# SMS SPAM CLASSIFIER

*MAJOR PROJECT REPORT*

**Submitted to**
Assistant Professor – Ms. Anjli Yadav

**Submitted By:**
Aniket Kumar Singh (19CS04)

*in Partial Fulfillment for the Award of the Degree*

*of*

**B.Tech**

in

**COMPUTER SCIENCE**


**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**Lingaya's Vidyapeeth**
**(Deemed to be University Under Section 3 of UGC Act, 1956)**

**Old Faridabad-Jasana Road, Nachauli, Faridabad**

**December 2022**

# BONAFIDE CERTIFICATE

Certified that this project report **"SMS Spam Classifier"** is the bonafide work of **Aniket Kumar Singh (19CS04),** who carried out the project in collaboration with **Deptt. of Computer Science & Engineering, Lingaya's Vidyapeeth**, embodies the work done by him under the guidance of **Anjli Yadav (Assistant Professor)** towards partial fulfilment of the requirements of the Degree of Bachelor of Technology in Computer Science and Engineering from Lingaya's Vidyapeeth, Haryana.. They have fulfilled all the requirements needed as per the rules of the Vidyapeeth, for the completion of Project. This work is original and has not been submitted in part or in full to any Vidyapeeth or Institution.

Signature of the Supervisor

**Anjli Yadav(Asst. Professor)**
Deptt. of Computer Science & Engg.,
Lingaya's Vidyapeeth, Faridabad
Haryana

# ACKNOWLEDGEMENT

In completing my project, we are very thankful to many individuals and we must place on record our sincere thanks to all of them.

First of all, we would like to express our deep sense of gratitude to our supervisor **Anjli Yadav (Assistant Professor)** who gave us his invaluable guidance glowing with his words of encouragement and inspiration, criticisms and discussions throughout the problem designing.

We again very much grateful to our supervisor and department for their valuable support and cooperation in conceptualizing the project/research work and to all those outstanding individuals with whom we have worked, who helped us in understanding the concept.

We are highly thankful to our family members for their all-time support in initiating us and bringing a spark in us to pursue the work.

# INDEX

# ABSTRACT

The spam detection is a big issue in mobile message communication due to which mobile message communication is insecure. In order to tackle this problem, an accurate and precise method is needed to detect the spam in mobile message communication. We proposed the applications of the machine learning-based spam detection method for accurate detection. The SMS spam collection data set is used for testing the method. The dataset is split into two categories for training and testing the research. Automatic Text Classification is a machine learning technique. Document can be set to predefined categories based on textual content and extraction features. It has important applications in spam filtering and text mining. An analysis of SMS SPAM filtering classification model has also been done using Automatic Text Classification.

