CSE6006 – NoSQL Databases

J Component - Project Report

Project Title – Spam Mail Classifier

By
20MCB1005 - Shweta Shewale
20MCB1014 - Diwakar Singh
20MCB1020 - Thanga Purni J S

MTech CSE (Big data Analytics)

Submitted to

Dr. A. Bhuvaneswari,
Assistant Professor Senior,
SCOPE, VIT, Chennai

School of Computer Science and Engineering

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

March 2021

ABSTRACT

In today's globalized world, email is a primary source of communication. This communication can vary from personal, business, corporate to government. With the rapid increase in email usage, there has also been increase in the SPAM emails. SPAM emails, also known as junk email involves nearly identical messages sent to numerous recipients by email. Apart from being annoying, spam emails can also pose a security threat to computer system. In this project, we use text mining to perform automatic spamfiltering to use emails effectively. We try to identify patterns using Data-mining classification algorithms to enable us classify the emails as HAM or SPAM.

TABLE OF CONTENTS

1. Introduction

E-mails are a major way of communicating over the internet. They are free, fast and not too informal. Ever since 1990s, people, including business professionals and students alike, have been using e-mails as a very important means of communication. E-mails, however can be used for purposes other than genuine communication. In this age, we have thousands of small businesses cropping up and people are trying to advertise their companies and products. The easiest way for such companies and groupsto reach a larger audience is through the internet. These groups start sending e-mails inthe form of promotions, deals, discounts and offers. While the intent of these e-mails isn't necessarily malicious, they are unwanted and people wouldn't want to have their mailboxes filled up with such undesired messages, which might actually lead to the lossof important data. These e-mails can be categorized as spam. While these are the non- malicious forms of spam e-mails, the more dangerous forms of spam e-mails are linkedto phishing and data theft through
these links.

1.1 Objective and goal of the project

The objective of Spam Email classification project is to give knowledge to theuser about the fake e-mails and relevant emails. And the goal of this project is to classify mail SPAM or HAM.

1.2 Problem Statement

To implement a system for classifying the upcoming email is SPAM or Hamusing machine learning techniques and MongoDB as database.

1.3 Motivation

We have chosen this project because now days there are lot of people trying to fool other people just by sending fake email like lottery winning and hack their account balance. To prevent this type of robbery we are going to implement this project.

1.4 Challenges

- It reduces network resource costs.
- It reduces IT administration cost.
- It reduces Legal liability cost.
- It increases security and control.

2. Literature Survey

1) **Analysis of Machine Learning Methods for Filtering Spam Messages in Email**
   In this paper describes the advantages and disadvantages of methods for detecting and protecting against spam messages in electronic mail services, considers the classification of spam messages and the Naive Bayes spam filter used to classify spam messages. The characteristics and effectiveness of spam filters based on machinelearning methods are analyzed.
   The effectiveness of a Bayesian spam filter can be increased with pre-processing steps that are applied to the spam keywords training. Machine learning algorithms play a central role in detection of spam e-mail. In this paper, we presented an empirical evaluation of three machine learning algorithms for spam filtering. These approaches, NB, k-NN, and SVM, were applied to different parts of an e-mail in order to compare their performance.

2) **Identifying Spam Patterns in SMS using Genetic Programming Approach**
   The model proposed in this paper generates regular expressions as individuals of population, using Genetic Programming Approach. The domain of SMS spam filteringhas not been explored widely. It is able to eliminate False Positive errors, thus saving legitimate messages from being misclassified. The performance tends to improve with higher number of generations. Performance and confusion matrix for different number of generations are tabulated.
   Preventing loss of crucial and urgent information. This is further utilized for SMS spamdetection with modifications in parameters including computing maximum population,specifying number of generations. The results achieved by the regular expressions generated by the proposed model for different number of generations are tabulated and their performances are compared. Confusion matrix is also drawn for different generations. As each generation produces better set of individuals in population performance improves with greater number of generations, yielding better accuracy andsensitivity.

3) **Performance Analysis of E-Mail Spam Classification using different Machine Learning Techniques**
   This paper elucidates the different Machine Learning Techniques such as J48 classifier,Adaboost, K-Nearest Neighbor, Naive Bayes, Artificial Neural Network, Support Vector Machine, and Random Forests algorithm for filtering spam emails using differentemail dataset. The overview of several Spam Filtering techniques and summarizing theaccuracy of different proposed approach regarding several parameters.
   Though all are effective but still now spam filtering system have some lacking which are the major concern for researchers and they are trying to generate next generation spam filtering process which have the ability to consider large number of multimedia data and filter the spam email more prominently. Random Forests provides better accuracy when compared to other Machine Learning techniques.

4) **Comparative Approach to Naïve Bayes Classifier and Support Vector Machine for Email Spam Classification**

The ultimate goal of this paper is to distinguish spam emails from legitimate emails. It was conducted depending on the content or body of the emails by applying Naïve BayesClassifier and Support Vector Machine. Furthermore, the accuracy of Naïve Bayes Classifier is compared with the accuracy of Support Vector Machine. From the results of classification analysis, Support Vector Machine has shown in its accuracy that is higher than Naïve Bayes Classifier in different sizes of training emails. In the battle of fighting spam emails, by applying Support Vector Machine, this system can be used tofilter the on-line email not to enter into user's inbox.

5) **Email Spam Detection using Bidirectional Long Short-Term Memory with Convolutional Neural Network**

In this paper, we propose a new model for detecting spam messages based on the sentiment analysis of the textual data of the email body. We incorporate Word-Embeddings and Bidirectional LSTM network to analyze the sentimental and sequentialproperties of texts. Furthermore, we speed up the training time and extract higher leveltext features for Bi-LSTM network using Convolution Neural Network. We involve twodatasets namely ling spam dataset and spam text message classification dataset and adoptrecall, precision and f-score for comparing and evaluating the performance of our proposed approach. For improving the performance of accuracy Apart from this, modeloutperforms not only to some popular machine learning classifiers but also to state of the art approaches for detecting spam messages and hence, proves the superiority by itself.

6) **Email Spam Detection using Naive Bayes Classifier**

They explained the classification of unwanted emails to identify spam and not spam. For this they used the Naïve Bayesian Classifier. In this project, they created an e-mailclassification system to classify spam and not spam. To do this, they created a custom dataset to run this experiment. With the help of Naïve Bayesian Classifier for which it calculates the probability of spam and non-spam mail and make a prediction whose value is greater.

