DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

A

PROJECT REPORT

ON

"SPAM DETECTION "

Submitted in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

BY

POOJA KUMARI
1NH16CS415

Under the guidance of

Mr. Vijay Kumar R
Assistant Professor,
Dept. of CSE, NHCE

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be impossible without the mention of the people who made it possible, whose constant guidance and encouragement crowned our efforts with success.

I have great pleasure in expressing my deep sense of gratitude to **Dr. Mohan Manghnani**, Chairman of New Horizon Educational Institutions for providing necessary infrastructure and creating good environment.

I take this opportunity to express my profound gratitude to **Dr. Manjunatha** , Principal NHCE, for his constant support and encouragement.

I am grateful to **Dr.Prashanth C.S.R**, Dean Academics, for his unfailing encouragement and suggestions, given to me in the course of my internship work.

I would also like to thank **Dr. B. Rajalakshmi** , Professor and Head, Department of Computer Science and Engineering, for her constant support.

I express my gratitude to **Mr. Vijay Kumar R** , Assistant Professor, my internship guide, for constantly monitoring the development of the project and setting up precise deadlines. Her valuable suggestions were the motivating factors in completing the work.

Finally a note of thanks to the teaching and non-teaching staff of Dept of Computer Science and Engineering, for their cooperation extended to me, and my friends, who helped me directly or indirectly in the course of the internship work.

POOJA KUMARI (1NH16CS415)

CONTENTS

1.	**INTRODUCTION**	**1**
	1.1 INTRODUCTION TO MACHINE LEARNING	1
	1.2 INTRODUCTION TO PROJECT	3
	1.3 OBJECTIVES	3
	1.4 PURPOSE OF THE PROJECT	4
	1.5 SCOPE OF THE PROPOSED PROJECT	4
	1.6 BACKGROUND THEORY	4
	1.7 ORGANIZATION OF PROJECT	4
2.	**METHODS**	**6**
	2.1 INTRODUCTION TO PYTHON LEARNING	6
	2.2 TOOLS USED FOR PYTHON	8
	2.3 MACHINE LEARNING	8
	2.4 MACHINE LEARNING TASKS	9
	2.5 MACHINE LEARNING APPLICATIONS	13
3.	**METHODOLOGY**	**15**
	3.1 DATA COLLECTION	17
	3.2 DATA PREPROCESSING	17
	3.3 ALGORITHEM USED	18
	3.4 FEATURE EXTRACTION AND INITIAL ANALYSIS	20
	3.5 CLASSIFICATION METHOD	21
	3.6 ARCHITECTURE OF SPAM DETECTION	23
4.	**REQUIREMENT SPECIFICATION**	**25**
	4.1 NON-FUNCTIONAL REQUIREMENT	25
	4.2 HARDWARE REQUIREMENT	27
	4.3 SOFTWARE REQUIREMENTS	27
	4.4 SOFTWARE USED	27
5.	**RESULT ANALYSIS**	**32**
6.	**CONCLUSION**	**39**
7.	**Future scope**	**40**
	REFERENCE	

S

ABSTRACT

This project addresses the problem of sentiment analysis in post i.e., classifying messages according to the slang words detected in them: toxic, severe_toxic, obscene, threat, insult, identity_hate.The Social Media these days is used ruthlessly by the user to post anything they want without thinking of the outcomes related to the post, sometimes the message posted by any user results to riots, other problems in society, and sometimes is personal to a person insult. Hence it is quite necessary to control the use of social media ruthlessly. The use of social media platforms has increased a lot in past 10 years and various cases were found where people got hurt from a user post or other cases riots cases have been noted due to these posts.

This model is basically built to control these ruthless social media messages from getting posted just by detecting the severity of the message by using machine learning classifiers Here one vs rest classifier is used to train the model with multiple classes the severity of the message will be classified with these models .The input text can be taken and the severity according to the class will define whether the message will be posted or not.

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be impossible without the mention of the people who made it possible, whose constant guidance and encouragement crowned our efforts with success.