# INTRODUCTION

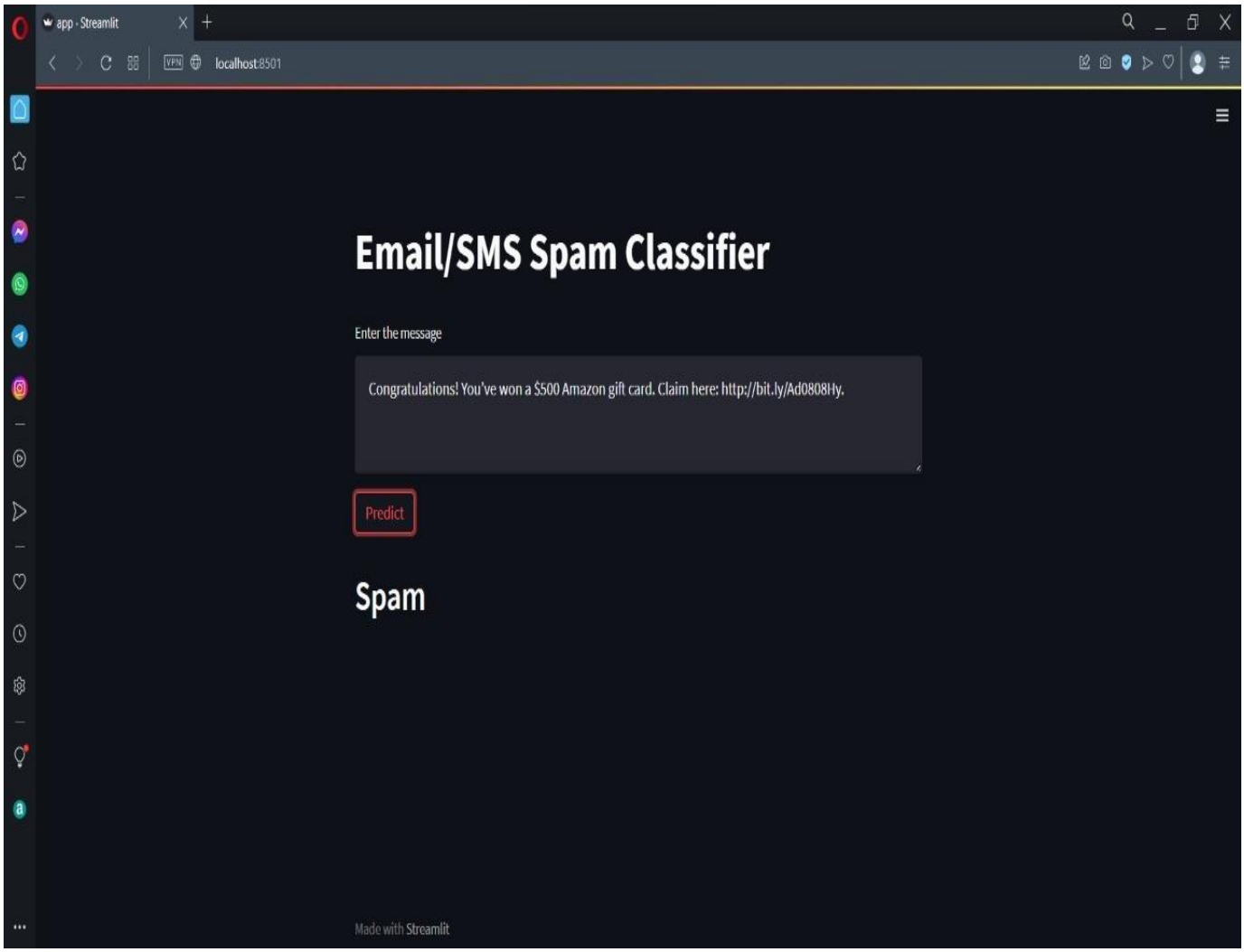
Mobile message is a way of communication among the people, and billions of mobile device users exchange numerous messages. However, such type of communication is insecure due to lack of proper message filtering mechanisms. One cause of such insecurity is spam, and it makes the mobile message communication insecure. Spam is considered to be one of the serious problems in e-mail and instance message services. Spam is a junk mail or message. Spam e-mails and messages are unwanted for receivers which are sent to the users without their prior permission. It contains different forms such as adult content, selling item or services, and so on. Due to these spam mails and messages, the values able e-mails and messages are affected because each user have limited Internet services, short time, and memory.

## Classification of SPAM/HAM SMS:

SPAM is the virus infected SMS which results malfunctioning of mobile. HAM is basics a virus free SMS. SPAM SMS can corrupt the operating system of the mobile. Mobile phone SPAM is originated from the text message and other communication services by mobile phones. Due to the extensive use of the mobile phones now a days advertisement through SMS has rapidly increased. For this reason the user cannot identify SPAM or HAM resulting the fall under the trap of fraudulent companies. A good text classifier is a classifier that efficiently categorizes large sets of text documents in a reasonable time frame and with acceptable accuracy, and that provides classification rules that are humanly readable for possible fine-tuning. If the training of the classifier is also quick, this could become in some application domains a good asset for the classifier. Many techniques and algorithms for automatic text categorization have been devised. The text classification task can be defined as assigning category labels to new documents based on the knowledge gained in a classification system at the training stage. In the training phase, we are given a set of documents with class labels attached, and a classification system is built using a learning method. Classification is an important task in both data mining and machine learning communities, however, most of the learning approaches in text categorization are coming from machine learning research.

**SMS Spam Classifier:**

http://sms-spam-classing.herokuapp.com



Spam is unsolicited and unwanted messages sent electronically and whose content may be malicious. Email spam is sent/received over the Internet while SMS spam is typically transmitted over a mobile network. We'll refer to user that sent spam as 'spammers'.

Spam SMS vs Spam Emails

| **SMS** | **EMAIL** |
|---|---|
| Short messages | Can have any length |
| Sent (mostly) through mobile connections | Transmitted through any internet connection |
| Ambiguous intetion | Greater length makes the intention clearer |
| Content can be plain text, string characters and possibly emojis | Has a subject, formatted text, multimedia content and attachments |
| Usually regarded as trustworthy | There is widespread awareness about spam emails. |

**Algorithms used in SMS Spam Classifier:**

- **Multinomial Naive Bayes Classifier**

  Multinomial Naive Bayes algorithm is a probabilistic learning method that is mostly used in Natural Language Processing (NLP). The algorithm is based on the Bayes theorem and predicts the tag of a text such as a piece of email or newspaper article. It calculates the probability of each tag for a given sample and then gives the tag with the highest probability as output.