7) **An Efficient Malicious Email Detection Using Multi Naive Bayes Classifier**

This algorithm consists of multiple Naïve Bayes algorithms that are responsible for thewhole classification of a data set. The comparison of other robust classifiers like Bayes Net, Naive Bayes and Naive Bayes Multinomal with Multi Naive Bayes classifier in terms of correctly and incorrectly classified instances have been done. Various measurements of performance like False Positive rate, Precision and Recall of various Classifiers have been computed.

8) **Spam image email filtering using K-NN and SVM**

SVM tend to set aside a long opportunity to prepare with an expansive information

set. On the off chance that "excess" examples are recognized and erased in pre-handling, the preparation time could be diminished fundamentally. We propose a k-nearest neighbor

(k-NN) based example determination strategy. The strategy tries to select the examples that are close to the choice limit and that are effectively named. The fundamental thought is to discover close neighbors to a question test and prepare a nearby SVM that jelly the separation work on the gathering of neighbors.

9) Email Spam Filtering Using Decision Tree Algorithm

ID3 is a non-incremental algorithm used to build a decision tree from a fixed set of observations. The resulting tree is used to classify test observations. Information gain uses entropy as a measure to calculate the amount of uncertainty in dataset. The proposed system will detect the spam emails sent and received and will give notification to the respective user. The dataset provided to the system is routinely updated so that it detects new type of spams and notifies the user

10) E-mail Spam Detection and Classification using SVM

The support vector machine has shown power in binary classification. It's Wise Theoretical Foundation and Well Perfect Learning Algorithm Rules. This leads to stable information classification. The only disadvantage is this is it's time and memory once the size of the information is large.

11) Machine Learning Methods for Spam E-Mail Classification

In this paper we review some of the most popular machine learning methods and of their applicability to the problem of spam e-mail classification. Descriptions of the algorithms are presented, and the comparison of their performance on the Spam Assassin spam corpus is presented, the experiment showing a very promising results specially in the algorithms that is not popular in the commercial e-mail filtering packages, spam recall percentage in the six methods has the less value among the precision and the accuracy values, while in term of accuracy we can find that the Naïve bayes and rough sets methods has a very satisfying performance among the other methods, more research has to be done to escalate the performance of the Naïve bayes and Artificial immune system either by hybrid system or by resolve the feature dependence issue in the naïve bayes classifier, or hybrid the Immune by rough sets. Finally, hybrid systems look to be the most efficient way to generate a successful anti-spam filter nowadays.

12) Clustering and classification of email contents

Classification algorithms are conducted to evaluate the performance of experimented cluster algorithms. True Positive TP rate is shown to be very high in all cases. However, FP rate was shown to be the best in case of NGram based clustering and classification. Such accuracy can also depend on the number of folders in the classification scheme. The major challenge we noticed in the analysis process is the large number of emails and the large number of unique terms that

are used as inputs to the clustering and classification processes. It is desirable in future that email servers or applications should include different types of pre-defined folders. The first category includes the general traditional folders: Mailbox, sent, trash, etc. It should also allow users to add new folders that can be user defined as well as intelligent or context aware. In convergence with social networks, users should be able to classify emails based on senders or content into different groups.

13) Random Forests Machine Learning Technique for Email Spam Filtering

In this study, we proposed Random Forests algorithm for effective and efficient email spam filtering. And evaluated the performance of RFs algorithm on Enron spam datasets using accuracy, TPR, FPR, precision and F-measure to determine the effectiveness and efficiency of the algorithm. We conclude by stating that RFs is a promising algorithm that can be adopted either at mail server or at mail client side to further decrease the volume of spam messages in email user's inbox.

14) Spam And Email Detection In Big Data Platform Using Naives Bayesian Classifier

To avoid spam/irrelevant mails we'd like effective spam filtering strategies. Spam mails area unit used for spreading virus or malicious code, for fraud in banking, for phishing, and for advertising. Spam messages are nuisance and huge problem to most users since they clutter their mailboxes and waste their time to delete all the junk mails before reading the legitimate ones. They also cost user money with dial up connections; waste network bandwidth and disk space. Bayesian classifier is one of the most important and widely used classifier and also it's the simplest classification method due to its manipulating capabilities of tokens and associated probabilities according to the users‟ classification decision and empirical performance. In this project, they implemented the system to analyze each and every mail. And also provide privacy-based detection system to encode the emails using Digest based system. Enhance the anomaly detector, to predict emails with pop up window with email tracking system. In the future work we have a plan to implement other algorithm to our classification method to achieve better performance.

15) An Anti-Spam System Using Artificial Neural Networks and Genetic Algorithms

An anti-spam filtering system was proposed which uses the multi-layer artificial neural network trained by the genetic algorithm. The results clearly show that the Subject and Body fields can contain enough information to classify e-mails into spam or legitimate. The results have also shown that the MLP with 15-30 neurons in the first hidden layer are sufficient to filter both easy spam and easy legitimate e-mails. The MLP architecture used to develop our system is good for filtering e-mails, if we do not take into account the long time needed to train the MLP. We have also investigated the effects of several GA parameters. The parameters that

have been found to be the most significant to the performance of the classifier are: size of the population pool, crossover, mutation probabilities and mutation method. It is important to remember that e-mail filtering is highly sensitive application of textual classification problem. The classifier must be able to handle many input features, with low false positive and low false negative.

