CSE 621

Comparing of Classifiers in Spam Detection Maria Bocharova

Due 05/03/2022

## Objectives

The goal of this project is to determine which supervised learning classifiers are best able to classify strings of text as spam or not. The Decision Tree, Logistic Regression, Support Vector Machine (SVM), and Naive Bayes methods of classification are trained against a dataset of spam and non-spam (ham) texts so that they can be used to test the validity of user inputted texts.

The data original was obtained from UCI website ([data source](https://archive.ics.uci.edu/ml/datasets/SMS%2BSpam%2BCollection)) in a csv format. Ot contains around 5500 samples with one message per line.

*Data Statistics:*

The dataset contains 5572 records.

The dataset has two attributes ( message body and spam classification).

Desired outputs will contain the output of each classifier with accuracy of the model.

## Background

In this project multiple techniques learned during CSE 621 course were used. First uploaded data was preprocessed using tokenization, removing of stop words, stemming and vectorization methods. Then several classification models were applied. The classifiers used in comparing classification results came from the CSE 621 course, these being Decision Tree, Logistic Regression, and Gaussian Naive Bayes classifiers.

Tokenization is splitting up larger text ( in this example - strings) into smaller subsets known as tokens. It was done using the word\_tokenizer function in the NLTK library. Then stop words were removed. Stop words are any word in the stop list that is predetermined by the python NLTK library. There is no universal list for stop words but mostly they are most commonly used words that do not bring significant weight to data analysis and machine learning models.

Third step was stemming. It is needed to analyze meaning behind words - stemming function uses stem of the word to produce morphological variants of a root/base of each word, and reduce them to their ‘stem’. Next and final step was vectorization, which converts text from bitmap to vectors. Sklearn CountVectorizer was used which converts text in the v1 column to a matrix of token counts. It takes given text into a vector on the basis of the frequency of each word.

After the preprocessing was done three classification functions were constructed: Decision Tree, Naive Bayes and Logistic Regression. Decision Tree creates a classification model by building a decision tree. Each node represents an attribute and each branch from that node corresponds to one of the possible values for that attribute. Decision trees, however, tend to overfit and provide lesser accuracy than other models. Therefore, it was expected that the decision tree model will be the least precise among the three models that were chosen.

The next model was Naive Bayes. Naive Bayes is one of the simplest but most accurate classifiers. It is a probabilistic type of classifier, meaning it predicts based on the probability of the object. It is based on Bayes theorem, which describes the probability of the event based on previous knowledge of conditions that might be related to that event. As for this project vectorization(divides text into vectors based on the frequency) was used, it was assumed that Naive Bayes would provide the most accurate results.

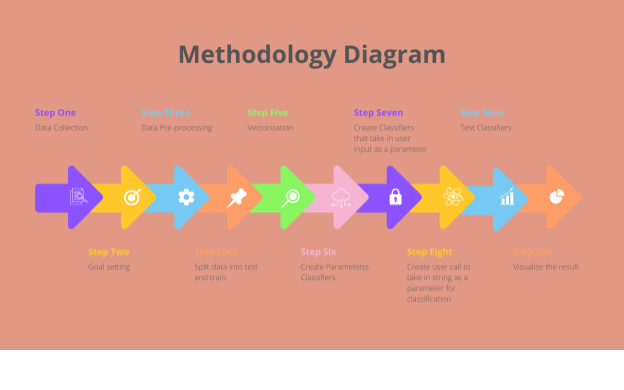
The last model used was Logistic Regression. Logistic regression is a fundamental classification technique, and part of linear classifiers, meaning it makes decisions based on the value of a linear combination of the given characteristics.

Logistic Regression is a process of modeling probabilities of a discrete outcome given an input variable. It is usually used when dependable variables are categorical.

To determine classification scores accuracy and ROC AUC were calculated.

Accuracy determines differences between actual and predicted models, and ROC curve shows performance of the classification model measuring the entire area under its curve.

## Methodology



Step 1.

The dataset selected was the SMS spam collection dataset. It contains just two columns of data. One column has a string of text, and the other classifies the string of text as spam or not spam (called ham).

Step 2.

Set the goal. The goal was two-fold: to see how three classifiers compare to each other when classifying on samples of the same dataset, and to see if we can get the classifiers to determine a user’s input as ham or spam.

Step 3.

Determine and apply preprocessing techniques. Main techniques used: tokenazation, stemming, stop words removal.

