FLIP ROBO

EMAIL SPAM DETECTION PROJECT

Submitted by:

Anchal Awasthi

ACKNOWLEDGMENT

I am thankful to Flip Robo Technology for providing me an opportunity to execute this project.

I express my gracious gratitude to our project guide **Khushboo Garg,** SME of Flip Robo Technologies, for her valuable assistance. I am very thankful for her encouragement that made project successful.

INTRODUCTION

- ## Business Problem Framing

Understanding the problem is a crucial first step in solvingany machine learning problem. Our aim is to build a model that will predict and identify whether an email is a spam email or a ham email. This is called Spam Detection, and it is a binary classification problem. Spam Emails cannot be ignored and performing spam classification is important as detecting unsolicited and unwanted emails, we can prevent spam messages from creeping into the user's inbox, thereby improving user experience.

- ## Conceptual Background of the Domain Problem

E-mail is a serious communication tool, on daily basis a person receives 100s and 1000s of email, out of which spam emails take up the larger space, which can be misleading and can be a trap to lose valuable data.

- ## Motivation for the Problem Undertaken

Due to Exponential Growth of online user's individual email accounts are generally flooded with heaps of emails. Most of the time it will be a tedious job to individually read and classify these emails based on the desired classification protocol. This project aims to ease the process and classify spam and ham emails into separate categories.

Analytical Problem Framing

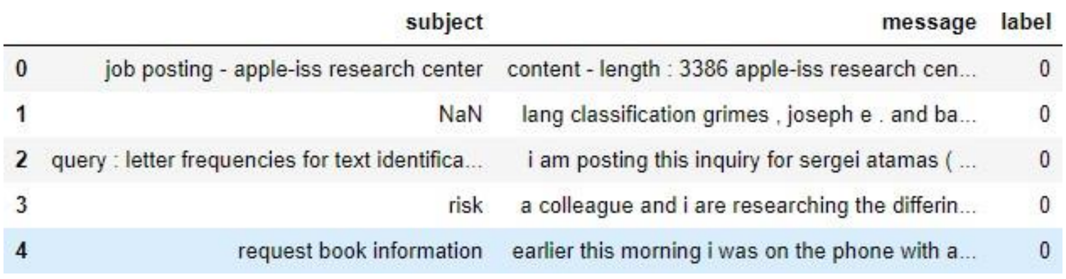
- Mathematical/ Analytical Modeling of the Problem

    1. Data was cleaned by removing punctuations, white spaces, and special characters. Email ids were replaced with keyword 'email' and phone numbers were replaced with keyword 'number'.
    2. Using the stop words package, all stop words were removed along with some other values which occurred often but did not have any meaning.
    3. Tf-idf vectoriser was used to convert text to vectors and weights were allocated for each text.
    4. Metrics like accuracy, and f1 score was used to evaluate the model's efficiency.

- Data Sources and their formats

    We will be using the messages.csv file t provided by the Fliprobo technologies, a dataset that contains 2893 emails, of which 481 are spam.

    1. A csv file containing subject of the mails, message content and its label was used. Label 0 represent non spam emails and 1 represented spam emails.
    2. It had 2893 rows and 3 columns.



	subject	message	label
0	job posting - apple-iss research center	content - length : 3386 apple-iss research cen...	0
1	NaN	lang classification grimes , joseph e . and ba...	0
2	query : letter frequencies for text identifica...	i am posting this inquiry for sergei atamas (...	0
3	risk	a colleague and i are researching the differin...	0
4	request book information	earlier this morning i was on the phone with a...	0

fig 1 : Sample Dataset

- Data Preprocessing Done

  Data usually comes from a variety of sources and often in different formats. For this reason, transforming your raw data is essential.

  1. It was assumed that label 0 was non spam email and label 1 was a spam email based on the messages given.

  2. Subject column had few values missing, hence dropped the column.

  3. Data cleaning - This phase involves the deletion of words or characters that do not add value to the meaning of the text. Some of the standard cleaning steps are listed below:

     - Lowering case
     - Handling of special characters
     - Removal of stopwords
     - Handling of hyperlinks
     - Handling of numbers

  4. Using the stopwords package, all stop words were removed along with some other values which occurred often but did not have any meaning.