3. Requirements Specification

3.1 Hardware Requirements

- Hard Disk – 500 GB or Above
- RAM required – 4 GB or Above
- Processor – Core i3 or Above

3.2 Software Requirements

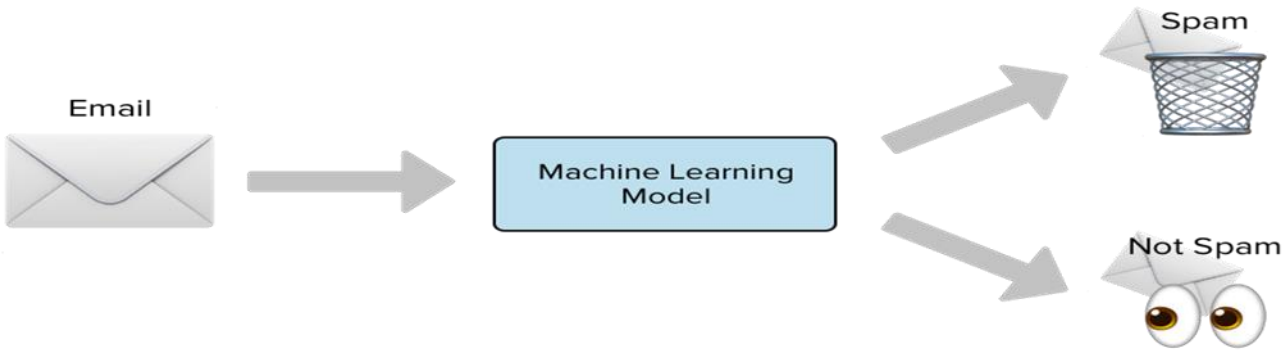
- Windows OS/ Linux
- MongoDB
- Jupyter Notebook/ PyCharm

3.3 Functional Requirement

- PROGRAMMING LANGUAGE : Python, JSON.
- Libraries: Numpy, Pandas, Seaborn, Matplotlib, NLTK
- DATABASE: MongoDB
- PLATFORM : Jupyter Notebook

4. Dataset details

- Dataset consist of Rows 5574 and 3 column
- Labels: it tells weather email is SPAM or HAM
- SMS: it shows the text sentence of data



5. Implementation

5.1 Data Pre-processing:

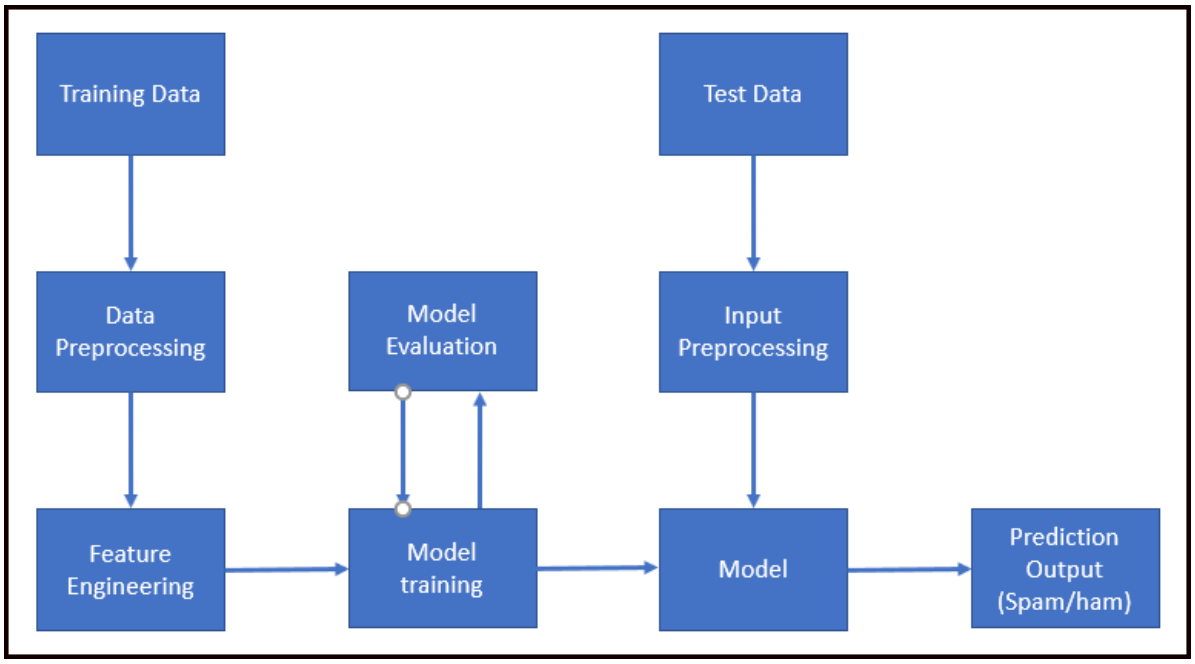
There are three steps involved in pre-processing viz. tokenization, stop word removal and Lemmatization.

I. **Tokenization** is not only breaking the text into components, pieces like words, punctuation etc. known as tokens. However it is more than that. Tokenizer which internally identify whether a "." is a punctuation and separate it into token or it is part of abbreviation like "U.S." and do not separate it.

II. **Stop words** are the words which carry nearly no information when considered form the text mining point of view. These words contain pronouns, prepositions and conjunctions like he, she, they, and, if, but, etc.

III. **Lemmatization:** Lemmatization is a more effective option than stemming because it converts the word into its root word, rather than just stripping the suffices.

Architecture:



5.2 Feature Selection:

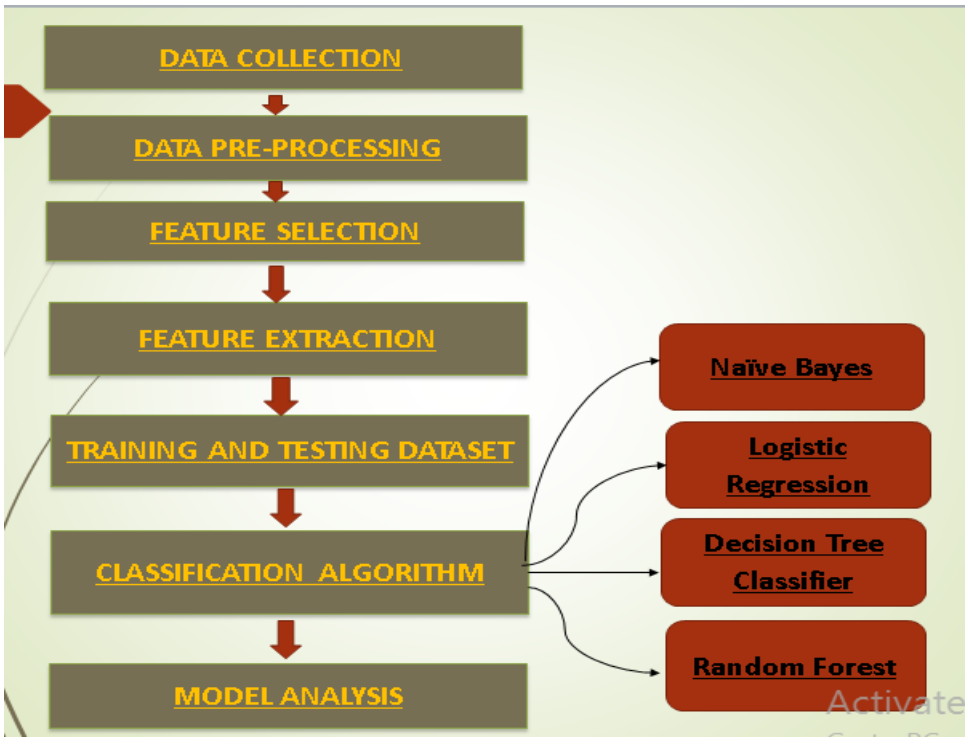
In this process we analyze the text and minutely to find out the features(i.e. words) which would be most useful in the classification. Then these features would be further used to train the classifier.