I have great pleasure in expressing my deep sense of gratitude to **Dr. Mohan Manghnani**, Chairman of New Horizon Educational Institutions for providing necessary infrastructure and creating good environment.

I take this opportunity to express my profound gratitude to **Dr. Manjunatha** , Principal NHCE, for his constant support and encouragement.

I am grateful to **Dr.Prashanth C.S.R**, Dean Academics, for his unfailing encouragement and suggestions, given to me in the course of my internship work.

I would also like to thank **Dr. B. Rajalakshmi** , Professor and Head, Department of Computer Science and Engineering, for her constant support.

I express my gratitude to **Mr. Vijay kumar R** , Assistant Professor, my internship guide, for constantly monitoring the development of the project and setting up precise deadlines. Her valuable suggestions were the motivating factors in completing the work.

Finally a note of thanks to the teaching and non-teaching staff of Dept of Computer Science and Engineering, for their cooperation extended to me, and my friends, who helped me directly or indirectly in the course of the internship work.

POOJA KUMARI (1NH16CS415)

CONTENTS

1. INTRODUCTION **1**

2. DESCRIPTION OF THE ORGANIZATION **2**

3. DESCRIPTION OF THE DEPARTMENT **4**

3.1. TEAM LEADER 4

3.2. ANDROID DEVELOPERS 4

4. TECHNICAL DESCRIPTION **5**

4.1. DESCRIPTION ABOUT THE INTERNSHIP WORK 5

4.2. TECHNOLOGY USED 5

4.3. SYSTEM REQUIREMENTS 6

5. GENERAL ROLES AND RESPONSIBILITIES **8**

6. DESCRIPTION OF VARIOUS ACTIVITIES CARRIED OUT **9**

7. TECHANICAL TAKEAWAY **12**

7.1. ANDROID STUDIO 12

7.2. ANDROID 13

7.3. JAVA 15

7.4. GOOD CODING PRACTICES 17

8. PROFESSIONAL TAKEAWAY **18**

8.1. ETIQUETTE 18

8.2. BUSINESS ATTIRE 20

8.3. HANDSAKE 21

9. CONCLUSION **23**

CHAPTER 1

INTRODUCTION

1.1INTRODUCTION TO MACHINE LEARNING

Machine Learning is an idea to learn from examples and experience, without being explicitly programmed. Instead of writing code, you feed data to the generic algorithm, and it builds logic based on the data given.

For example, one kind of algorithm is a classification algorithm. It can put data into different groups. The classification algorithm used to detect handwritten alphabets could also be used to classify emails into spam and not-spam.

Consider playing checkers.

E = the experience of playing many games of checkers

T = the task of playing checkers.

P = the probability that the program will win the next game.

Examples of Machine Learning

There are many examples of machine learning. Here are a few examples of classification problems where the goal is to categorize objects into a fixed set of categories.

Face detection: Identify faces in images (or indicate if a face is present).

Email filtering: Classify emails into spam and not-spam.

Medical diagnosis: Diagnose a patient as a sufferer or non-sufferer of some disease.

Weather prediction: Predict, for instance, whether or not it will rain tomorrow.

Need of Machine Learning:

Machine Learning is a field which is raised out of Artificial Intelligence (AI). Applying AI, we wanted to build better and intelligent machines. But except for few mere tasks such as finding the shortest path between point A and B, we were unable to program more complex and constantly evolving challenges. There was a realisation that the only way to be able to achieve this task was to let machine learn from itself. This sounds similar to a child learning from its self. So machine learning was developed as a new capability for computers. And now machine learning is present in so many segments of technology, that we don't even realise it while using it.

Finding patterns in data on planet earth is possible only for human brains. The data being very massive, the time taken to compute is increased, and this is where Machine Learning comes into action, to help people with large data in minimum time.If big data and cloud computing are gaining importance for their contributions, machine learning as technology helps analyse those big chunks of data, easing the task of data scientists in an automated process and gaining equal importance and recognition.

The techniques we use for data mining have been around for many years, but they were not effective as they did not have the competitive power to run the algorithms. If you run deep learning with access to better data, the output we get will lead to dramatic breakthroughs which is machine learning.