  Naive Bayes classifier is a collection of many algorithms where all the algorithms share one common principle, and that is each feature being classified is not related to any other feature. The presence or absence of a feature does not affect the presence or absence of the other feature.

  **Advantages**

  ▪ It is easy to implement as you only have to calculate probability.
  ▪ You can use this algorithm on both continuous and discrete data.
  ▪ It is simple and can be used for predicting real-time applications.
  ▪ It is highly scalable and can easily handle large datasets.

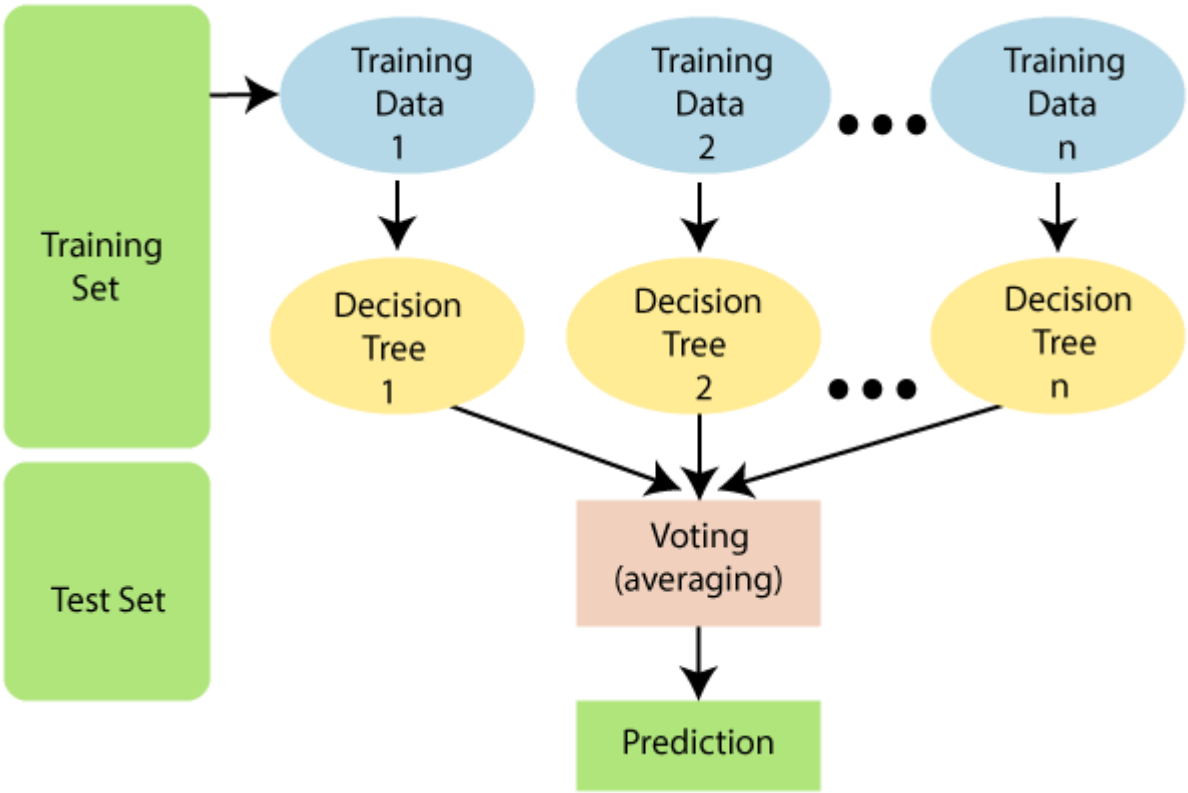
  **Disadvantages**

  ▪ The Naive Bayes algorithm has the following disadvantages:
  ▪ The prediction accuracy of this algorithm is lower than the other probability algorithms.
  ▪ It is not suitable for regression. Naive Bayes algorithm is only used for textual data classification and cannot be used to predict numeric values.

  **Applications**

  ▪ Face recognition
  ▪ Weather prediction
  ▪ Medical diagnosis
  ▪ Spam detection
  ▪ Age/gender identification
  ▪ Language identification
  ▪ Sentimental analysis

- **Random Forest**
  - Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML.
  - It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.
  - As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset."
  - Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.
  - The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.



**Python Libraries that are used in SMS Spam Classifier are:**

- NLTK (Natural Language Toolkit)
  The Natural Language Toolkit (NLTK) is a platform used for building Python programs that work with human language data for applying in statistical natural language processing (NLP). NLTK is a standard python library that provides a set of diverse algorithms for NLP. It is one of the most used libraries for NLP.