TF-IDF stands for “Term Frequency – Inverse Document Frequency”.
TF-IDF is a numerical statistic which measures the importance of the word in a document.

Term Frequency: Number of time a word appears in a text document.
Inverse Document Frequency: Downscales the words that appear a lot across the documents.
Notice that the words which have the lowest IDF values. This is expected as these words appear in each and every document in our collection. The lower the IDF value of a word, the less unique it is to any particular document

5.3 Feature Extraction:

Feature extraction technique is used to extract important and relevant features from the email body. Feature replaces email 2D vector space features numbers. These features are mapped from the dictionary list.



5.4 Connection Between MONGODB to Jupyter

Here we are connecting MONGODB database to Jupyter Notebook through python

```
client = pymongo.MongoClient("mongodb://localhost:27017/")

# Database Name
db = client["project"]

# Collection Name
col = db["spam_ham"]

spam_data= col.find()
_id=[]
Label=[]
SMS=[]
for data in spam_data:
    _id.append(data["_id"])
    Label.append(data["Label"])
    SMS.append(data["SMS"])
```

Dataset insight:

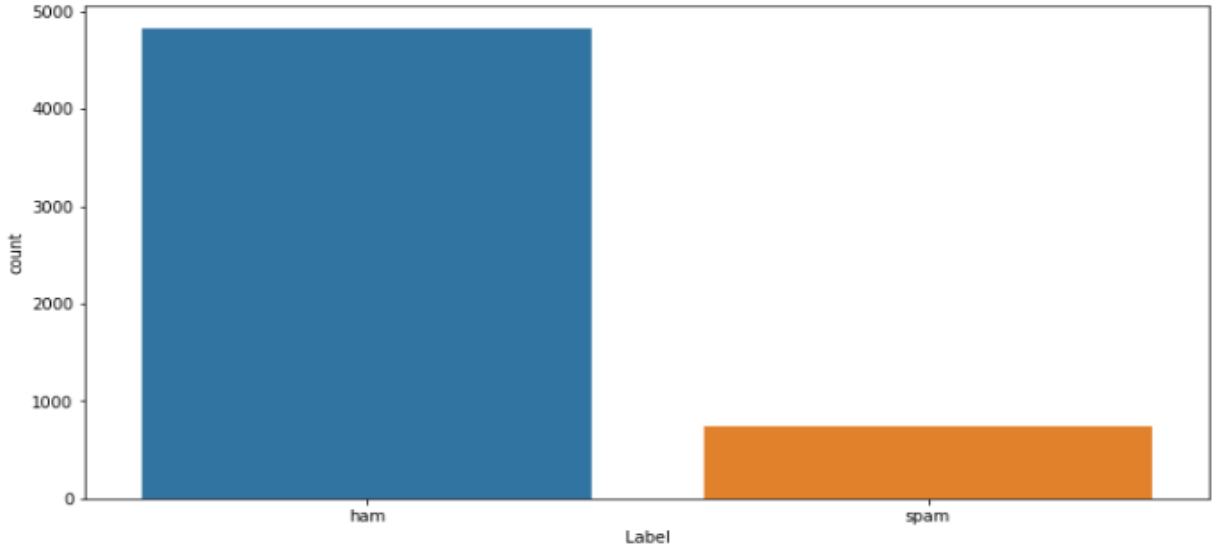
```
print(spam_data)
```

```
     Label                                                SMS
0      ham  Go until jurong point, crazy.. Available only ...
1      ham                      Ok lar... Joking wif u oni...
2     spam  Free entry in 2 a wkly comp to win FA Cup fina...
3      ham  U dun say so early hor... U c already then say...
4      ham  Nah I don't think he goes to usf, he lives aro...
5     spam  FreeMsg Hey there darling it's been 3 week's n...
6      ham  Even my brother is not like to speak with me. ...
7      ham  As per your request 'Melle Melle (Oru Minnamin...
8     spam  WINNER!! As a valued network customer you have...
9     spam  Had your mobile 11 months or more? U R entitle...
10     ham  I'm gonna be home soon and i don't want to tal...
11    spam  SIX chances to win CASH! From 100 to 20,000 po...
12    spam  URGENT! You have won a 1 week FREE membership ...
13     ham  I've been searching for the right words to tha...
14     ham                I HAVE A DATE ON SUNDAY WITH WILL!!
15    spam  XXXMobileMovieClub: To use your credit, click ...
16     ham                         Oh k...i'm watching here:)
17     ham  Eh u remember how 2 spell his name... Yes i di...
18     ham  Fine if that's the way u feel. That's the way ...
19    spam  England v Macedonia - dont miss the goals/team...
20     ham          Is that seriously how you spell his name?
```

	Label	SMS	word_count	char_count	avg_word	stopwords	stopwords_count	hastags	address
0	ham	go jurong point crazy available bugis n great ...	20	111	4.600000	[until, only, in, there]	4	0	0
1	ham	ok lar joking wif u oni	6	29	4.000000	[]	0	0	0
2	spam	free entry 2 wkly comp win fa cup final tkts 2...	28	155	4.571429	[in, a, to, to, to]	5	0	0
3	ham	u dun say early hor u c already say	11	49	3.545455	[so, then]	2	0	0
4	ham	nah i dont think goes usf lives around though	13	61	3.769231	[don't, he, to, he, here]	5	0	0
5	spam	freemsg hey darling 3 weeks word back id like ...	32	147	3.625000	[there, it's, been, now, and, no, some, you, u...	13	0	0
6	ham	even brother like speak they treat like aids p...	16	77	3.875000	[my, is, not, to, with, me]	6	0	0
7	ham	as per request melle melle oru minnaminunginte...	26	160	5.192308	[your, has, been, as, your, for, all, to, your]	9	0	0
8	spam	winner as valued network customer selected rec...	26	157	5.076923	[a, you, have, been, to]	5	0	0
9	spam	had mobile 11 months u r entitled update lates...	29	154	4.344828	[your, or, to, to, the, with, for, on]	8	0	0