1.2 INTRODUCTION TO PROJECT

Short messaging service (henceforth referred to as SMS) has become an inseparable part of modern society with the explosive penetration of the mobile phones. Spammers take advantage of this fact and make use of SMS message to reach potential customers to drive their business interest. This issue is growing by the day, thereby necessitating a mechanism for mobile SMS spam filtering. Mobile SMS spam filtering challenge is similar to email spam filtering, with the difference that they can send a limited number of characters only. It is noticed that almost all spam SMS text may contain a very close pattern due to this limitation.

It incorporates some “catch words” to attract potential “customers” and then some contact information, usually a call back number, reply SMS number or a URL (Uniform Resource Locator) that they can visit, at the least, a keyword that they can search. The fact that the number of characters in each message is limited should make it possible for the search methods to come out with better results. The spam filtering problem essentially is a case of text classification.

We will evaluate various algorithms used for spam filtering on SMS spam corpus in an attempt to identify the better methods so that they can be further optimized for the SMS text paradigm.

1.3OBJECTIVES

The following are the objectives of the project:

1. user needs to be able to distinguish spam from legitimate message. To do this he needs to identify typical spam characteristics and practices.

2.We need to stop the spams as much as possible.

3.Updation of database of the three lists i.e. black list, white list and suspicious list.

1.4 PURPOSE OF THE PROJECT

The purpose of Spam Sms Detection is to explore the results of applying machine learning techniques to detect Message spam detection. SMS spam (sometimes called cell phone spam) is any junk message delivered to a mobile phone as text messaging through the Short

Message Service (SMS). The dataset for this project originates from the UCI Machine Learning Repository.

1.5 SCOPE OF PROPOSED PROJECT

Use different approaches to establish relation between the text and the category SPAM or HAM like, based on size of message, word count, special keywords. Then build classification models using different techniques to distinguish spam sms. Compare accuracy of each technique and plot the accuracy graphs in a single bar plot.

1.6 BACKGROUND THEORY

We motivate the need for content-based SMS spam filtering. We discuss similarities differences between email and SMS spam filtering. We review recent research in SMS spam filtering. We analyse recent SMS spam messages and make a dataset available.

Early days, no consensus yet on best techniques but significant challenges exists. Mobile or SMS spam is a real and growing problem primarily due to the availability of very cheap bulk pre-pay SMS packages and the fact that SMS engenders higher response rates as it is a trusted and personal service.

SMS spam filtering is a relatively new task which inherits many issues and solutions from email spam filtering. However it poses its own specific challenges. It motivates work on filtering SMS spam and reviews recent developments in SMS spam filtering. Analyses a large corpus of SMS spam, and provides some initial benchmark results.

1.7 ORGANISATION OF PROJECT

The project report till gives only the introduction to application the description that follows gives you detail of what the system is and how system work. The most important part to follow in report is how we have realized our project including technologies and tools used, requirements analysis, how we planned to meet deadlines, software and hardware requirements both at server and client side, preliminary product description, various conception modes including class diagram, collaboration diagram, sequence diagram, use case diagram, activity diagram, ER diagram and other's.

Then follows system design that includes basic modules, data design, procedural design. User interfaces, security issues and test cases design. Finally report consists of implementing and testing details and at last conclusion future extension and improvements

CHAPTER 2

METHODS

2.1 INTRODUCTION TO PYTHON FOR MACHINE LEARNING

Python is one among the most popular dynamic programming languages that is being used today. Python is an open-source and object-oriented programming language developed by Dutchman Guido van Possum in 1980s. This language can be utilized for a wide range of applications like scripting, developing and testing. Due to its elegance and simplicity, top technology organizations like Dropbox, Google, Quora, Mozilla, Hewlett-Packard, Qualcomm, IBM, and Cisco have implemented Python.