- Scikit-Learn
  Scikit-learn is a free machine learning library for Python. It features various algorithms like support vector machine, random forests, and k-neighbours, and it also supports Python numerical and scientific libraries like NumPy and SciPy.

- Pandas
  Pandas is a Python library for data analysis. Pandas is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language.

- Numpy
  NumPy is a Python library used for working with arrays. NumPy stands for Numerical Python. NumPy contains a multi-dimensional array and matrix data structures.

- Seaborn
  Seaborn is a library that uses Matplotlib underneath to plot graphs. It will be used to visualize random distributions. Seaborn is a library for making statistical graphics in Python. Seaborn helps you explore and understand your data.

- Matplotlib
  Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib makes easy things easy and hard things possible. matplotlib.pyplot is a collection of command style functions that make matplotlib work like MATLAB.

## Platform used is Jupyter Notebook

- The Jupyter Notebook is the original web application for creating and sharing computational documents.
- It offers a simple, streamlined, document-centric experience.
- The Jupyter Notebook is an open source web application that you can use to create and share documents that contain live code, equations, visualizations, and text.
- JupyterLab is the latest web-based interactive development environment for notebooks, code, and data.
- Its flexible interface allows users to configure and arrange workflows in data science, scientific computing, computational journalism, and machine learning. A modular design invites extensions to expand and enrich functionality.

## Streamlit

- Streamlit is an open source app framework in Python language.
- It helps us create web apps for data science and machine learning in a short time.
- It is compatible with major Python libraries such as scikit-learn, Keras, PyTorch, SymPy(latex), NumPy, pandas, Matplotlib etc.
- This is the library that allows us to build frontend for our machine learning and data science apps by writing all the code in Python.
- Streamlit is an open source python based framework for developing and deploying interactive data science dashboards and machine learning models.

## Stages to build SMS Spam Classifier:

1. Data Cleaning
2. EDA
3. Text Preprocessing
4. Model Building
5. Evaluation
6. Improvement
7. Website
8. Deploy

## 1) Data Cleaning

Data cleaning is one of the important parts of machine learning. It plays a significant part in building a model. However, the success or failure of a project relies on proper data cleaning. We will clean messages by removing the unnecessary things.

In [6]:
```
df.info()
```

```
RangeIndex: 5572 entries, 0 to 5571
Data columns (total 5 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   v1          5572 non-null   object
 1   v2          5572 non-null   object
 2   Unnamed: 2  50 non-null     object
 3   Unnamed: 3  12 non-null     object
 4   Unnamed: 4  6 non-null      object
dtypes: object(5)
memory usage: 217.8+ KB
```

In [7]:
```
# drop last 3 cols
df.drop(columns=['Unnamed: 2','Unnamed: 3','Unnamed: 4'],inplace=True)
```

In [8]:
```
df.sample(5)
```

Out[8]:

| | v1 | v2 |
|---|---|---|
| 1947 | ham | The battery is for mr adewale my uncle. Aka Egbon |
| 2712 | ham | Hey you still want to go for yogasana? Coz if ... |
| 4428 | ham | Hey they r not watching movie tonight so i'll ... |
| 3944 | ham | I will be gentle princess! We will make sweet ... |
| 49 | ham | U don't know how stubborn I am. I didn't even ... |

## 2) EDA

EDA stands for Exploratory Data Analysis. Exploratory Data Analysis (EDA) is an approach to analyze the data using visual techniques. It is used to discover trends, patterns, or to check assumptions with the help of statistical summary and graphical representations.

In [29]:
```
df.head()
```

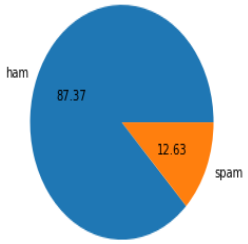
Out[29]:

| | target | text |
|---|---|---|
| 0 | 0 | Go until jurong point, crazy.. Available only ... |
| 1 | 0 | Ok lar... Joking wif u oni... |
| 2 | 1 | Free entry in 2 a wkly comp to win FA Cup fina... |
| 3 | 0 | U dun say so early hor... U c already then say... |
| 4 | 0 | Nah I don't think he goes to usf, he lives aro... |

In [31]:
```
df['target'].value_counts()
```

Out[31]:
```
0    4516
1     653
Name: target, dtype: int64
```

In [33]:
```
import matplotlib.pyplot as plt
plt.pie(df['target'].value_counts(), labels=['ham','spam'],autopct="%0.2f")
plt.show()
```



## 3) Text Preprocessing

Text preprocessing involves transforming text into a clean and consistent format that can then be fed into a model for further analysis and learning. Text preprocessing is a method to clean the text data and make it ready to feed data to the model. Text data contains noise in various forms like emotions, punctuation, text in a different case. It helps to get rid of unhelpful parts of the data, or noise, by converting all characters to lowercase, removing punctuations marks, and removing stop words.