Bar plot describes the count of the class labels

```
# Barplot describes the count of the class labels
plt.figure(figsize = (12, 6))
sns.countplot(data = spam_data, x = 'Label');
```



Imported Packages:

```
from sklearn.feature_extraction.text import CountVectorizer, TfidfTransformer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import classification_report, f1_score, accuracy_score, confusion_matrix
from sklearn.pipeline import Pipeline
from sklearn.tree import DecisionTreeClassifier
```

Splitting the dataset into 70% for training and 30% for testing set.

We have data imbalance, so to balance the dataset we are using Shuffling. Shuffling the dataset to distribute the data equally as we have ham data much more than spam data.

Imbalance dataset can overshadow the useful information about the data which could be necessary for building rule-based classifiers such as Random Forests and make the model biased as well not suitable for future data.

```
[ ]  1  #shuffle dataframes for ham/spam mails as no of ham mail are much higher than spam mail
     2  from sklearn.utils import shuffle
     3  spam_data = shuffle(spam_data)
     4  len(spam_data)
```

```
5574
```

After splitting the dataset is divided into following no:

```
1  df_train.groupby(['Label']).count()
```

Label	SMS
ham	3376
spam	525

```
1  df_test.groupby(['Label']).count()
```

Label	SMS
ham	1451
spam	222

For doing Data Preprocessing few operations we have performed such as:

Lemmatization: Lemmatization is a more effective option than stemming because it converts the word into its root word, rather than just stripping the suffices. It makes use of the vocabulary and does a morphological analysis to obtain the root word. Therefore, we usually prefer using lemmatization over stemming.

```
from textblob import Word
spam_data.loc[:, ('SMS')] = spam_data.loc[:, ('SMS')].apply(lambda x: " ".join([Word(word).lemmatize() for word in x.split()]))
spam_data['SMS'].head(10)
```

For creating the dictionary of the dataset I have used bag of word because it our dataset is small and context is domain specific, BoW may work better than Word Embedding here and the simple ways to build BoW mode are Count Vectorizer() and lemmatization. Here we are using lemmatization as it cut down the words into root word or base word.
TF-IDF (term frequency–inverse document frequency) to balance the less frequent word with most frequent word document.

```
#for each word in the email text, get the base form of the word and return the list of base words
def split_into_lemmas(SMS):
    message = SMS.lower()
    words = TextBlob(message).words
    # for each word, take its "base form" = lemma
    return [word.lemma for word in words]
```

```
#function to apply the count vectorizer(BOW) and TF-IDF transforms to a set of input features
def features_transform(mail):
    #get the bag of words for the mail text
    bow_transformer = CountVectorizer(analyzer=split_into_lemmas).fit(X_train)
    print(len(bow_transformer.vocabulary_))
    messages_bow = bow_transformer.transform(mail)
    #apply the TF-IDF transform to the output of BOW
    tfidf_transformer = TfidfTransformer().fit(messages_bow)
    messages_tfidf = tfidf_transformer.transform(messages_bow)
    print(messages_tfidf.shape)
    #return result of transforms
    return messages_tfidf
```

5.5 Algorithms

I. Naïve Bayes classifier

Naive Bayes is based on Bayes' Theorem Formula with an assumption of independence among predictors. Given a Hypothesis A and evidence B, Bayes' Theorem calculator states that the relationship between the probability of Hypothesis before getting the evidence P(A) and the probability of the hypothesis after getting the evidence P(A|B) is

Naive Bayes work on **dependent events** and the probability of an event occurring in the future that can be detected from the previous occurring of the same event .

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}$$

```
#create and fit NB model
modelNB=MultinomialNB()
```

```
modelNB.fit(train_features,y_train)
```

```
MultinomialNB(alpha=1.0, class_prior=None, fit_prior=True)
```

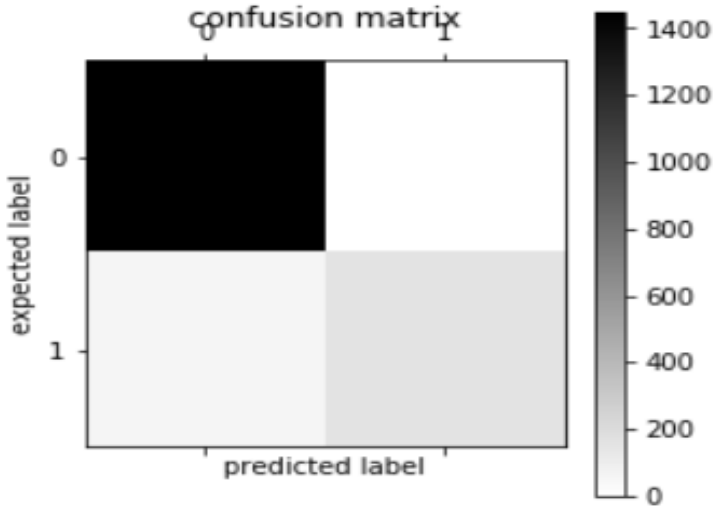
```
#transform test features to test the model performance
test_features=features_transform(X_test)
```

```
6477
(1673, 6477)
```

```
#NB predictions
predicted_class_NB=modelNB.predict(test_features)
```

```
model_assessment(y_test,Naive_Bayse)
```

```
confusion matrix
[[1451    0]
 [  60  162]]
accuracy
0.9641362821279139
precision
1.0
recall
0.7297297297297297
f-Score
0.8437499999999999
AUC
0.8648648648648649
```



