CSE 587 Assignment 3 Report

Team members:

Srikanth Ammineni
UB ID: samminen
UB person#: 50317818

Goutham Krishna Reddy Sagam
UB ID: gsagam
UB person#: 50313948

Hemanth Inakollu
UB ID: hemanthi
UB person#: 50316838

Kaggle Team name: Triple frontier

Environment

Spark version:

Preprocessing

Libraries used: findspark, pyspark
Functions used: regexp_replace**,** Tokenizer, StopWordsRemover

Preprocessing steps performed:

- Imported the train, test and mapping files using pandas and created spark dataframes out of them.
- Text cleaning was performed on the plot column for both train and test data using regex replace
- Plot column was tokenized using the Tokenizer function from pyspark.ml.feature,
- All the stop words were removed using StopWordsRemover function from pyspark.ml.feature ,
- True labels for each movie in Training data were converted into binary values using the RDD map function

Part-1 Basic model

Libraries used: pyspark, mllib
Functions used:
CountVectorizer, LogisticRegressionWithSGD, LabeledPoint

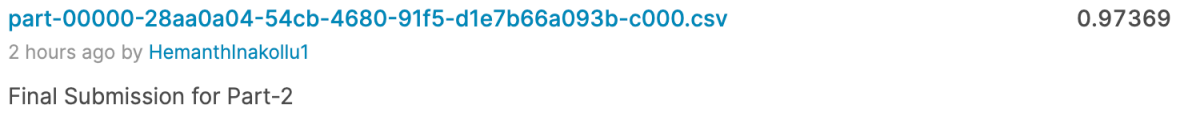
- Document-Term matrix was created using CountVectorizer for both Train and Test data
    - Hyper parameter, minDf = 0.05
- RDD Map function and LabeledPoint were used to convert the Features to the format that LogisticRegressionWithSGD function would expect.
- Looping through each genre, LogisticRegressionWithSGD model was trained and predictions were generated for test data.
    - iterations=30
- Predictions for each genre were consolidated and written to CSV file in the format Kaggle would expect.
- Below is the F1 score reported in Kaggle for part-1

Part-2 Use TF-IDF to improve the model

Libraries used: pyspark, mllib
Functions used:
HashingTF, IDF, LogisticRegressionWithSGD, LabeledPoint, Pipeline

- TF-IDF based feature engineering was implemented to further improve the model, we have used HashingTF and IDF functions from pyspark.ml.feature to implement this and was done for both Train and test data
    - Hyper Parameter, numFeatures= Same as number of features used for Part-1
- RDD Map function and LabeledPoint were used to convert the Features to the format that LogisticRegressionWithSGD function would expect.

- Looping through each genre, LogisticRegressionWithSGD model was trained and predictions were generated for test data.
    - iterations=30
- Predictions for each genre were consolidated and written to CSV file in the format Kaggle would expect.
- Below is the F1 score reported in Kaggle for part-2



Part-3 Custom Feature Engineering

Libraries used: pyspark, mllib
Functions used:
Word2Vec, LogisticRegressionWithSGD, LabeledPoint, Pipeline

- Word2Vec based feature engineering was implemented to further improve the model, we have used Word2Vec functions from pyspark.ml.feature to implement this and was done for both Train and test data.
- RDD Map function and LabeledPoint were used to convert the Features to the format that LogisticRegressionWithSGD function would expect.
- Looping through each genre, LogisticRegressionWithSGD model was trained and predictions were generated for test data.
- Predictions for each genre were consolidated and written to CSV file in the format Kaggle would expect.
- Below is the F1 score reported in Kaggle for part-3