Step 4.

Processed data was split intp test and train sets using test\_train\_split function.

Step 5.

Vectorization was applied on the split datase.

Step 6.

Three base parametless classifier functions were written. Functions didn’t take in any parameters but their train outputs were later used to classify and predict accuracy of user input.

Step 7.

Similar classification functions were created. They took in user input as a parameter, which was preprocessed and vectorized inside the function.

Step 8.

User call was created to take an input.

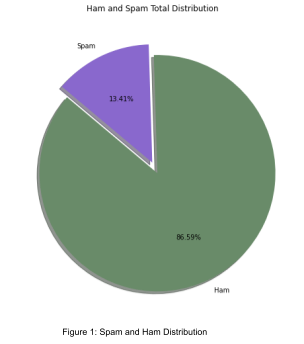
Step 9.

Classifiers with parameters were run. They returned classification of the string followed by accuracy and auc roc.

Step 10.

Graphs were created to visually represent the results.

## Experimental Results



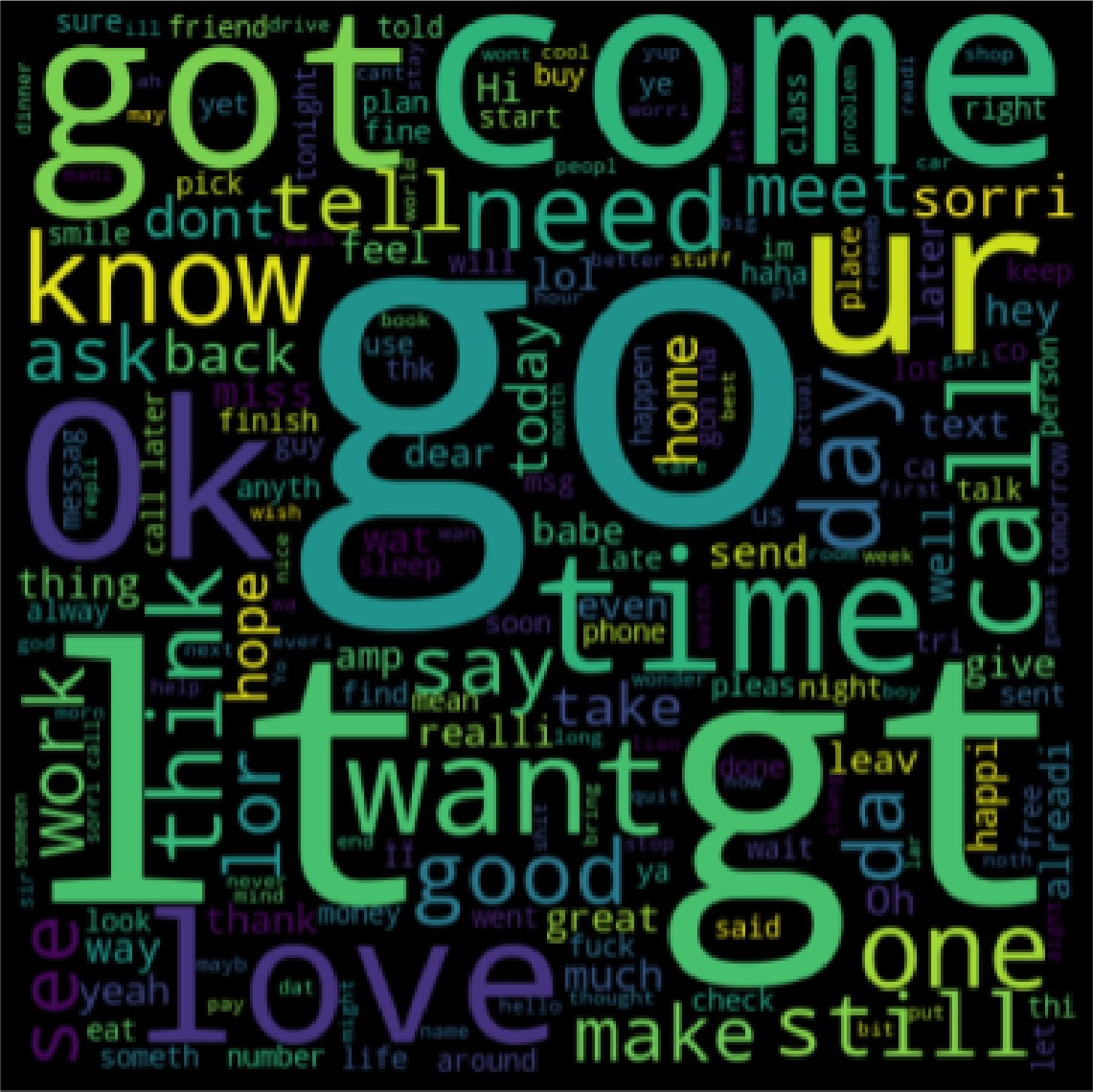
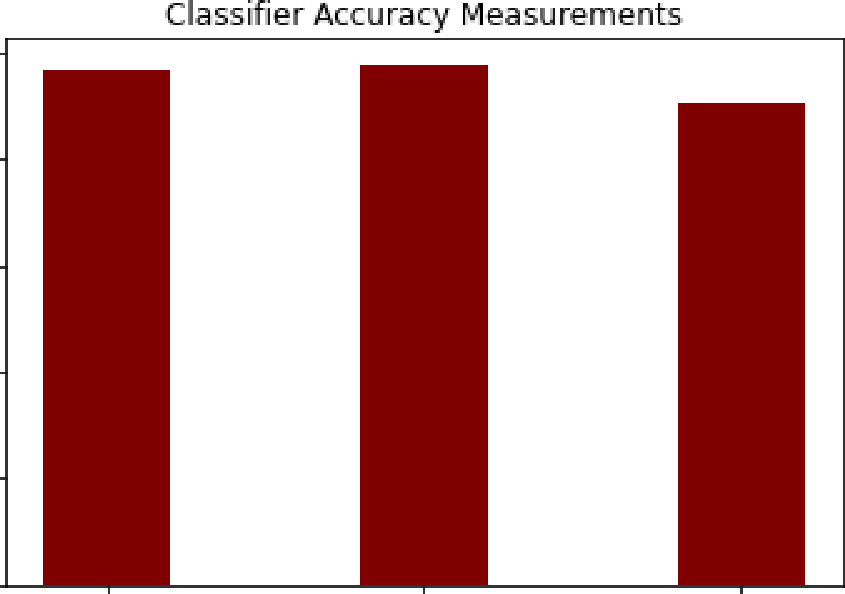