Several websites state that Python is one among the most famous programming language of 2016. Because of its implementation and syntax, it pressures more on code readability. When compared to other programming languages like C++ and Java, it requires the programmer to develop lesser codes. It offers automatic memory management and several standard libraries for the programmer. Once a programmer completes Python certification training, he can gain knowledge and experience in a wide range of top IT organizations. It is a general-purpose and high-level coding language.

Because of its features, a large number of programmers across the world, showing interest in making use of this language to develop websites, GUI applications, and mobile applications. The main reason that brings Python one among the top coding languages is that it allows the developers to figure out the concepts by developing readable and less code. Several advantages of Python supports the programmers to alleviate the effort as well as the time required for developing complex and large applications.

WHY PYTHON FOR MACHINE LEARNING

Machine learning, in layman terms, is to use the data to make a machine make intelligent decision. For example — You can build a spam detection algorithm where the rules can be learned from the data or an anomaly detection of rare events by looking at previous data or arranging your email based on tags you had assigned by learning on email history and so on.

Machine learning is nothing but to recognise patterns in your data.An important task of a Machine learning engineer in his/her work life is to extract, process, define, clean, arrange and then understand the data to develop intelligent algorithms.

So for a Machine learning engineer/Computer Vision Engineer like me or a budding Data Scientist/Machine Learning/Algorithm Engineer/Deep learning engineer why I would recommend Python, **because it's easy to understand**.

Sometimes the concepts of Linear Algebra, Calculus are so complex, that they take the maximum amount of effort. A quick implementation in Python helps a ML engineer to validate an idea.

Data is the key

So it totally depends on the type of the task where you want to apply Machine learning. I work in computer vision projects. So the input data for me is the image or video. For someone else it would be a series of points over time or collection of language documents spreaded across various domains or audio files given or just some numbers.

Imagine everything that exists around you is data. **And it's raw, unstructured, bad, incomplete, large.** How Python can tackle all of them ? Lets see.

Packages, Packages everywhere !

Yes you guessed it right. It's the collection and code stack of various open source repositories which is developed by people (still in process) to continuously improve upon the existing methods.

Want to work with images — numpy, opencv, scikit

Want to work in text — nltk, numpy, scikit

Want to work in audio — librosa

Want to solve machine learning problem — pandas, scikit

Want to see the data clearly — matplotlib, seaborn, scikit

Want to use deep learning — tensorflow, pytorch

Want to do scientific computing — scipy

Want to integrate web applications — Django

2.2 TOOL USED FOR PYTHON

Spyder is the Scientific PYthon Development EnviRonment:

- A powerful interactive development environment for the Python language with advanced editing, interactive testing, debugging and introspection features
- A numerical computing environment thanks to the support of *IPython* (enhanced interactive Python interpreter) and popular Python libraries such as *NumPy* (linear algebra), *SciPy* (signal and image processing) or *matplotlib* (interactive 2D/3D plotting).

2.3 MACHINE LEARNING

Machine learning is an interdisciplinary field that uses statistical techniques to give computer systems the ability to "learn" (e.g., progressively improve performance on a specific task) from data, without being explicitly programmed.

The name machine learning was coined in 1959 by Arthur Samuel. Machine learning explores the study and construction of algorithms that can learn from and make predictions on data – such algorithms overcome following strictly static program instructions by making data-driven predictions or decisions, through building a model from sample inputs. Machine learning is employed in a range of computing tasks where designing and programming explicit algorithms with good performance is difficult or infeasible; example applications include email filtering, detection of network intruders, and computer vision.

Machine learning is closely related to (and often overlaps with) computational statistics, which also focuses on prediction-making through the use of computers. It has strong ties to

mathematical optimization, which delivers methods, theory and application domains to the field. Machine learning is sometimes conflated with data mining, where the latter subfield focuses more on exploratory data analysis and is known as unsupervised learning.

Within the field of data analytics, machine learning is a method used to devise complex models and algorithms that lend themselves to prediction; in commercial use, this is known as predictive analytics. These analytical models allow researchers, data scientists, engineers, and analysts to "produce reliable, repeatable decisions and results" and uncover "hidden insights" through learning from historical relationships and trends in the data.