  5. Tf-idf vectoriser was used to convert text to vectors and weights were allocated for each text.

- Data Inputs- Logic- Output Relationships

  1. The data was cleaned, and all stop words were removed. EDA was performed by creating word clouds that helped to identify frequently occurring words.
  2. The message column was converted into vectors and was used as an input.
  3. Various classifiers like Logistic Regression, Multinomial Naive Bayes, Random Forest Classifier, SVC, Decision Tree Classifier, and KNeighbors Classifier were used to train the model.
  4. Accuracy and cross val scores of all the classifiers were compared and at the end, AdaBoost was predicting the messages with high accuracy (98.02%).

Model/s Development and Evaluation

- Identification of possible problem-solving approaches :

The dataset was then split into training and testing dataset.

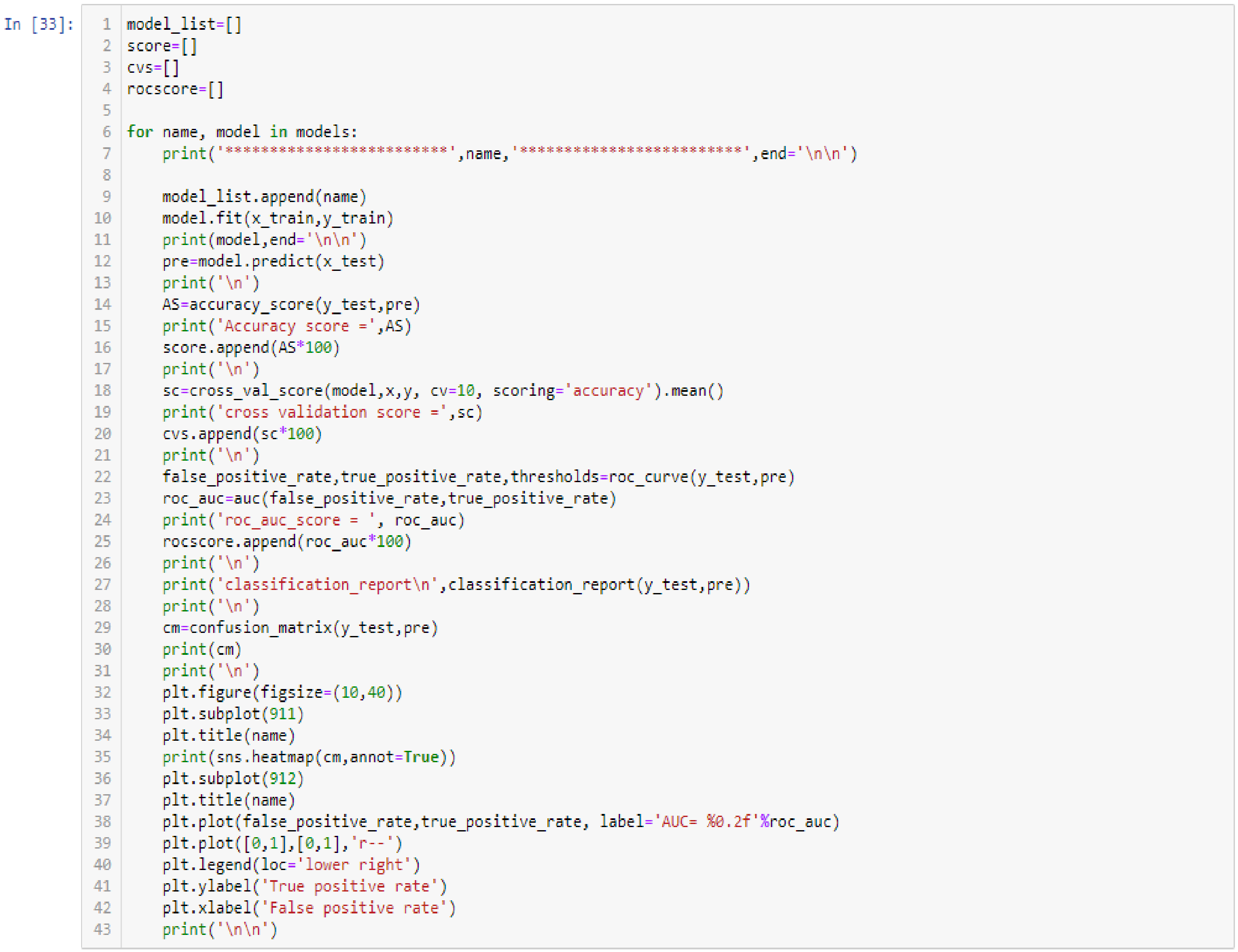
Then, by using various classifiers, models were built. Different metrics were used to evaluate the best performing model. Finally using the best model, predicted on the testing data and saved the results.