Figure 3: **MCKSI Gammon** Ha11111 Frequencie,s Won:J:Clood

100



80

"""''

"

E<)

,g:-

l':' 40

E:

*;J.*

*20*

0

Decision Tree Logiis ic Regression Classifiers

Naive Bayes

Figure 4: Classiner Accuracies **for** Dataset Samples

I!::

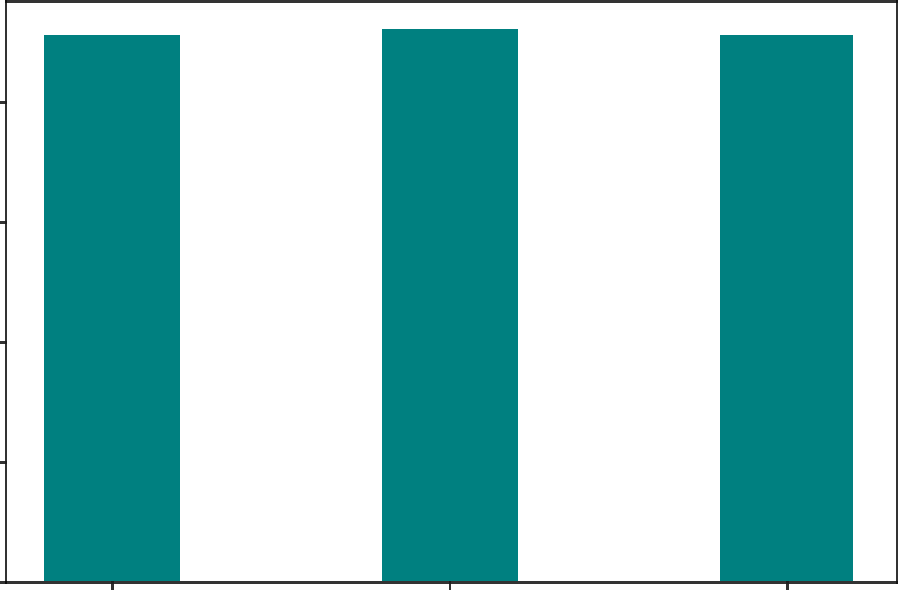
0u

vl

u

*<[*

Classiifier ROC AUC Scores

BO

50

u 40

0

*et:*

20

0

Decision Tree Log1istic Regressio11 Classifiers

Naive Bayes

**Figure 5: Classirfier ROC AUC Scores for Dataset Samples**

"'