2.4 MACHINE LEARNING TASKS

Machine learning tasks are typically classified into several broad categories:

Supervised learning: The computer is presented with example inputs and their desired outputs, given by a "teacher", and the goal is to learn a general rule that maps inputs to outputs. As special cases, the input signal can be only partially available, or restricted to special feedback.Supervised learning is the machine learning task of learning a function that maps an input to an output based on example input-output pairs.It infers a function from *labelled training data* consisting of a set of *training examples*. In supervised learning, each example is a *pair* consisting of an input object (typically a vector) and a desired output value (also called the *supervisory signal*).Some basic methods:

- Regression

  A regression problem is when the output variable is a real or continuous value, such as “salary” or “weight”. Many different models can be used, the simplest is the linear regression. It tries to fit data with the best hyper-plane which goes through the points.

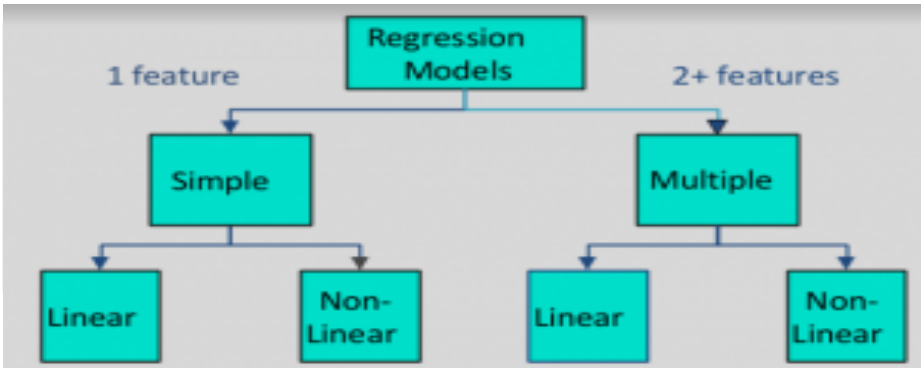


Fig. Types of regression models

- Classification

  A classification problem is when the output variable is a category, such as "red" or "blue" or "disease" and "no disease". A classification model attempts to draw some conclusion from observed values. Given one or more inputs a classification model will try to predict the value of one or more outcomes.

  For example, when filtering emails "spam" or "not spam", when looking at transaction data, "fraudulent", or "authorized". In short Classification either predicts categorical class labels or classifies data (construct a model) based on the training set and the values (class labels) in classifying attributes and uses it in classifying new data. There are a number of classification models. Classification models include logistic regression, decision tree, random forest, gradient-boosted tree, multilayer perceptron, one-vs-rest, and Naive Bayes.

Semi-supervised learning: The computer is given only an incomplete training signal: a training set with some (often many) of the target outputs missing.

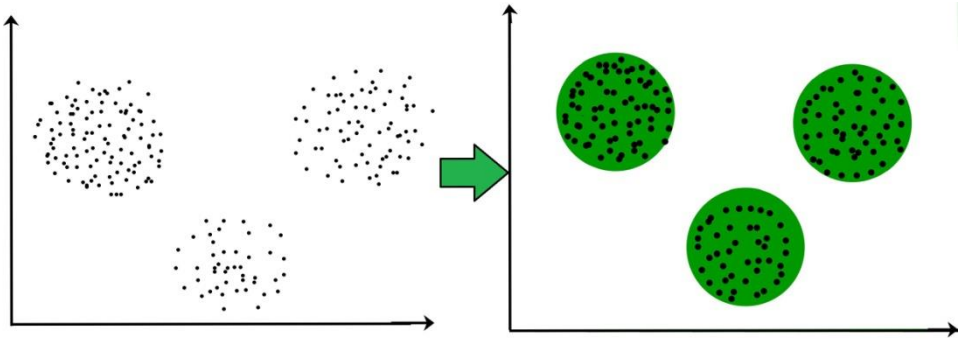
Active learning: The computer can only obtain training labels for a limited set of instances (based on a budget), and also has to optimize its choice of objects to acquire labels for. When used interactively, these can be presented to the user for labelling.