- Lower case - There is a common approach to lowercasing everything for the sake of simplicity. It helps to maintain the consistency flow during the NLP tasks and text mining. The lower() function makes the whole process quite straightforward.
- Tokenization - Tokenizing is like splitting a whole sentence into words. You can consider a simple separator for this purpose. But a separator will fail to split the abbreviations separated by "." or special characters, like U.A.R.T., for example. Challenges increase when more languages are included. The "word_tokenize" module breaks the words into tokens and these words act as an input for the normalization and cleaning process.
- Removing special characters - Special characters are non-alphanumeric characters. These characters are most often found in comments, references, currency numbers etc. These characters add no value to text-understanding and induce noise into algorithms. Regular-expressions (regex) can be used to get rid of these characters and numbers.
- Removing stop words and punctuation - Stopwords are often added to sentences to make them grammatically correct, for example, words such as *a,* is, an, the, and etc. These stopwords carry minimal to no importance. These should be removed so machine learning algorithms can better focus on words which define the meaning of the text. We are using list from nltk.corpus and this list can further be enhanced by adding or removing custom words based on the situation at hand.

  This can be clubbed with step of removing special characters. Removing punctuation is fairly easy. It can be achieved by using string.punctuation and keeping everything which is not in this list.
- Stemming - There are many variations of words that do not bring any new information and create redundancy, ultimately bringing ambiguity when training machine learning models for predictions. Take "He likes to walk" and "He likes walking," for example. Both have the same meaning, so the stemming function will remove the suffix and convert "walking" to "walk."

In [187…

```
def transform_text(text):
    text = text.lower()
    text = nltk.word_tokenize(text)

    y = []
    for i in text:
        if i.isalnum():
            y.append(i)

    text = y[:]
    y.clear()

    for i in text:
        if i not in stopwords.words('english') and i not in string.punctuation:
            y.append(i)

    text = y[:]
    y.clear()

    for i in text:
        y.append(ps.stem(i))


    return " ".join(y)
```

In [192…

```
transform_text("I'm gonna be home soon and i don't want to talk about this stuff anymore tonight, k? I've cried enough today.")
```

Out[192… 'gon na home soon want talk stuff anymor tonight k cri enough today'

In [191…

```
df['text'][10]
```

Out[191… "I'm gonna be home soon and i don't want to talk about this stuff anymore tonight, k? I've cried enough today."

In [195…

```
df.head()
```

Out[195…

| | target | text | num_characters | num_words | num_sentences | transformed_text |
|---|---|---|---|---|---|---|
| 0 | 0 | Go until jurong point, crazy.. Available only ... | 111 | 24 | 2 | go jurong point crazi avail bugi n great world... |
| 1 | 0 | Ok lar... Joking wif u oni... | 29 | 8 | 2 | ok lar joke wif u oni |
| 2 | 1 | Free entry in 2 a wkly comp to win FA Cup fina... | 155 | 37 | 2 | free entri 2 wkli comp win fa cup final tkt 21... |
| 3 | 0 | U dun say so early hor... U c already then say... | 49 | 13 | 1 | u dun say earli hor u c alreadi say |
| 4 | 0 | Nah I don't think he goes to usf, he lives aro... | 61 | 15 | 1 | nah think goe usf live around though |

## 4) Model Building

Building an machine learning model requires splitting of data into 3 three sections which are 'Training data' ,'Validation data' and 'Testing data'. You train the classifier using 'training data set', tune the parameters using 'validation set' and then test the performance of your classifier on unseen 'test data set'.