"'

C:

**t;:**

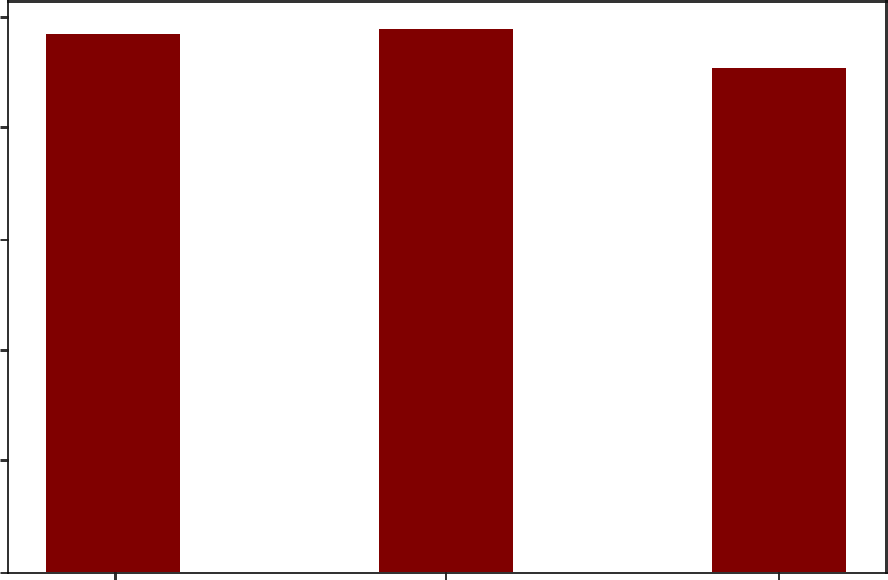
100

80

f:O

Classifier Accuracy Measmements

>,



u

!!:! 40

;;,

u

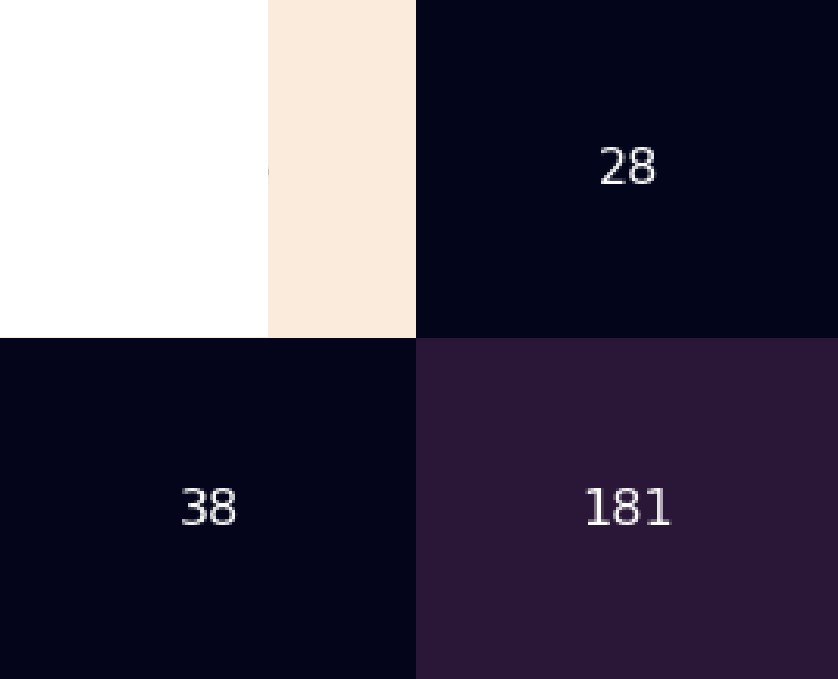
20

0

Decision Tree Log isbic Regressio11 Classifiers

Naive Bayes

**Figure 6: Classifier Accuracies for User Input**

F**lo.** c'll

|  |  |  |
| --- | --- | --- |
| IDeci  E  'll  a.  *"'*I  Ill  (]11 -I-ii  ..c :il  m | ision Tree Confusion Matrix  1425 | -1400  1200  1000  800 |
| E  I  'll |  | 600  4100 |
| **::::i**  **-Iu-ii** |  | -200 |