- Testing of Identified Approaches :

The algorithms used for the training and testing were :

LogisticRegression, KNeighbourClassifier, Support Vector Classifier, Decision Tree Classifier,Random Forest Classifier, Multinomial NB

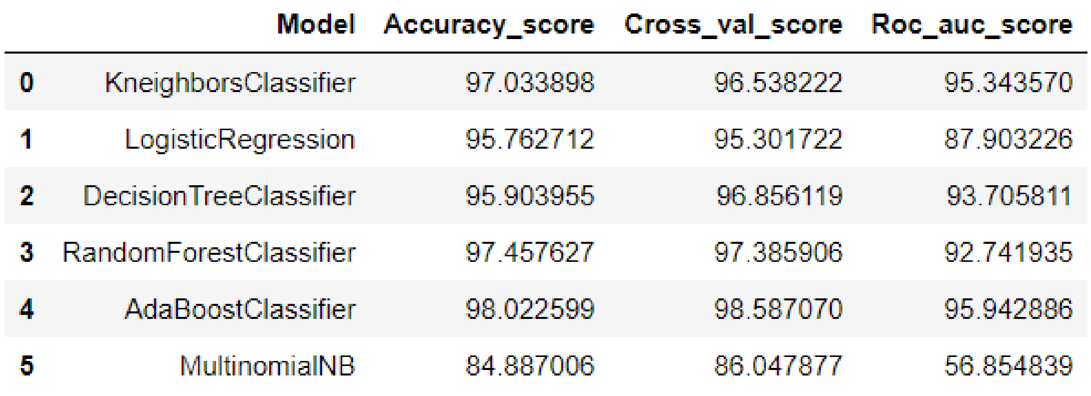
● Run and Evaluate selected models.



```
In [33]:
model_list=[]
score=[]
cvs=[]
rocscore=[]

for name, model in models:
    print('*************************',name,'*************************',end='\n\n')

    model_list.append(name)
    model.fit(x_train,y_train)
    print(model,end='\n\n')
    pre=model.predict(x_test)
    print('\n')
    AS=accuracy_score(y_test,pre)
    print('Accuracy score =',AS)
    score.append(AS*100)
    print('\n')
    sc=cross_val_score(model,x,y, cv=10, scoring='accuracy').mean()
    print('cross validation score =',sc)
    cvs.append(sc*100)
    print('\n')
    false_positive_rate,true_positive_rate,thresholds=roc_curve(y_test,pre)
    roc_auc=auc(false_positive_rate,true_positive_rate)
    print('roc_auc_score = ', roc_auc)
    rocscore.append(roc_auc*100)
    print('\n')
    print('classification_report\n',classification_report(y_test,pre))
    print('\n')
    cm=confusion_matrix(y_test,pre)
    print(cm)
    print('\n')
    plt.figure(figsize=(10,40))
    plt.subplot(911)
    plt.title(name)
    print(sns.heatmap(cm,annot=True))
    plt.subplot(912)
    plt.title(name)
    plt.plot(false_positive_rate,true_positive_rate, label='AUC= %0.2f'%roc_auc)
    plt.plot([0,1],[0,1],'r--')
    plt.legend(loc='lower right')
    plt.ylabel('True positive rate')
    plt.xlabel('False positive rate')
    print('\n\n')
```



