MINI-PROJECT 1: YELP BUSINESS RATING PREDICTION USING PANDAS AND SKLEARN

Version 1.0

CSC215, Fall 2018

Sept 24th, 2018

Prepared by: Chandini Nagendra

Siddharth Chittora

Contents

1. Problem Statement

Implementation of machine-learning based system to predict a business's star rating using all the reviews of that business and review count. This problem is a regression problem where the output predicted is a continuous value, the star rating for the business.

2. Design

2.1. Data Preprocessing

- Json files used are Business.json, Review.json
- Converted json files to TSV with only selected columns
- Loaded the TSV file to pandas dataframe
- Group all the reviews by each business and create a new dataframe, where each line is a business with all its reviews
- Join dataframes based on the value of a common column, we joined Business dataframe with review dataframe based on business_id column
- Clean the reviews text data by removing stop words, punctuations and white spaces.
- Convert text data into TFIDF vectors for Feature Extraction.
- Normalize the review count field so it is comparable
- Adding the normalized count column to the result of the TFIDF vectorizer.
- Split the data into train and test data (80-20 split ratio).

Now the data is ready to be used for training the model

2.2. Supervised Learning Framework

2.2.1. Problem:

The system is trying to predict star rating of a business based on the review and the review count

2.2.2. Features used as Input:

- Business_ID
- Review Count
- Review Text
- Postal Code
- Categories

2.2.3. Features used as Output:

- Business star rating

2.2.4. Models Used for Data Training

- Linear Regression Model
- Logistic Regression Model

- Nearest Neighbor
- Support Vector Machine
- Multinomial Naïve Bayes

2.3. Testing

To compare the predicted stars with the actual stars we are printing the list of business to show both the predicted stars and the actual stars.

3. Task Division

3.1. Chandini Nagendra:

- Data Preprocessing
- Linear Regression
- Nearest Neighbor Model
- Multinomial Naïve Bayes
- Report

3.2. Siddharth Chittora

- Data Preprocessing
- Linear Regression
- Logistic Regression Model
- Support Vector Machine
- Report

Discussed together on how to improve the model and came up with the following solution

4. Project Reflection

- Data Preprocessing took most of the time, in the process learnt to use the libraries, numpy, pandas, scikit learn, nltk
- We learnt that MNB does not accept negative normalized values, so we normalized values using formula "**new value = (old value – min column value)/(max col. value – min col.value)**"which only gives positive output
- We used label encoded stars to train all the models, but the linear regression model gave incorrect results (negative result) therefore we did not perform label encoding on stars for the linear regression model. For linear regression it requires continues values. if we performed label encoding it will give us discrete values and hence we think we should not perform label encoding on Linear Regression.
- We also noticed that the KNN gives the best result with one neighbor(k=1), we tuned the model for five neighbors(k=5) and it gives us an accuracy of 4% as opposed to 24%
- For SVM model we changed the kernel parameter from default that is RBF(Radial Basis Function) to linear. it improved the accuracy from 27% to 32%
- when trying to extract the features from categories using the entire dataframe, we were getting type error on the same dataframe which worked with the earlier models, so we had to cut the dataframe to 10000 rows.

- Performed one hot coding without extracting the primary word in categories. This created too many unique categories which would not have been useful for increasing the accuracy of the model.
- Extracted a primary category from each row of category and performed one hot coding.

5. Additional Features

Additional features used: Categories and postal code

- Tried using **neighborhood** column as a feature to train the models, however, neighborhood had too many NULL values, which might have affected the model accuracy. Hence, we decided to use postal code as feature instead.
- Used additional columns like the **postal code** to improve the accuracy, here we also performed one hot coding on the postal code and used the result to train the model and test. And this improved the accuracy of SVM model by 10%
- Used One hot coding on **category** column in business dataframe. Here we extracted the primary category from each row of category and performed one hot coding on this column. Here we gained an accuracy rate of 3-4% on each model