itl

predicted-spam pr,edic:ted-ham

# Prediicted laibels.

Figure 7: Decis.ton Tree Contusion!Matrix

# E

**a**'ll**.**

*"'*

I

Ill **'ll**

...;... **::::i**

**..**(]**c**11 -I-ii

# -1200

1000

800

m

F**lo...**

**u**ro

E

.c**'ll**

I

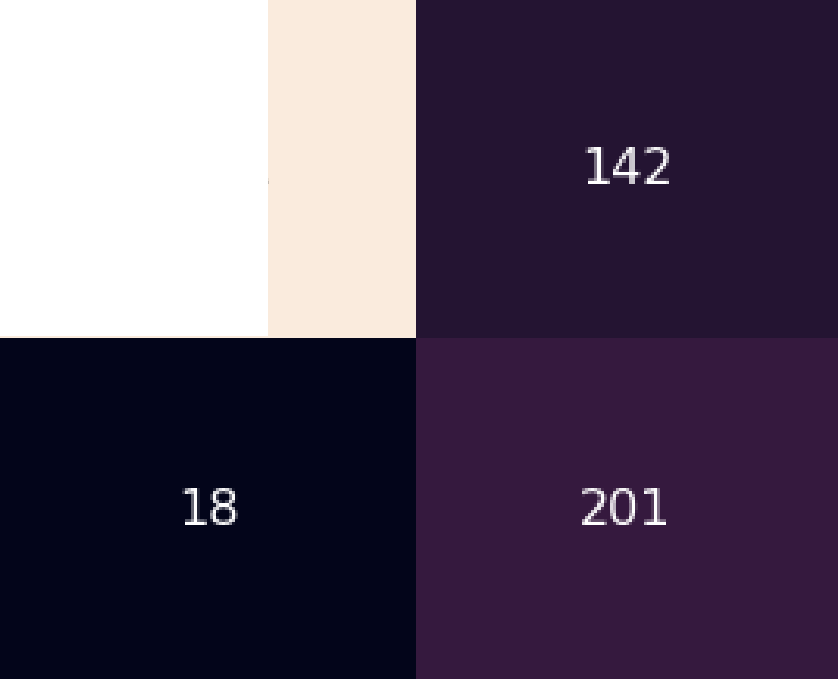
'll

**::::i**

**-Iu-ii**

itl

predicted-spam pr,edic:ted-ham



INaive IBayes Confusion Matrix

1311

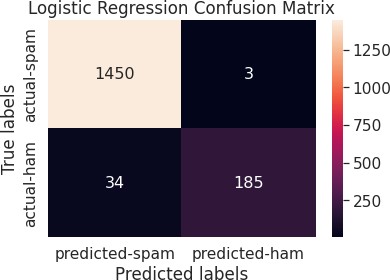
# Prediicted Iaibels.

# 600

4100

# 200

Figure 8: Naive Bayes Confusion Matrix





|  |  |  |  |
| --- | --- | --- | --- |
|  | Obvious Ham | Obvious Spam | Ambiguous |
| Decision Tree | Ham | Spam | Ham |
| Logistic Regression | Ham | Spam | Ham |
| Naive Bayes | Ham | Spam | Spam |



## Analysis of Results

It can be seen through the experimental results that all three of the classifiers are fairly accurate in a vacuum. When ham or spam classification is obvious, the classifiers seem to be correct every time. When the classification is more ambiguous, the Decision Tree and Logistic Regression classifiers tend to lean towards ham while the Naive Bayes classifier tends to lean towards spam. This is likely because Naive Bayes takes too strong of assumptions towards spam, so spam is easier for it to choose.

## Conclusion

All three classifiers in this project do fairly well when detecting ham or spam, even when the text being classified is not obviously one class or the other. Gaussian Naive Bayes is consistently the least accurate while Decision Tree is consistently the most accurate. But even Gaussian Naive Bayes is still incredibly accurate, at above a 90% accuracy rating on average. Vectorization was the strongest preprocessing aspect for the dataset text. While attempting to find a solution, the team found that vectorization gave the classifiers the strongest accuracy by far, along with the other text cleaning tasks. While the classifier accuracies are above 90%, more preprocessing could have been done to bring up the accuracies even further. If the goal was to determine a single classifier to use in spam detection, the team would likely select the Decision Tree method.