classification problems like this one.
- A logistic regression model predicts a dependent data variable by analyzing the relationship between one or more existing independent variables.
- Here in this project, the default parameters of Logistic Regression implementation in scikitlearn were used since it did not perform well for choosing the C and penalty parameters with Grid Search Cross Validation of 10 folds.
- Thus, the accuracy score of Logistic Regression prediction model on Kaggle is **0.65733**.

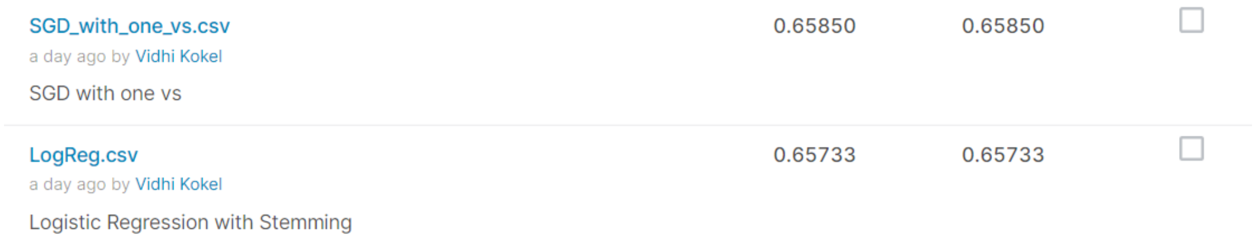
iii. Support Vector Machine (SVM)

- SVM is a supervised machine learning algorithm that can be used for both classification and regression challenges. However, it is mostly used in classification problems. Thus, we chose this model for our problem.
- In the SVM algorithm, each data item is plotted as a point in n-dimensional space (where n is a number of features) with the value of each feature being the value of a particular coordinate. Then, the classification is performed by finding the hyper-plane that differentiates the two classes very well.
- Here, in this project, we use linear SVM algorithm.
- Moreover, we utilized Grid Search Cross Validation technique to choose the best value for parameter C which turned out to be 10. Later we trained the model with this parameter and further predicted the results for test data.
- For the above parameters, the accuracy score of SVC prediction model on Kaggle is **0.65000.**

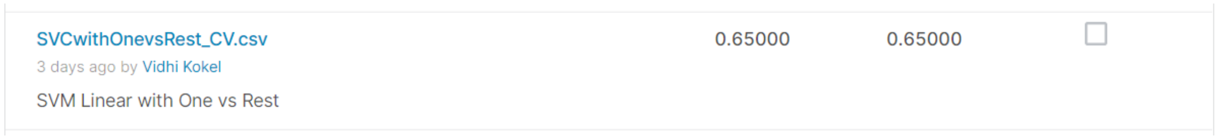
Thus, from the given dataset and the experiments performed, **Stochastic Gradient Descent (SGD)** model gave the best accuracy from all the models.

5. Kaggle Scores

i. Final Submissions



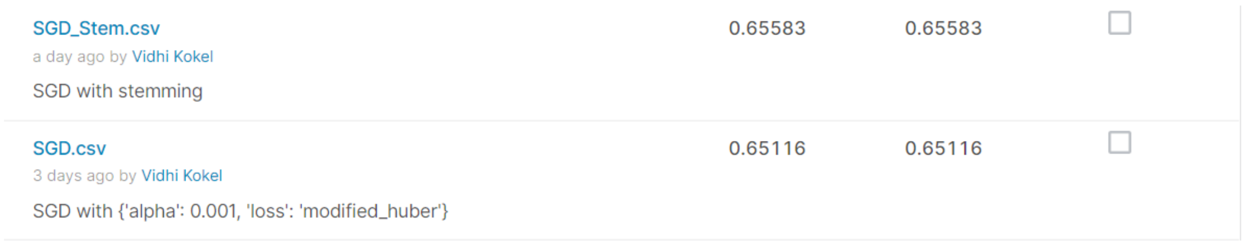
SGD_with_one_vs.csv a day ago by Vidhi Kokel SGD with one vs	0.65850	0.65850
LogReg.csv a day ago by Vidhi Kokel Logistic Regression with Stemming	0.65733	0.65733