	Model	Accuracy_score	Cross_val_score	Roc_auc_score
0	KneighborsClassifier	97.033898	96.538222	95.343570
1	LogisticRegression	95.762712	95.301722	87.903226
2	DecisionTreeClassifier	95.903955	96.856119	93.705811
3	RandomForestClassifier	97.457627	97.385906	92.741935
4	AdaBoostClassifier	98.022599	98.587070	95.942886
5	MultinomialNB	84.887006	86.047877	56.854839

We can observe that AdaBoost provides highest accuracy of about 98.02%

- Visualizations

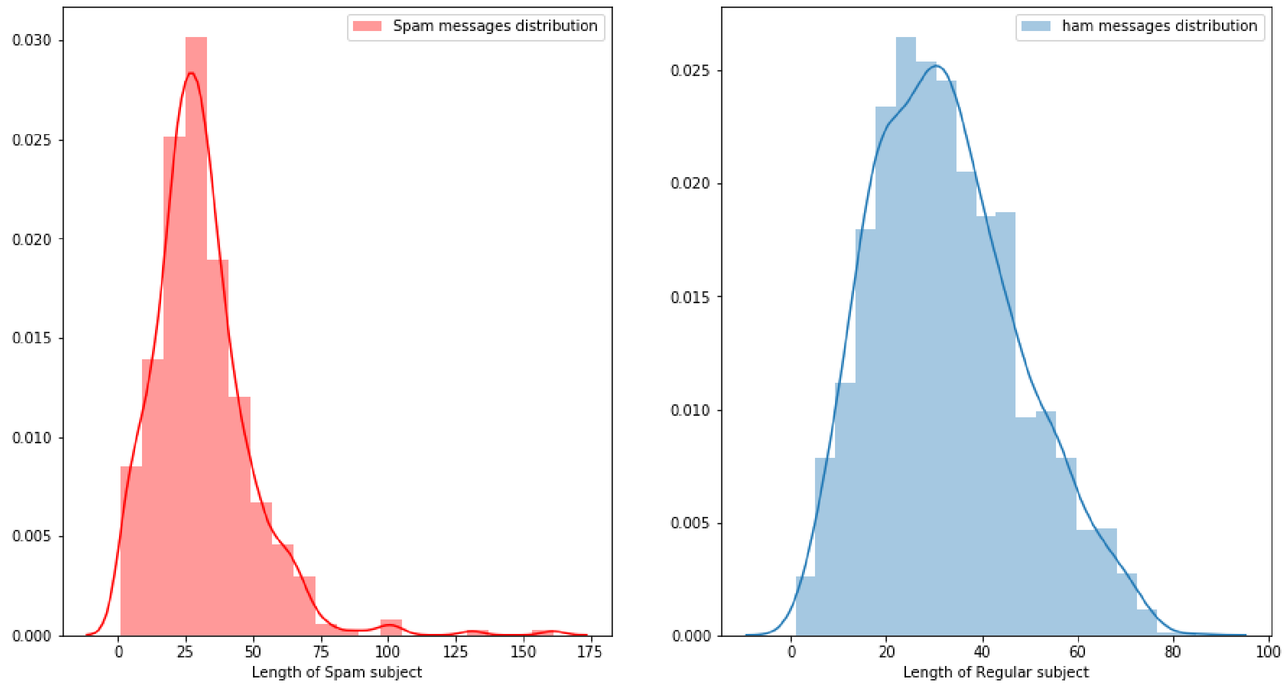


Fig 2: Word count distribution BEFORE cleaning for subject column.

