Ben Noriega

CS777

2.22.23

Term Project

Project Repo: https://github.com/bennordi/cs-777-term-project-bennordi

Dataset

The dataset is a CSV file found on Kaggle under [Spam Text Message Classification](https://www.kaggle.com/datasets/team-ai/spam-text-message-classification). It contains 5572 rows of two columned data: Category and Message. Category can simply be described as the ‘class’ or ‘label’ of this dataset as it has a simple binary value: ham or spam. Spam represents the text in the ‘message’ column is from a text deemed to be spam. The opposite ‘ham’ represents the text was deemed to be, not spam.

Research Question

This is a simple classification dataset, which we will leverage PySpark and various libraries to analyze text data to build various machine learning models that will predict `ham or spam` based on the given text data. We will then review each model and its results. There are 5572 rows of data to analyze, and luckily the data was preformatted so that `ham/spam` are the only values in the category column, and the message column will need to be formatted into features columns for us to build models off of. The main need for preprocessing was removing non-letters from the text, lemmatizing the text, and then vectorizing the data (with MLlib’s pipeline using Tokenizer(), StopWordsRemover(), CountVectorizer(), IDF(). I was going to use ChiSqSelector() but we learned in Assignment 5 that with larger datasets this can be very computationally expensive, and knowing that the goal of this assignment is to be scalable, I opted to not perform this selection.

This data will be split into training and test sets (70/30 split) and will be seeded for consistency of the split. The classification models will be trained on the same exact training data and will be tested with the same test dataset. Then during the results, I can do some anecdotal tests for messages I create to test out our models.

Machine Learning Model

Used 6 different machine learning models from PySpark's MLlib library, and fit the model with our training dataset, predicted on our test dataset. Output of my local machine run can be found on output.md.

- Logistic Regression

- Linear SVC

- Naive Bayes classifier

- Decision Tree classifier

- Gradient-Boosted Tree classifier (GBT)

- Random Forest Classifier

Results

Timing of each step in the process:

* Loading data, preprocessing, vectorization: 26.433 secs
* Logistic Regression: 7.879 secs
* Linear SVC: 3.963 secs
* Naïve Bayes classifier: 2.631 secs
* Decision Tree classifier: 6.258 secs
* Gradient-Boosted Tree classifier: 15.095 secs
* Random Forest classifier: 3.701 secs
* Total time: 65.961 secs

**Logistic Regression:**

|  |  |  |
| --- | --- | --- |
|  | predicted(spam) | predicted(ham) |
| actual(spam) | 1431 | 16 |
| actual(ham) | 20 | 201 |

Accuracy: 0.978

Precision: 0.910

Recall: 0.926

F1: 0.918

**Linear SVC:**

|  |  |  |
| --- | --- | --- |
|  | predicted(spam) | predicted(ham) |
| actual(spam) | 1428 | 19 |
| actual(ham) | 15 | 206 |

Accuracy: 0.980

Precision: 0.987

Recall: 0.990

F1: 0.988

**Naïve Bayes:**

|  |  |  |
| --- | --- | --- |
|  | predicted(spam) | predicted(ham) |
| actual(spam) | 1372 | 75 |
| actual(ham) | 12 | 209 |

Accuracy: 0.948

Precision: 0.948

Recall: 0.991

F1: 0.969

**Decision Tree:**

|  |  |  |
| --- | --- | --- |
|  | predicted(spam) | predicted(ham) |
| actual(spam) | 1432 | 15 |
| actual(ham) | 73 | 148 |

Accuracy: 0.947

Precision: 0.990

Recall: 0.951

F1: 0.970

**Gradient-Boosted Tree:**

|  |  |  |
| --- | --- | --- |
|  | predicted(spam) | predicted(ham) |
| actual(spam) | 1431 | 16 |
| actual(ham) | 59 | 171 |

Accuracy: 0.960

Precision: 0.989

Recall: 0.966

F1: 0.977

**Random Forest:**

|  |  |  |
| --- | --- | --- |
|  | predicted(spam) | predicted(ham) |
| actual(spam) | 1447 | 0 |
| actual(ham) | 193 | 28 |

Accuracy: 0.884

Precision: 1.000

Recall: 0.882

F1: 0.937

**Local run output can be found on <output.md>**

Conclusion:

The most shocking result from this study was the precision measure of the Random Forest classifier, meaning that it correctly predicted all actual spam texts correctly. It did not do as good in terms of ham prediction; however, this is likely due to the balance of the spam/ham distribution between not only the base dataset itself (87% of the data was classified as ‘ham’). The random split may have contributed to this; however, the Linear SVC model had a much higher accuracy with a very high precision measure as well. This may indicate that the Random Forest classifier caters more towards the positive class in the data, but it is hard to say based on the format of the data as raw text values.