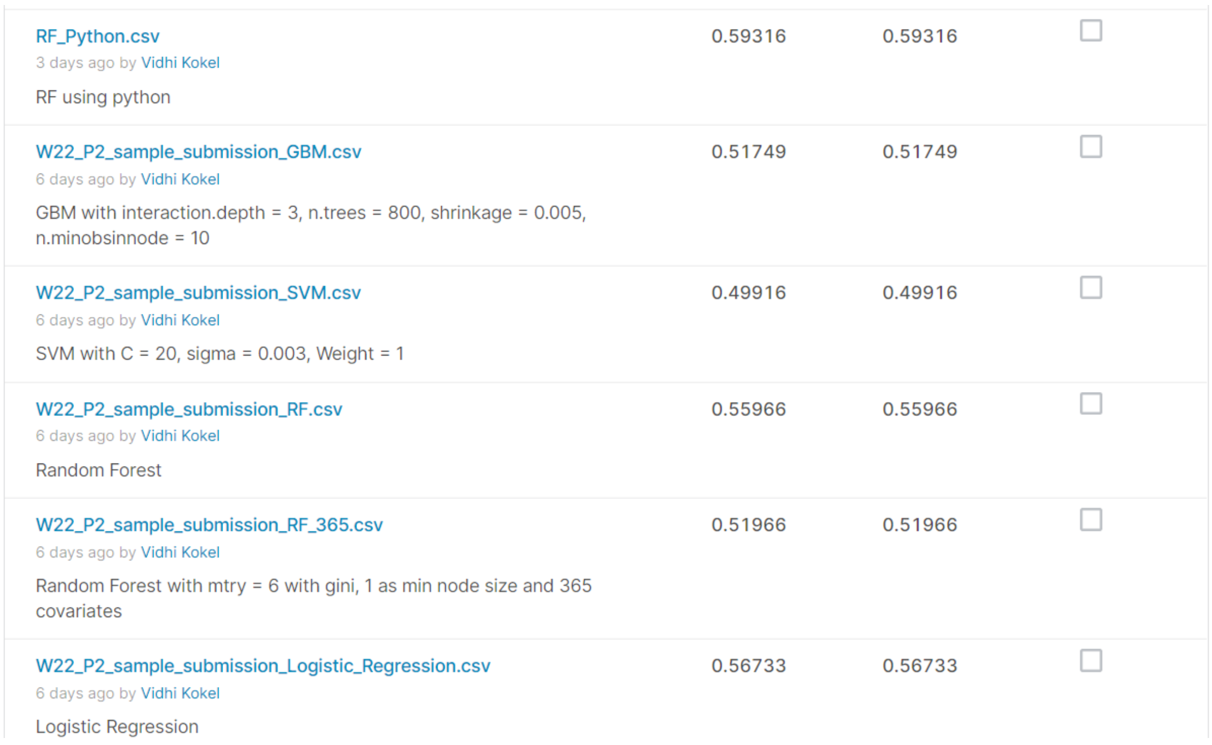
SVCwithOnevsRest_CV.csv 3 days ago by Vidhi Kokel SVM Linear with One vs Rest	0.65000	0.65000

ii. Other Attempts

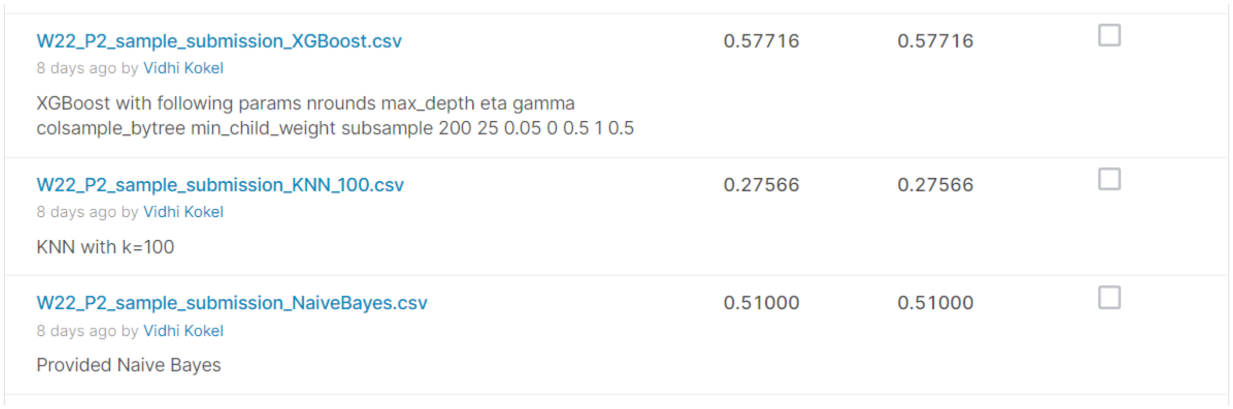
We also tried implementing other models like Random Forest, extreme gradient boosting, KNN and Naïve Bayesian in R technology but they did not yield competent accuracy. Moreover, we tried fine tuning the various parameters of the mentioned models and all their respective scores are as follows:



SGD_Stem.csv a day ago by Vidhi Kokel SGD with stemming	0.65583	0.65583
SGD.csv 3 days ago by Vidhi Kokel SGD with {'alpha': 0.001, 'loss': 'modified_huber'}	0.65116	0.65116



RF_Python.csv 3 days ago by Vidhi Kokel RF using python	0.59316	0.59316
W22_P2_sample_submission_GBM.csv 6 days ago by Vidhi Kokel GBM with interaction.depth = 3, n.trees = 800, shrinkage = 0.005, n.minobsinnode = 10	0.51749	0.51749
W22_P2_sample_submission_SVM.csv 6 days ago by Vidhi Kokel SVM with C = 20, sigma = 0.003, Weight = 1	0.49916	0.49916
W22_P2_sample_submission_RF.csv 6 days ago by Vidhi Kokel Random Forest	0.55966	0.55966
W22_P2_sample_submission_RF_365.csv 6 days ago by Vidhi Kokel Random Forest with mtry = 6 with gini, 1 as min node size and 365 covariates	0.51966	0.51966
W22_P2_sample_submission_Logistic_Regression.csv 6 days ago by Vidhi Kokel Logistic Regression	0.56733	0.56733



W22_P2_sample_submission_XGBoost.csv 8 days ago by Vidhi Kokel XGBoost with following params nrounds max_depth eta gamma colsample_bytree min_child_weight subsample 200 25 0.05 0 0.5 1 0.5	0.57716	0.57716	☐
W22_P2_sample_submission_KNN_100.csv 8 days ago by Vidhi Kokel KNN with k=100	0.27566	0.27566	☐
W22_P2_sample_submission_NaiveBayes.csv 8 days ago by Vidhi Kokel Provided Naive Bayes	0.51000	0.51000	☐