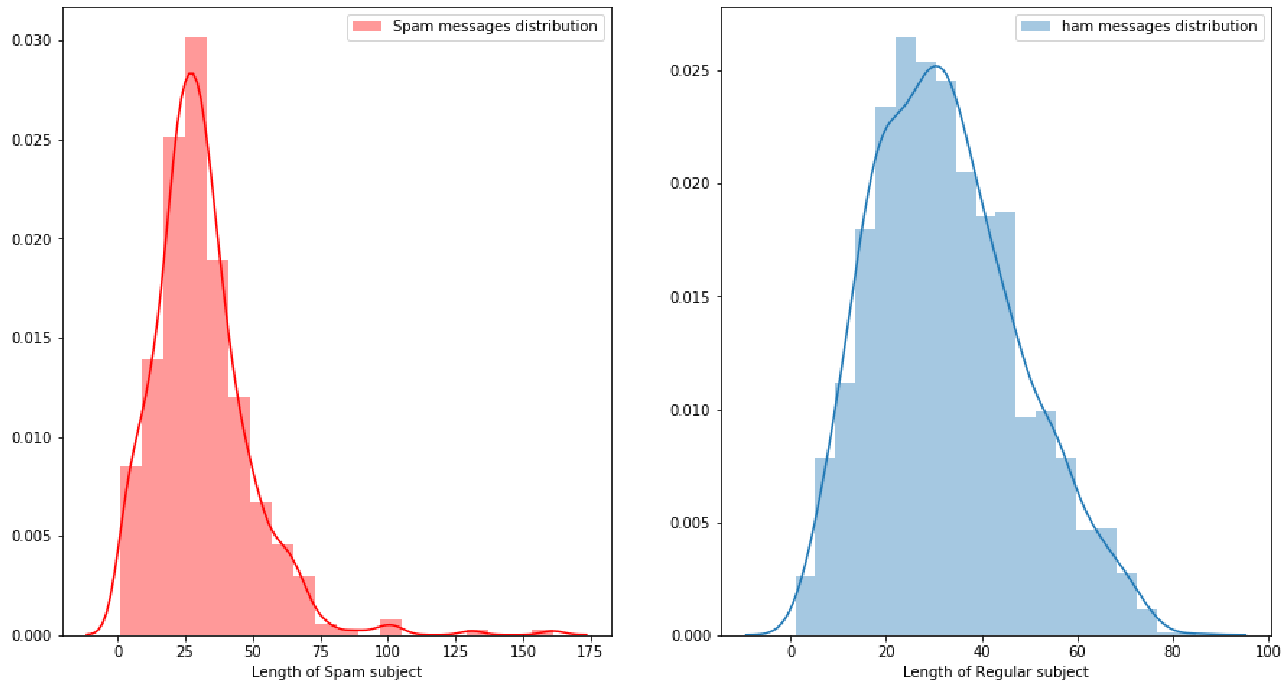


Fig 3: Word count distribution AFTER cleaning for subject column.