In [525…
```
y = df['target'].values
```

In [526…
```
from sklearn.model_selection import train_test_split
```

In [527…
```
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size=0.2,random_state=2)
```

In [528…
```
from sklearn.naive_bayes import GaussianNB,MultinomialNB,BernoulliNB
from sklearn.metrics import accuracy_score,confusion_matrix,precision_score
```

In [489…
```
gnb = GaussianNB()
mnb = MultinomialNB()
bnb = BernoulliNB()
```

In [490…
```
gnb.fit(X_train,y_train)
y_pred1 = gnb.predict(X_test)
print(accuracy_score(y_test,y_pred1))
print(confusion_matrix(y_test,y_pred1))
print(precision_score(y_test,y_pred1))
```

```
0.8916827852998066
[[808  88]
 [ 24 114]]
0.5643564356435643
```

In [529…
```
mnb.fit(X_train,y_train)
y_pred2 = mnb.predict(X_test)
print(accuracy_score(y_test,y_pred2))
print(confusion_matrix(y_test,y_pred2))
print(precision_score(y_test,y_pred2))
```

```
0.971953578336557
[[896   0]
 [ 29 109]]
1.0
```

In [492…
```
bnb.fit(X_train,y_train)
y_pred3 = bnb.predict(X_test)
print(accuracy_score(y_test,y_pred3))
print(confusion_matrix(y_test,y_pred3))
print(precision_score(y_test,y_pred3))
```

```
0.9835589941972921
[[895   1]
 [ 16 122]]
0.991869918699187
```

**TF-IDF** stands for term frequency-inverse document frequency and it is a measure, used in the fields of information retrieval (IR) and machine learning, that can quantify the importance or relevance of string representations (words, phrases, lemmas, etc) in a document amongst a collection of documents(also known as a corpus).

TF-IDF is a widely used statistical method in natural language processing and information retrieval. It measures how important a term is within a document relative to a collection of documents.

TF-IDF can be broken down into two parts *TF (*term frequency*)* and *IDF (*inverse document frequency).

$$TF = \frac{\text{number of times the term appears in the document}}{\text{total number of terms in the document}}$$

$$IDF = log(\frac{\text{number of the documents in the corpus}}{\text{number of documents in the corpus contain the term}})$$

TF-IDF (Term Frequency - Inverse Document Frequency) is a handy algorithm that uses the frequency of words to determine how relevant those words are to a given document. It's a relatively simple but intuitive approach to weighting words, allowing it to act as a great jumping off point for a variety of tasks. This includes building search engines, summarizing documents, or other tasks in the information retrieval and machine learning domains.

In [522…
```
from sklearn.feature_extraction.text import CountVectorizer,TfidfVectorizer
cv = CountVectorizer()
tfidf = TfidfVectorizer(max_features=3000)
```

In [523…
```
X = tfidf.fit_transform(df['transformed_text']).toarray()
```

In [470…
```
#from sklearn.preprocessing import MinMaxScaler
#scaler = MinMaxScaler()
#X = scaler.fit_transform(X)
```

In [483…
```
# appending the num_character col to X
#X = np.hstack((X,df['num_characters'].values.reshape(-1,1)))
```

In [524…
```
X.shape
```

## 5) Evaluation

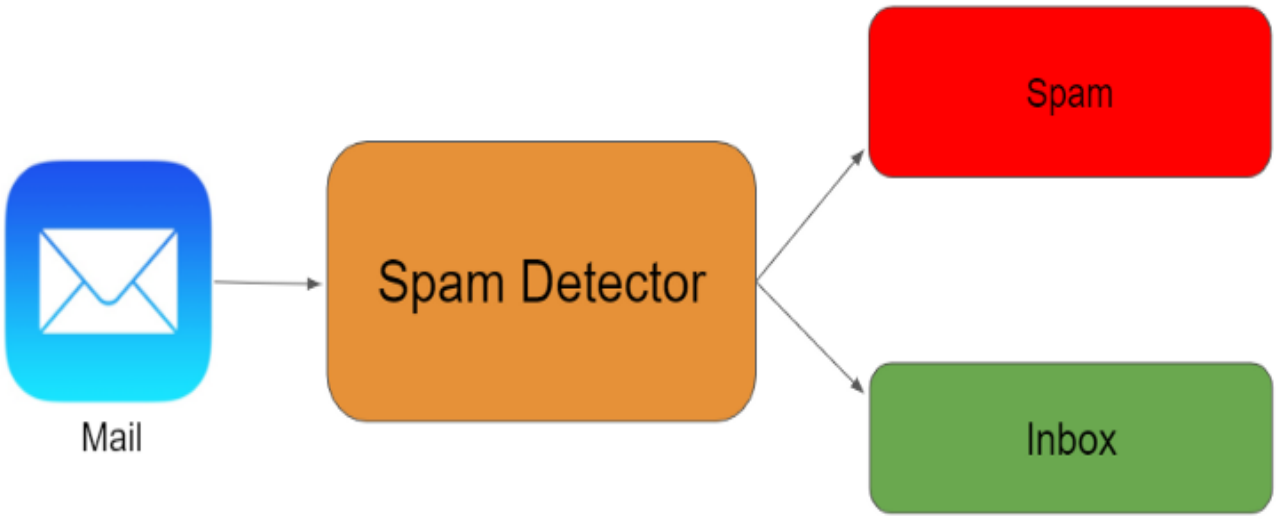
Model evaluation is the process of using different evaluation metrics to understand a machine learning model's performance, as well as its strengths and weaknesses. Model Evaluation is an integral part of the model development process. It helps to find the best model that represents our data and how well the chosen model will work in the future.