II. Logistic Regression

Logistic Regression measures the relationship between the categorical dependent variable and one or more independent variables by estimating probabilities using a logistic function. In this algorithm, the probabilities detailing the outcome of our field of interest are modeled using a logistic function which is the basic equation in logistic regression. The outcome of logistic regression is a simple binary result '1' or '0' signifying if an email is a spam or not.

```
# document term vector (dtv)
dtv = cVect.transform(X_train)
dtv = dtv.toarray()
print(f"Number of Observations: {dtv.shape[0]}\nTokens/Features: {dtv.shape[1]}")
```

```
Number of Observations: 3901
Tokens/Features: 6470
```

```
%%time
lr = LogisticRegression(verbose=1)
lr = LogisticRegression(solver='liblinear', penalty ='l2' , C = 1.0)
lr.fit(dtv, y_train)
```

```
Wall time: 677 ms
```

Evaluate on the Test data

```
# Preprocess the test data
test_dtv = cVect.transform(X_test)
test_dtv = test_dtv.toarray()
print(f"Number of Observations: {test_dtv.shape[0]}\nTokens/Features: {test_dtv.shape[1]}")
```

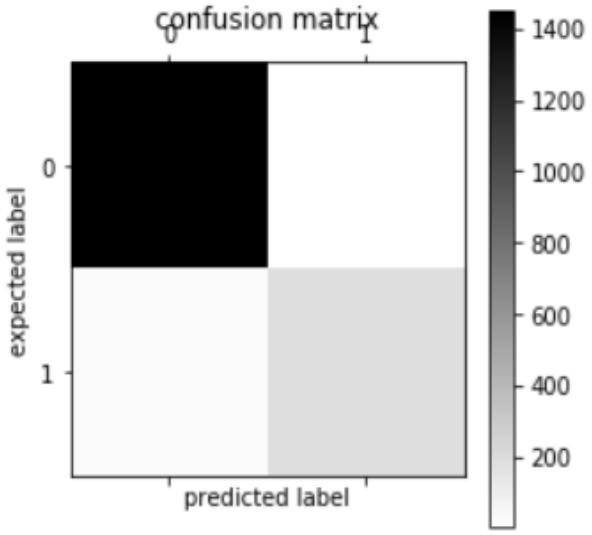
```
Number of Observations: 1673
Tokens/Features: 6470
```

```
%%time
pred = lr.predict(test_dtv)
```

```
Wall time: 133 ms
```

```
model_assessment(y_test,Logistic_Regression)
```

```
confusion matrix
[[1450    1]
 [  31  191]]
accuracy
0.9808726838015541
precision
0.9947916666666666
recall
0.8603603603603603
f-Score
0.9227053140096617
AUC
0.9298355902422063
```



III. Decision tree classifier

Algorithm for constructing decision tree usually works top-down, by choosing a variable at each step that best splits the set of items and its basically output as binary.

```
[77]  1  #create and fit tree model
      2  model_tree=DecisionTreeClassifier()
```

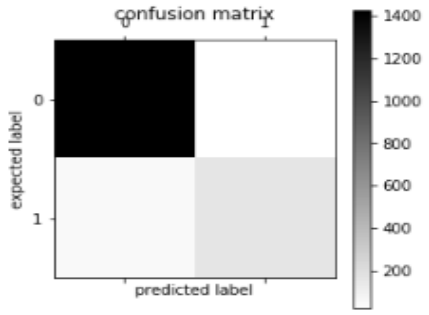
```
[78]  1  model_tree.fit(train_features,y_train)
```

```
DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='gini',
                       max_depth=None, max_features=None, max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, presort='deprecated',
                       random_state=None, splitter='best')
```

```
[79]  1  #run model on test and print metrics
      2  predicted_class_tree=model_tree.predict(test_features)
```

```
1  model_assessment(y_test,predicted_class_tree)
```

```
confusion matrix
[[1429   22]
 [  57  165]]
accuracy
0.9527794381350867
precision
0.8823529411764706
recall
0.7432432432432432
f-Score
0.8068459657701712
AUC
0.8640406429861978
```



IV. Random Forest Classifier

Decision trees are the building blocks of the random forest model. Although it is like intuitive process
Random forest work on ensemble i.e. consists of a large number of individual decision trees. Each individual tree predicts class and the class with the most votes become our model's prediction, here this has been estimated by 20 tress.

Random Forest

```
[81]  1  from sklearn.ensemble import RandomForestClassifier
```

```
      1  #create and fit model
      2  model_rf=RandomForestClassifier(n_estimators=20,criterion='entropy')
```

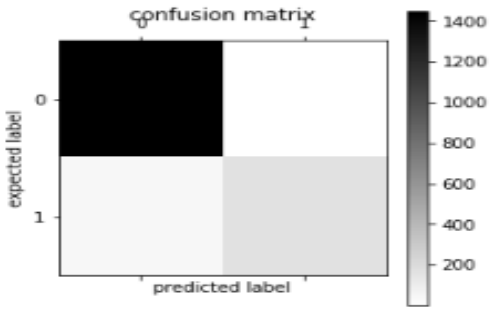
```
[83]  1  model_rf.fit(train_features,y_train)
```

```
RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                       criterion='entropy', max_depth=None, max_features='auto',
                       max_leaf_nodes=None, max_samples=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, n_estimators=20,
                       n_jobs=None, oob_score=False, random_state=None,
                       verbose=0, warm_start=False)
```

```
[84]  1  #run model on test and print metrics
      2  predicted_class_rf=model_rf.predict(test_features)
```

```
[85]  1  model_assessment(y_test,predicted_class_rf)
```

```
confusion matrix
[[1450    1]
 [  48  174]]
accuracy
0.9707112970711297
precision
0.9942857142857143
recall
0.7837837837837838
f-Score
0.8765743073047859
AUC
0.8915473019539181
```



Here, using confusion matrix to understand the reliability of the model by accuracy, precision and fscore.

Precision tells us how many of the correctly predicted cases are actually turned out to be correct.
Recall tells us how many of the actual correct cases we were able to predict correctly with our model.
F-score is a harmonic mean of Precision and Recall, and so it gives a combined idea about these two metrics. It is maximum when Precision is equal to Recall.