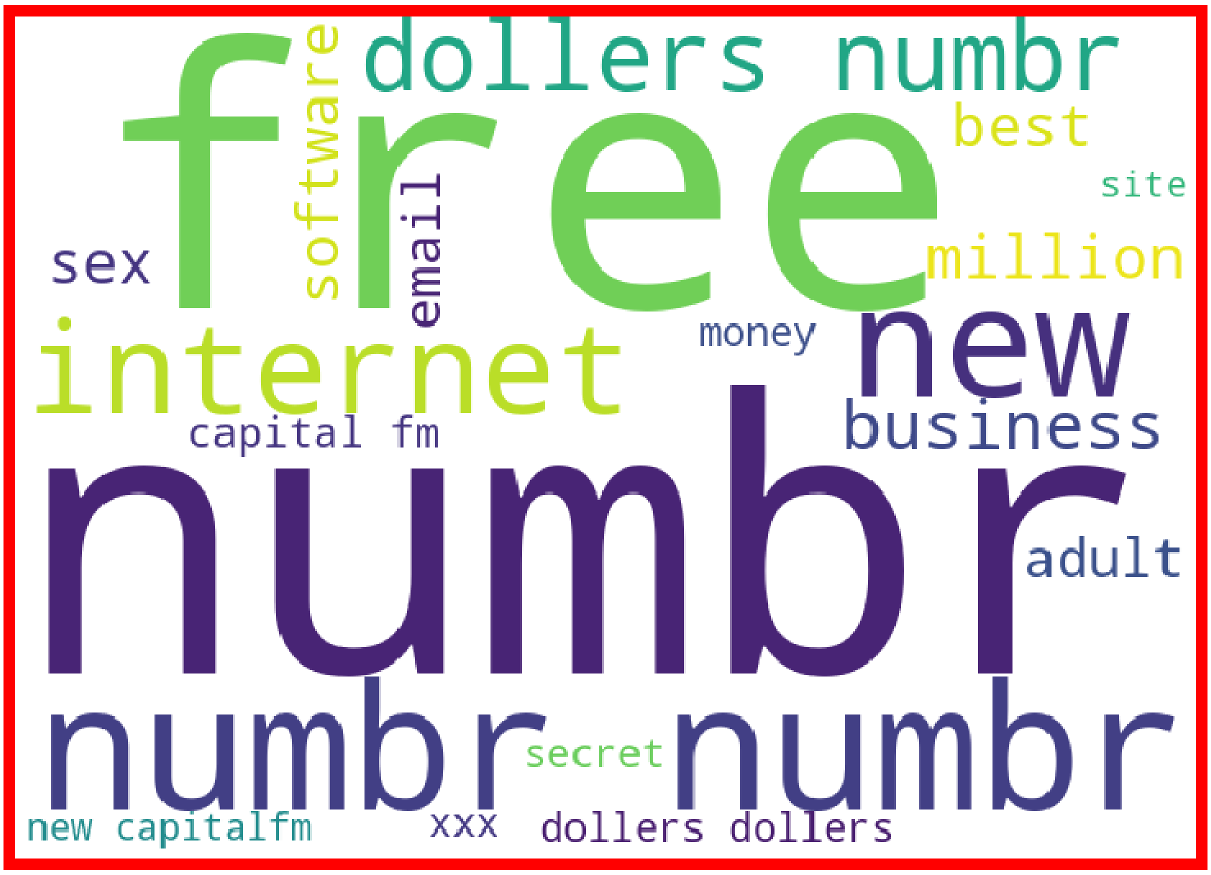


Fig 4: Displaying the loud words in spam for subject column.

