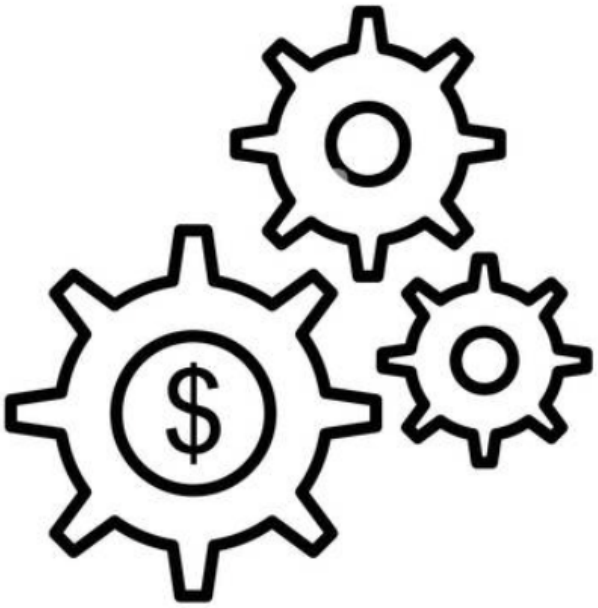

CUNY Hunter College
CSCI 49376 Big Data Technology

Report by : Iris Shakya

Instructor: Professor Lei Xei, Ph.D

Data:

https://www.kaggle.com/datasets/team-ai/spam-text-message-classification

Reference:

1) https://www.irjmets.com/uploadedfiles/paper//issue_5_may_2022/23528/final/fin_irjmets1652900919.pdf

2) http://www.textbook.ds100.org/ch/17/ms_cv.htm

3) http://www.textbook.ds100.org/ch/19/classification_log_reg.html

4) https://goodboychan.github.io/python/datacamp/pyspark/machine_learning/2020/08/10/03-Classification-in-PySpark.html#Training-a-spam-classifier

5) https://ai.plainenglish.io/build-naive-bayes-spam-classifier-on-pyspark-58aa3352e244

6) https://datascience-enthusiast.com/Python/PySpark_ML_with_Text_part1.html

7) https://www.youtube.com/watch?v=32q7Gn9XjiU&ab_channel=MSparks

8) https://www.youtube.com/watch?v=jG5hfyyU2Jk&ab_channel=DataScienceforEveryone
For tokenization and some preprocessing.

Spam or Ham

Spam emails or messages belong to the broad category of unsolicited messages received by a user. Spam occupies unwanted space and bandwidth, amplifies the threat of viruses like trojans, and in general exploits a user's connection to social networks. Spams are also used in various DDoS and DoS attacks by phishing. Most of the spam filters out in the industry are content-based instead of ISP because the bulk of text data with specific keywords are sent out through emails and sms services. I take the statistical approach with algorithms followed in class to try my take on this model training.

I take the data from Kaggle which has about 5576 samples with the binary label, spam or ham. Spam pointing to unsolicited spam messages and ham to non-spam labels. I take the bag-of-words approach to this problem by implementing HashingTF and IDF model. The model outputs unique id and idf for each word/token. The word is numerised because the NLP ML algorithm does not understand text data. The extracted information is feature engineered through pipeline, we do not require context of text here; only number of occurrences. Instead of using individual words for hashing, I use grab couple words to better understand the pattern. I.e. 'win voucher of $50' vs ['win', 'voucher' '$50']. The separated and grouped words give out a significant pattern in classifying spam texts.

Since this is categorical data, I execute supervised learning model using Decision Tree. Decision trees are widely used since they are easy to interpret, handle categorical features, extend to the multi-class classification, do not require feature scaling, and are able to capture non-linearities and feature interactions.

Data splits is done 5 times with different seed to input in the k-fold Cross Validation and to train and test data, instead of nested CV. Split is done in the ratio of 0.8:0.2 on train and test.

```
Precision_1: 0.8780
Recall_1: 0.8863

Precision_2: 0.8705
Recall_1: 0.8718

Precision_3: 0.8999
Recall_1: 0.9026

Precision_4: 0.8954
Recall_1: 0.9031

Precision_5: 0.8977
Recall_1: 0.9015
```

```
The average for Precision is: 0.8883
The average for Recall is: 0.8931
```

```
cv = CrossValidator(estimator=DecisionTreeClassifier(),\
    estimato (variable) cv: Any
    evaluato (variable) cv: CrossValidator ator(), \
    seed=645
cv_model_1 = cv.fit(train_1)
cv_model_2 = cv.fit(train_2)
cv_model_3 = cv.fit(train_3)
cv_model_4 = cv.fit(train_4)
cv_model_5 = cv.fit(train_5)

print('CV_1: %0.4f' %cv_model_1.avgMetrics[0])
print('CV_2: %0.4f' %cv_model_2.avgMetrics[0])
print('CV_3: %0.4f' %cv_model_3.avgMetrics[0])
print('CV_4: %0.4f' %cv_model_4.avgMetrics[0])
print('CV_5: %0.4f' %cv_model_5.avgMetrics[0])
```

✓ 1m 42.2s

```
CV_1: 0.8648
CV_2: 0.8710
CV_3: 0.8621
CV_4: 0.8683
CV_5: 0.8658
```

With logistic Regression, we do reach a higher level of accuracy, however the weight is heavily dependent on single feature which creates biasness. Decision Tree algorithm failed to reach precision of 90% here. One reason for the low performance can be the range of different features that it had to take(we had set numFeatures=6000, also numFeat ~ size of document).

Decision trees do not assume independence of the input features and can thus encode complicated formulas related to the relationship between the variables whereas logistic regression treats each feature independently. However, decision trees are more likely to overfit the data since they can split on many different combinations of features whereas in logistic regression we associate only one parameter with each feature.

We can achieve better results by using Ensemble methods like randomised forest, adaboost, etc which incorporates bagging, boosting, stacking, and randomising.

Classifier	Accuracy
Ensemble Decision Tree (# of tree = 25)	96.40
Adaboost	95.00
Stacking	93.80
SVM	93.40
Bagging	92.80
Decision Tree	92.58
Neural Network	90.80
Naive Bayes	89.57
Nearest Neighbor (k = 5)	89.40

Source: https://courses.cs.washington.edu/courses/cse573/04au/Project/mini2/Ravi&Indri/spamfilter_ravi_indri.pdf