After evaluation we need to do some improvement by using tf-idf (term frequency-inverse document frequency) if needed and then we will make a website and deployment of model shold be done. Heroku is a platform as a service (PaaS) that enables developers to build, run, and operate applications entirely in the cloud. Heroku is used to deploy our project of SMS Spam Classifier.

**NOTE:** In our spam detection model, precision score matters more than accuracy because the data given is imbalance.

# SCOPE & OBJECTIVE

Implementing spam filtering is extremely important for any organization. Not only does spam filtering help keep garbage out of email inboxes, it helps with the quality of life of business emails because they run smoothly and are only used for their desired purpose. Spam filtering is essentially an anti-malware tool, as many attacks through email are trying to trick users to click on a malicious attachment, asking them to supply their credentials, and much more. Understanding the problem is a crucial first step in solving any machine learning problem. We will explore and understand the process of classifying emails as spam or not spam. This is called Spam Detection, and it is a binary classification problem. The reason to do this is simple: by detecting unsolicited and unwanted emails, we can prevent spam messages from creeping into the user's inbox, thereby improving user experience.




*Emails are sent through a spam detector. If an email is detected as spam, it is sent to the spam folder, else to the inbox.*

# METHODOLOGY

**System Architecture:**

The main objective of our approach is to classify the spam SMS messages as soon as it received on the mobile phone, regardless of newly created spam message (zero-hour attack). In this, we firstly collected dataset and finalized the features for our experiment. After finalizing features, we extracted the features from the messages (ham and spam) to create a feature vector.

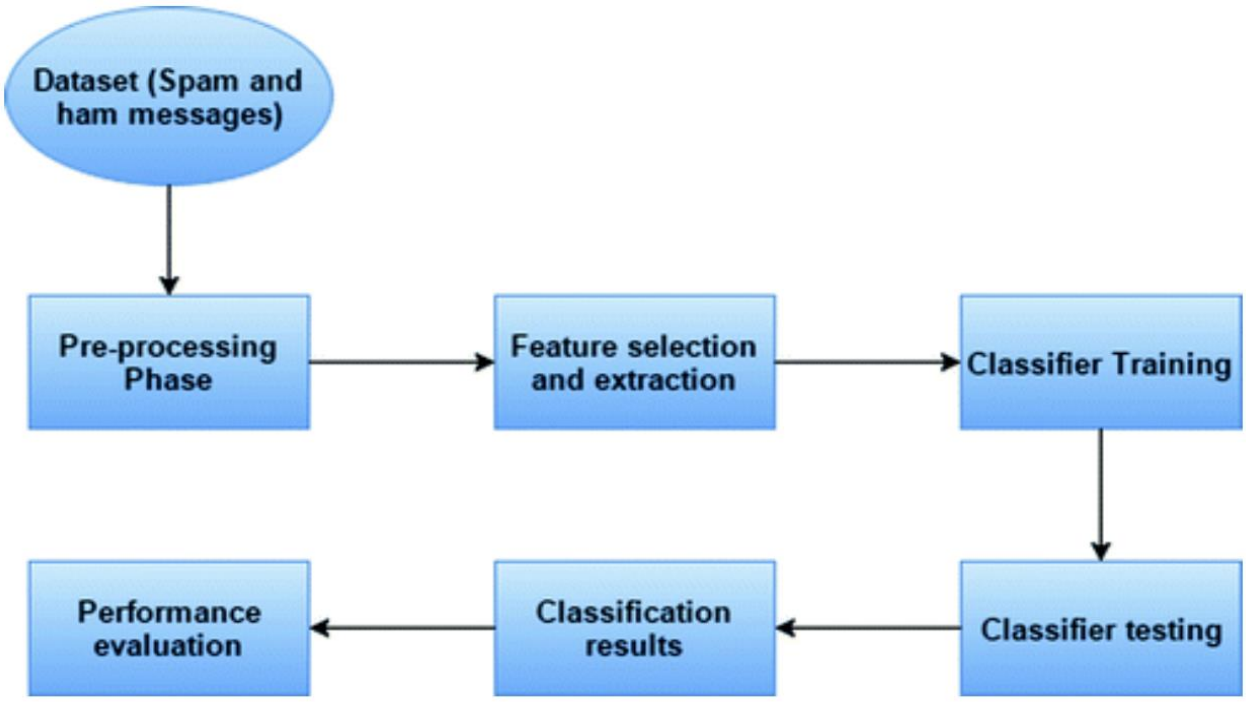



**Fig.1** System Architecture

Figure1 shows the system architecture of our proposed approach. In the coaching section, a binary classifier is generated by applying the feature vectors of spam and ham messages.