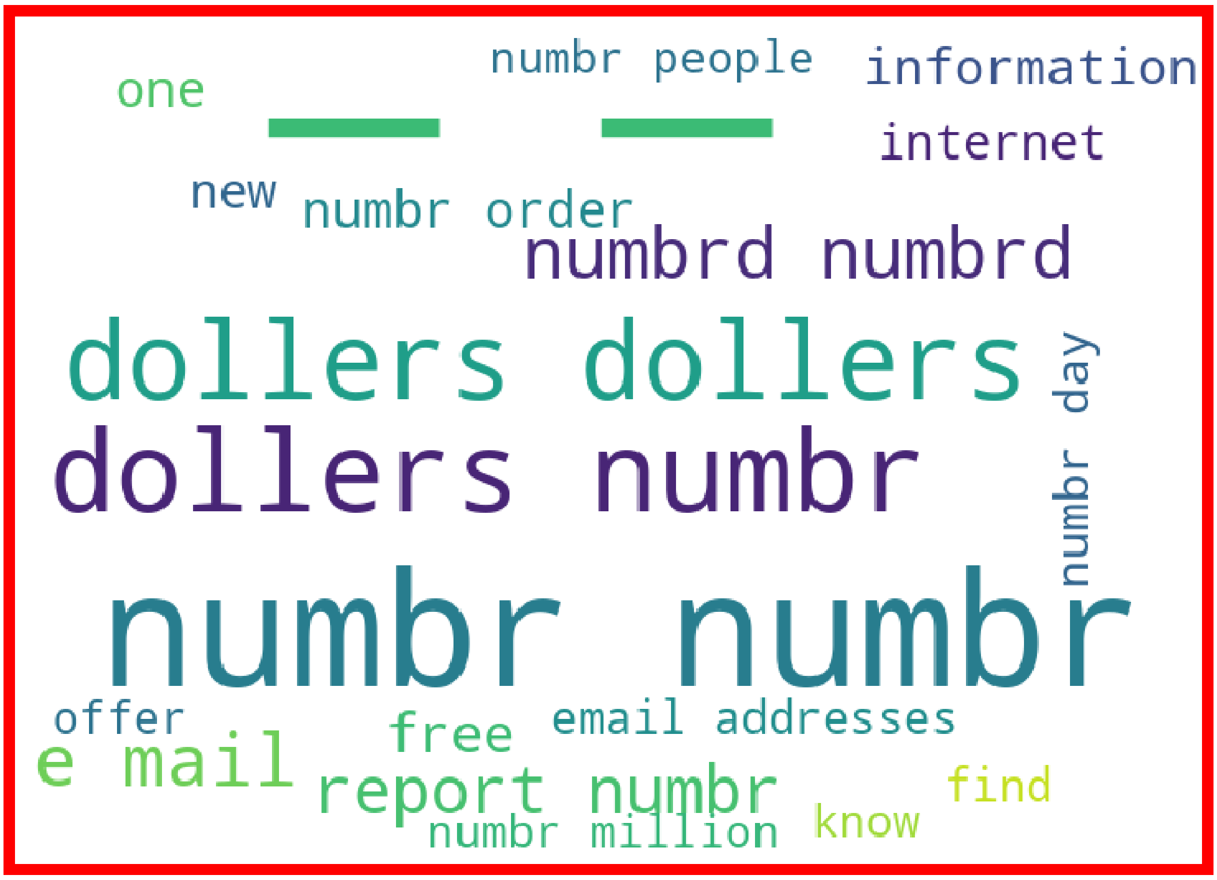


Fig 5: Displaying the loud words in spam for message column.

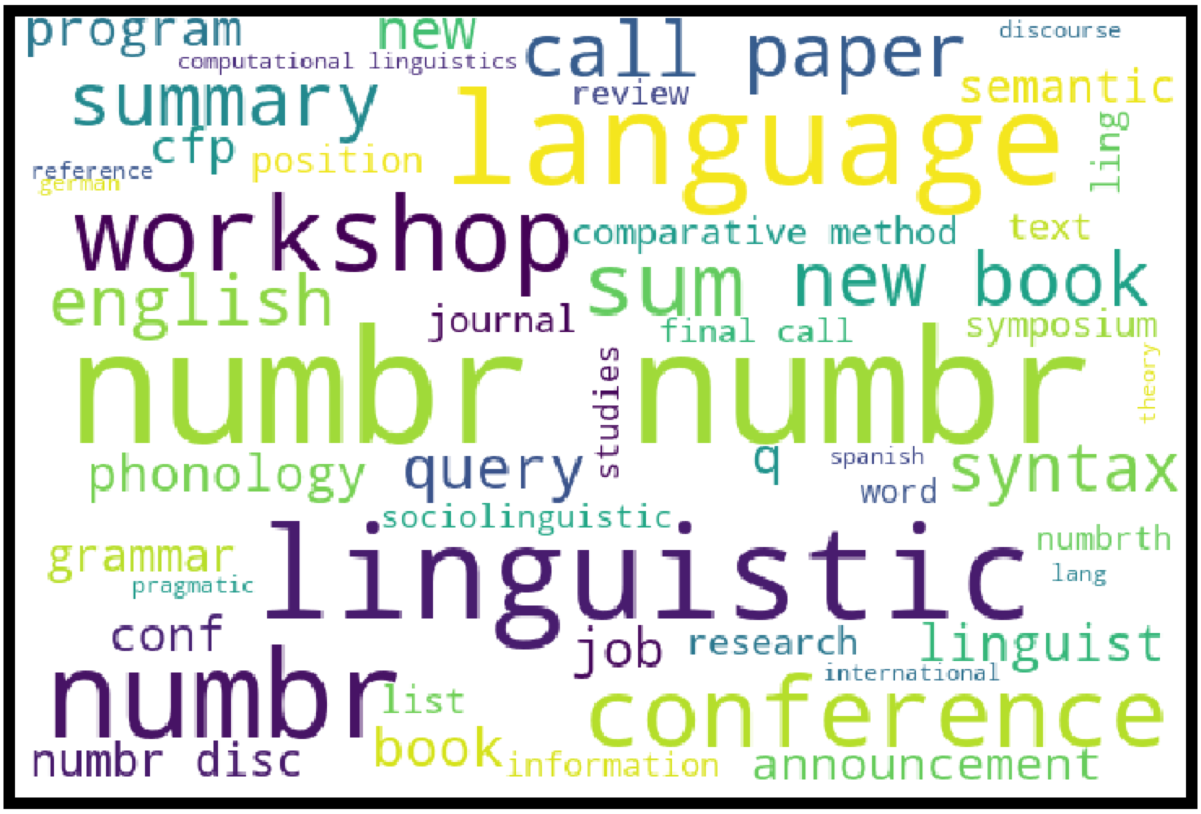


Fig 6: Displaying the loud words in regular emails for subject column.