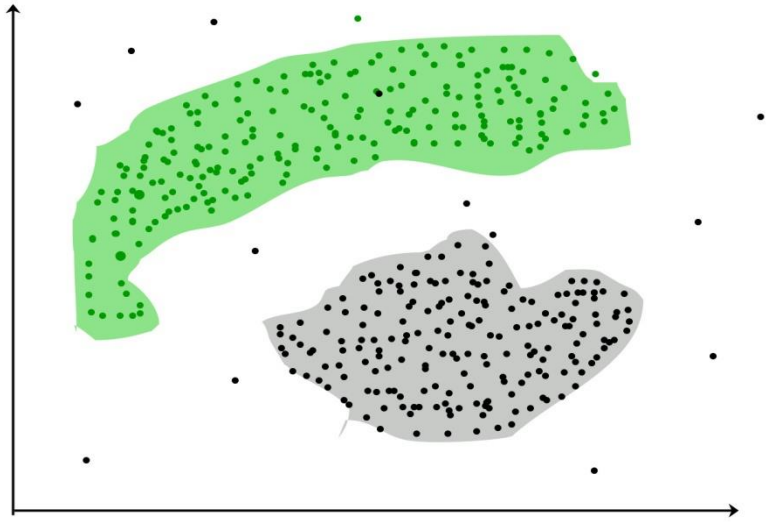
Unsupervised learning: No labels are given to the learning algorithm, leaving it on its own to find structure in its input. Unsupervised learning can be a goal in itself (discovering hidden patterns in data) or a means towards an end (feature learning).is the machine learning task of inferring a function that describes the structure of "unlabeled" data (i.e. data that has not been classified or categorized).Here some clustering methods are used.

- Clustering

  Clustering is the task of dividing the population or data points into a number of groups such that data points in the same groups are more similar to other data points in the same group and dissimilar to the data points in other groups. It is basically a collection of objects on the basis of similarity and dissimilarity between them.

  For ex– The data points in the graph below clustered together can be classified into one single group. We can distinguish the clusters, and we can identify that there are 3 clusters in the below picture.





Reinforcement learning: Data (in form of rewards and punishments) are given only as feedback to the program's actions in a dynamic environment, such as driving a vehicle or playing a game against an opponent.

2.5 MACHINE LEARNING APPLICATIONS

Another categorization of machine learning tasks arises when one considers the desired *output* of a machine-learned system:

In classification, inputs are divided into two or more classes, and the learner must produce a model that assigns unseen inputs to one or more (multi-label classification) of these classes. This is typically tackled in a supervised way. Spam filtering is an example of classification, where the inputs are email (or other) messages and the classes are "spam" and "not spam".

In regression, also a supervised problem, the outputs are continuous rather than discrete. In clustering, a set of inputs is to be divided into groups. Unlike in classification, the groups are not known beforehand, making this typically unsupervised task. Density estimation finds the distribution of inputs in some space.

Dimensionality reduction simplifies inputs by mapping them into a lower-dimensional space. Topic modelling is a related problem, where a program is given a list of human language documents and is tasked to find out which documents cover similar topics.

Among other categories of machine learning problems, learning to learn learns its own inductive bias based on previous experience. Developmental learning, elaborated for robot learning, generates its own sequences (also called curriculum) of learning situations to cumulatively acquire repertoires of novel skills through autonomous self-exploration and social interaction with human teachers and using guidance mechanisms such as active learning, maturation, motor synergies, and imitation. First, in testing and training,the data must be pre-processed so that they able to be used by the classifier. Then, the classifier operates in training or testing mode to either learn from the pre-processed data or classify future data.