In the testing section, the classifier determines whether or not a replacement message may be a spam or not. At the end we get classification results for different machine learning algorithms and performance is evaluated for each machine learning algorithm such that we can get the best algorithm for our proposed approach.

Feature choice may be a vital task for the SMS Spam filtering. Selected options ought to be correlate to the message sort specified accuracy for detection of spam message are often enlarged. There is a length limit for SMS message and it contains solely text (i.e.no file attachments, graphics, etc.) while in the email, there is no text limit and it contains attachments, graphics, etc. SMS message is usually of two types i.e. ham (legitimate) message and spam message. Fig 1.System Architecture Identification of fine feature that may expeditiously filter spam. SMS messages could be a difficult task. Moreover, we have a tendency to study the characteristics of spam messages exhaustive and notice some options, that area unit helpful within the economical detection of spam SMS.

# ADVANTAGES / DISADVANTAGES

**Advantages of SMS Spam Classifier:**

- Spam filters save time that you could have wasted on removing spam from your Inbox.
- The previously reported spam messages will no longer hit your Inbox.
- Spam classifier can be used for sms and emails also.
- It can easily classify spam and non-spam messages.

**Disadvantages of SMS Spam Classifier:**

- Thousands of spam emails may reach Inboxes before a spammer's email address, IP or domain is blacklisted.
- Spam filtering is machine-based so there is a room for mistakes called "false positives."
- Bayesian filters may be fooled by spammers, e.g. in a case of using large blocks of legitimate text.
- A legitimate email sender may use potentially "suspicious" words without knowing it and trigger a spam filter.

# CONCLUSION

Detection of spam is important for securing message and e-mail communication. The accurate detection of spam is a big issue, and many detection methods have been proposed by various researchers. However, these methods have a lack of capability to detect the spam accurately and efficiently.

To solve this issue, we have proposed a method for spam detection using machine learning predictive models. The method is applied for the purpose of detection of spam. The experimental results obtained show that the proposed method has a high capability to detect spam. The proposed method achieved 99% accuracy which is high as compared with the other existing methods. Thus, the results suggest that the proposed method is more reliable for accurate and on-time detection of spam, and it will secure the communication systems of messages and e-mails. The whole project was divided into several iterations.

Each iteration was completed by completing four phases: inception, where the idea of work was identified; elaboration, where architecture of the part of the system is designed; construction, where existing code is implemented; transition, where the developed part of the project is validated.

# FUTURE SCOPE

In the future, we plan to deal with more challenging problems such as the analysis and management of report in spam SMS filters storing. Solution for this problem is another focus of work in the future.

However, there are still some parts that can be improved: for example, adding additional filtering techniques or changing aspects of the existing ones. The changes such as incrementing or decrementing the number of interesting words of the message and reorganizing the formula for calculating interesting rate can be done later.

We introduced Machine Learning concepts for prediction of spam based on the content data like only message content. In future work we can elaborate this topic to prediction by using content and context data like Host address of the SMS, sender, number of times received, URL‟s in the messages etc.

# REFERENCES


- SMS Spam Collection Data Set from UCI Machine Learning Repository, http://archive.ics.uci.edu/ml/datasets/SMS+Spam+Collection.
- SMS, "Spam collection dataset," 2019, https://www.kaggle.com/datasets.
- https://www.ijstr.org/final-print/feb2020/Spam-Detection-In-Sms-Using-Machine-Learning-Through-Text-Mining.pdf
- https://www.ijcsmc.com/docs/papers/June2021/V10I6202102.pdf
- https://ieeexplore.ieee.org/document/9734128/
- https://www.hindawi.com/journals/scn/2020/8873639/
- https://towardsdatascience.com/spam-detection-in-emails-de0398ea3b48
- https://www.kaggle.com/datasets/uciml/sms-spam-collection-dataset
- https://medium.com/analytics-vidhya/sms-spam-classifier-natural-language-processing-1751e2b324ed
- https://www.researchgate.net/publication/340607093_An_Effective_Model_for_SMS_Spam_Detection_Using_Content-based_Features_and_Averaged_Neural-Network