Assignment 3: Predictive Analytics with Spark

Aswin Shakil Balasubramanian Ubit Name: aswinsha

Sanidhya Chopde Ubit Name: schopde

Shashank Raghunathan Ubit Name: raghuna2

Introduction

In this project, we are going to implement a movie genre prediction model using Apache Spark. It is a multi-label classification problem where we have to predict the all genres associated with the movie.

Part 1: Basic Model

- Here, we start by creating spark data frame for train, test and mapping data.
- We read the csv files using the Pandas library and then start pre-processing this data.
- First, we apply the regex tokenizer on train and test data, which takes text in the plot column and breaks it into individual terms.
- Then, we remove stop words from the train and test data using the StopWordsRemover.
- Then we use CountVectorizer to create the term document matrix and apply all the transformations to the train data and the test data.
- We have enlisted the genres with their corresponding encodings and then we have written a function that maps the genre with it's number.
- After that we have a written a function that converts the mapped column to the label column which has the value 1 if that genre is associated with the movie plot or else it is 0.
- Then we are creating 20 logistic regression models, that is one model for each genre and training them to predict which plot belongs to what all genres.
- In the final output we display the movie id with its corresponding prediction.
- Part 1 F1 Score: 0.97180

Part 2: Using TF-IDF to improve the model

- Here too, we start by creating spark data frame for train, test and mapping data.
- We read the csv files using the Pandas library and then start pre-processing this data.
- First, we apply the regex tokenizer on train and test data, which takes text in the plot column and breaks it into individual terms.
- Then, we remove stop words from the train and test data using the StopWordsRemover.
- We will now use TF-IDF to improve our model. We will use HashingTF to transform our data.
- HashingTF takes sets of terms and converts those sets into fixed length feature vectors.
- We will then use IDF, which takes vectors created by HashingTF and scales each column.
- We will apply all of these transforms on our train and test data.
- We then enlist the genres with their corresponding encodings and then we have written a function that maps the genre with it's number.
- After that we have a written a function that converts the mapped column to the label column which has the value 1 if that genre is associated with the movie plot or else it is 0.
- Then we are creating 20 logistic regression models, that is one model for each genre and training them to predict which plot belongs to what all genres.
- In the final output we display the movie id with its corresponding prediction.
- Part 2 F1 Score: 0.97436

Part 3: Custom Feature Engineering

- Here too, we start by creating spark data frame for train, test and mapping data.
- We read the csv files using the Pandas library and then start pre-processing this data.
- First, we apply the regex tokenizer on train and test data, which takes text in the plot column and breaks it into individual terms.
- Then, we remove stop words from the train and test data using the StopWordsRemover.
- We will then transform our train and test data using Word2Vec estimator. It takes sequences of words representing models and trains a Word2Vec model. The model then maps each word to a unique fixed size vector.
- We then enlist the genres with their corresponding encodings and then we have written a function that maps the genre with it's number.
- After that we have a written a function that converts the mapped column to the label column which has the value 1 if that genre is associated with the movie plot or else it is 0.
- Then we are creating 20 logistic regression models, that is one model for each genre and training them to predict which plot belongs to what all genres.
- In the final output we display the movie id with its corresponding prediction.
- Part 3 F1 Score: 1.00000

Bonus:

- For this part of the assignment, we submitted our predictions in Kaggle for all the models and got the highest score of 1.00000 for the 3rd part which had the Word2Vec model.

Video URL :

https://www.youtube.com/watch?v=CWzaroReHgM

References:

- https://spark.apache.org/docs/2.1.0/ml-features.html
- https://spark.apache.org/docs/latest/
- Lecture Videos