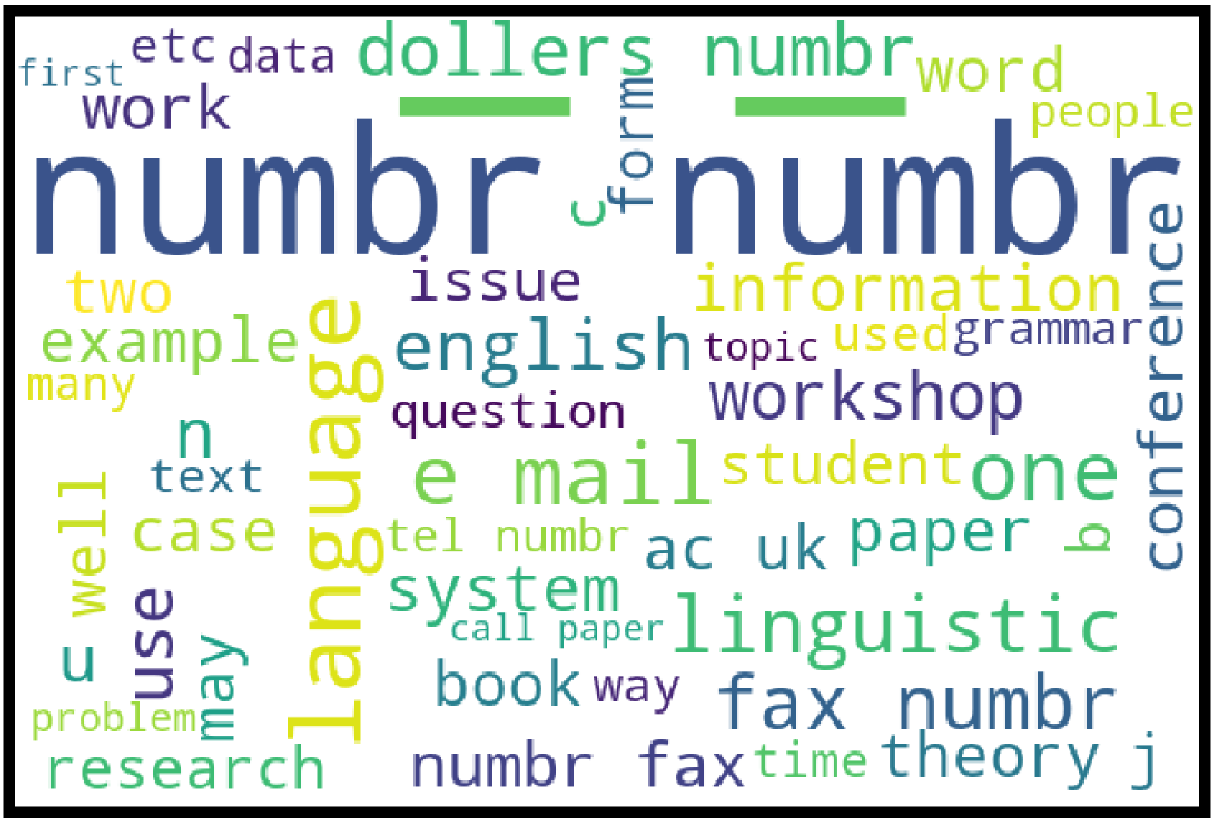


Fig 7: Displaying the loud words in regular emails for message column.

CONCLUSION

- Key Findings and Conclusions of the Study

    1. Spam emails consisted of various keywords like Free, Dollars, Internet, Sex, Number, Business and Report.
    2. Ham emails consisted of various keywords like Workshop, Linguistic, Conference, Number, Language, English and Job.