And for all three cases i.e. Precision, Recall and Fscore Random Forest Classifier has higher value than Decision tree Classifier. Hence RF model is more reliable than Decision tree model. Though Logistic Regression has come out as best suited model for our whole project.

```
#function which takes in y test value and y predicted value and prints the associated model performance metrics
def model_assessment(y_test,predicted_class):
    print('confusion matrix')
    print(confusion_matrix(y_test,predicted_class))
    print('accuracy')
    print(accuracy_score(y_test,predicted_class))
    print('precision')
    print(precision_score(y_test,predicted_class,pos_label='spam'))
    print('recall')
    print(recall_score(y_test,predicted_class,pos_label='spam'))
    print('f-Score')
    print(f1_score(y_test,predicted_class,pos_label='spam'))
    print('AUC')
    print(roc_auc_score(np.where(y_test=='spam',1,0),np.where(predicted_class=='spam',1,0)))
    plt.matshow(confusion_matrix(y_test, predicted_class), cmap=plt.cm.binary, interpolation='nearest')
    plt.title('confusion matrix')
    plt.colorbar()
    plt.ylabel('expected label')
    plt.xlabel('predicted label')
```

5.6 Comparison of Different ML Algorithm

Models	Accuracy
Naive Bayes	0.9665271966527197
Logistic Regression	0.9898726838015541
Decision Tree	0.9527794381350867
Random Forest	0.9707112970711297

Therefore we can say all our model is working extensively well (>95%), keep in consider that we have sample proportion of data.

In this project we implemented 4 algorithms, from that we can conclude that, Logistic Regression is giving best accuracy our sample dataset for classification of SMS that is Spam or ham.
Finally we are classifying the given SMS as Ham or Spam by using Logistic Regression.

6. Result:

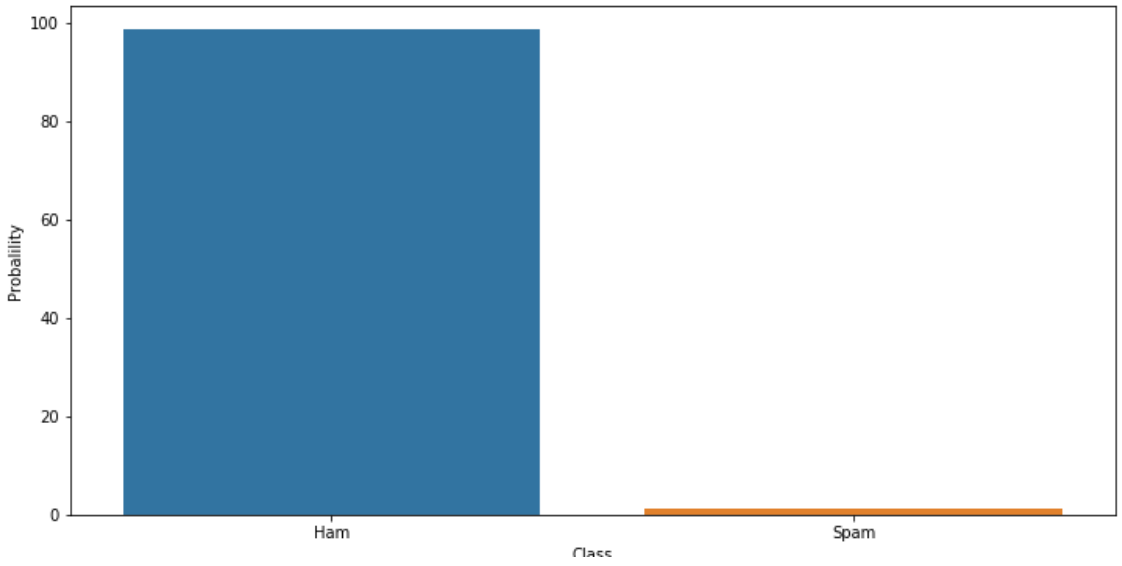
Predict Class label for the unseen data i.e., Spam or Ham

```
def predict_class(lr):
    text = input('Enter Text(Subject of the mail): ')
    text = [' '.join([ word for word in word_tokenize(text)  if not word in stop_words])]
    t_dtv = cVect.transform(text).toarray()
    prob = lr.predict_proba(t_dtv)*100
    print(f"Ham: {prob[0][0]}%\nSpam: {prob[0][1]}%")
    plt.figure(figsize=(12, 6))
    sns.barplot(x =['Ham', 'Spam'] , y = [prob[0][0], prob[0][1]])
    plt.xlabel('Class')
    plt.ylabel('Probalility')
    plt.show()
```

Here we are predicting the SMS/MAIL from the dataset and classifying that weather it's is HAM or SPAM .

```
predict_class(lr)
```

```
Enter Text(Subject of the mail): even brother like speak they treat like aid patent
Ham: 98.72743725046357%
Spam: 1.2725627495364322%
```



7. Contribution

1) Shweta Contribution

Our project title is Spam mail classifier in this project my contribution is, searching for dataset, sorting of dataset and finalizing the appropriate dataset.

Data preprocessing & Data cleaning
Data preprocessing and cleaning includes word count, character count in given SMS, removing stop words, counting stop words, removing whitespaces, Lemmatization, removing numerical value etc.

Implementation of Naïve Bayes Algorithm without using library.

- I learned and used Lambda function. I implemented the Naïve Bayes algorithm without using library by computing prior probabilities necessary for classification and created vocabulary and fetch the words from vocabulary.
- I learned from this project is, for the same dataset all classification algorithm gives the different accuracy, precision and recall value as we used No free launch theorem.
- Also, I learned how to make use of lambda function and how to prepare and clean the data before passing to model.

2) Purni Contribution

Mongo dB Compass to Python connection

- From the Backend MongoDB taking the dataset into jupyter notebook

Pre-processing
Done the Pre-Processing for Dataset using Count vectorizer, TF-IDF and I have learned how it works and how it differs from basic Pre-Processing Method using only the stop word , counting stop words, removing whitespaces, Lemmatization, removing numerical value etc.

- **Count vectorizer:** It is used to transform a given text into a vector on the basis of the frequency (count) of each word that occurs in the entire text. This is helpful when we have multiple such texts, and we wish to convert each word in each text into vectors
- **Term Frequency:** Number of time a word appears in a text document.
- **Inverse Document Frequency:** Downscales the words that appears a lot across the documents.

Implemented the Logistics Regression Algorithm and Prediction of the unseen data

- Finally done the Predict Class label for the unseen data i.e., Spam or Ham using Logistic Regression.

3) Diwakar Contribution

In this project I know and understood every concept and library used but my main contribution is

Implementing of two algorithm

- Decision tree classification
- Random forest Classification

Comparison between Algorithms

- Using confusion matrix and other measuring methods like precision, recall and fscore.

We have data imbalance, so to balance the dataset I used Shuffling. Shuffling the dataset to distribute the data equally as we have ham data much more than spam data.

Precision, Recall and Fscore Random Forest Classifier has higher value than Decision tree Classifier. Hence RF model is more reliable than Decision tree model. Though Logistic Regression has come out as best suited model for our whole project.

Comparison of Different ML Algorithm

Models	Accuracy
Naive Bayes	0.9665271966527197
Logistic Regression	0.9898726838015541
Decision Tree	0.9527794381350867
Random Forest	0.9707112970711297

Therefore we can say all our model is working extensively well (>95%), keep in consider that we have sample proportion of data.

In this project we implemented 4 algorithms, From that we can conclude that, Logistic Regression is giving best accuracy for classification of SMS that is Spam or ham.

8. References

1. Yasmine Khalid Zamil, Suhad A. Ali, Mohammed Abdullah Naser Department of Computer Science, College of Science for Women, University of Babylon, Iraq et al ,. **Spam image email filtering using K-NN and SVM ,** International Journal of Electrical and Computer Engineering (IJECE), Vol. 9, No. 1, February 2019.

2. Divesh Palival, Kevin Printer, Ramchandra Devre, Asst.Prof. Nikita Lemos et al ,. **Email Spam Filtering Using Decision Tree Algorithm,** International Journal of Scientific & Engineering Research ISSN 2229-5518 Volume 9, Issue 3, March-2018 .

3. Shivam Pandey, Ashish Taralekar, Ruchi Yadav, Shreyas Deshmukh and Prof. Shubhangi Suryavanshi Department of Computer Engineering G.H.Raisoni Institute of Engineering & Technology, Wagholi et al ,. **E-mail Spam Detection and Classification using SVM,**(IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 11 (1) , 2020.

4. Megha Tope ME Student, Computer Science and Engineering, CSMSS College of Engineering, Aurangabad, India et al ,. **Email Spam Detection using Naive Bayes Classifier,** IJSDR |Volume 4, Issue 6 June 2019 .

5. Mansi Goyal, Ankita Sharma CSE& Kurukshetra University Haryana, India et al ,. **An Efficient Malicious Email Detection Using Multi Naive Bayes Classifier,** International Journal of Advanced Research in Computer Science and Software Engineering, Volume 5, Issue 5, May 2015.

6. Ganiev Salim Karimovich Information Security Provision, professor Tashkent Universityof Information Technologies named after Muhammad al-Khwarizmi Tashkent, Uzbekistan ,Khamidov Sherzod Jaloldin ugli Cryptology department, assistant Tashkent University of Information Technologies named after Muhammad alKhwarizmi Tashkent, Uzbekistan Olimov Iskandar Salimbayevich,Cryptology department, assistant Tashkent University of Information Technologies named after Muhammad al-Khwarizmi , Tashkent,Uzbekistan et al ., **Analysis Of Machine Learning Methods For Filtering Spam Messages In Email Services**, 2020 International Conference on Information Science and Communications Technologies (ICISCT).

7. Dimple Sharma Department of Computer Science & Engineering National Institute of Technology, Raipur Aakanksha Sharaff Department of Computer Science & Engineering National Institute of Technology et al,. **Identifying Spam Patterns in SMS using Genetic Programming Approach**

8. V. Sri Vinitha Department of Information Technology Bannari Amman Institute of Technology Sathyamangalam, Erode, India srivinithavellingiri@gmail.com D. Karthika Renuka Department of Information Technology PSG College of Technology Coimbatore et al ,. **Performance Analysis of E-Mail Spam Classification using different Machine Learning Techniques**

9. Thae Ma Graduate School of Engineering University of Miyazaki, KunihitoYAMAMORI Graduate School of Engineering University of Miyazaki, Aye Thida Faculty of Computer Science University of Computer Studies, Mandalay Mandalay, et al ,. **A Comparative Approach to Naïve Bayes Classifier and Support Vector Machine for Email Spam Classification**

10. Sefat E Rahman Department of Computer Science and Engineering Khulna University of Engineering & Technology Khulna-9208, Bangladesh Email: sefat.e.rahman@gmail.com Shofi Ullah Department of Computer Science and Engineering Khulna University of Engineering et al,. **Email Spam Detection using Bidirectional Long Short-TermMemory with Convolutional Neural Network**, 2020 IEEE Region 10 Symposium (TENSYMP), 5-7 June 2020

11. W.A. Awad1 and S.M. ELseuofi2 1Math.&Comp.Sci.Dept., Science faculty, Port Said University et al ,. **Machine Learning Methods for Spam E-Mail Classification**, International Journal of Computer Science & Information Technology (IJCSIT), Vol 3, No 1, Feb 2011.

12. Izzat Alsmadi Department of Computer Science, Boise State University, Ikdam Alhami Yarmouk University, Jordan et al ,. **Clustering and classification of email contents Received** 7 January 2015.

13. E. G. Dada and S. B. Joseph Department of Computer Engineering, University of Maiduguri, Maiduguri – Borno State, Nigeria.et al ., **Random Forests Machine Learning Technique for Email Spam Filtering**, University of Maiduguri Faculty of Engineering Seminar Series Volume 9 number 1, July 2018 .

14. G.Vijayasekaran[1], S.Rosi[2] 1Asst.Professor/Department of CSE 2PG Scholar/Department of CSE, Sir Issac Newton College of Engineering and Technology, Nagapattinam, et al ,. **Spam and Email Detection in Big Data Platform Using Naives Bayesian Classifier,** June 2017.

15. Abduelbaset M. Goweder, Tarik Rashed, Ali S. Elbekaie, and Husien A. Alhammi, The High Institute of Surman for Comperhensive Professions, Surman-Libya, The HighInstitute of Zahra for Comperhensive Professions, Zahra-Libya The High Institute of Computer Technology, Tripoli-Libya, The High Institute of Zawia for Comperhensive Professions, et al ,. **An Anti-Spam System Using Artificial Neural Networks and Genetic Algorithms,** may 2014.