CHAPTER 3

METHODOLOGY

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. These feature vectors are used for training and testing purposes. Our proposed system takes the decision based on ten features. In the training phase, a binary classifier is generated by applying the feature vectors of spam and ham messages. In the testing phase, the classifier determines whether a new message is 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 selection is a very important task for the SMS Spam filtering. Selected features should be correlated to the message type such that accuracy for detection of spam message can be increased. There is a length limit for SMS message and it contains only text (i.e. no file attachments, graphics, etc.) while in the email, there is no text limit and

Data Pre-processing / Data Munging: Data Pre-processing is carried in following steps:

1. Tokenize the tweet, i.e. split words from body of text.
2. Remove stop-words from the tokens (stop-words are the commonly usedwords which are irrelevant in text analysis like I, am, you, are, etc.)
3. Do POS (part of speech) tagging of the tokens and select on significantfeatures/tokens like adjectives, adverbs, etc.
4. Pass the tokens to a sentiment classifier which classifies the tweetsentiment as positive, negative or neutral by assigning it a polarity between -1.0 to 1.0Regression analysis is an important tool for modelling and analyzing data. Regression analysis also allows us to compare the effects of variables measured on different scales, such as the effect of price changes and the number of promotional activities.

These benefits help market researchers / data analysts / data scientists to eliminate and evaluate the best set of variables to be used for building predictive models. We split data into two category train and test. In which the testing data are 20% of all data and remains training data.

Types of Regressions –

- Linear Regression
- Logistic Regression
- Multinomial Regression
- Random forest

Classification methods: In machine learning and statistics, classification is the problem of identifying to which of a set of categories (sub- populations) a new observation belongs, on the basis of a training set of data containing observations (or instances) whose category membership is known. Classification is considered an instance of supervised learning, i.e. learning where a training set of correctly identified observations is available. An algorithm that implements classification, especially in a concrete implementation, is known as a classifier. The term "classifier" sometimes also refers to the mathematical function, implemented by a classification algorithm that maps input data to a category.

Multinomial logistic regression (often just called 'multinomial regression') isused to predict a nominal dependent variable given one or more independent variables. It is sometimes considered an extension of binomial logistic regression to allow for a dependent variable with more than two categories. Logistic Regression - In this machine learning algorithm the dependent variable is categorical and measures the relationship between the independent variable and categorical dependent variable using the logistic function. Random Forest - Random Forest algorithm is best machine algorithm for a large number of datasets. It basically constructs a set of decision trees at training phase and then each tree operates on randomly chosen attributes.

Data Visualization: Libraries 'matplotlib' and 'seaborn' are used to visualize the different Graphs and relations of the tweets. Bar plot of percentage of sentiments, relation sentiment score with time and number of tweets per month are plotted.

3.1 Data Collection

A testing of the intelligent methods needs to be done on SMS messages so that a conclusion can be drawn on the topic in consideration. We made use of a fairly large collection of SMS with over 5000 separate text messages of which about 15% are spam messages.

- The SMS Spam Collection is a public set of SMS labeled messages that have been collected for mobile phone spam project. It has one collection composed by 5,572 English, real and non-encoded messages, tagged according being legitimate (ham) or spam.

The algorithms we are interested in do not read strings and so we convert the data into feature vectors. I have used the Weka "StringToWordVector" function for this conversion.

3.2 Data Pre-processing / Data Munging:

Data Pre-processing is carried in following steps:

Tokenize the SMS, i.e. split words from body of text.

1. Remove stop-words from the tokens (stop-words are the commonly used words which are irrelevant in text analysis like I, am, you, are, etc.)
2. Do POS (part of speech) tagging of the tokens and select only significant features/tokens like adjectives, adverbs, etc.
3. Pass the tokens to a sentiment classifier which classifies the SMS sentiment as spam or ham by assigning it a polarity between -1.0 to 1.0

3.3 Algorithms Used

ONE VS REST CLASSIFIER ALGORITHM

One-vs.-rest (or *one-vs.-all*, OvA or OvR, *one-against-all*, OAA) strategy involves training a single classifier per class, with the samples of that class as positive samples and all other

samples as negatives. This strategy requires the base classifiers to produce a real-valued confidence score for its decision, rather than just a class label; discrete class labels alone can lead to ambiguities, where multiple classes are predicted for a single sample.

In pseudocode, the training algorithm for an OvA learner constructed from a binary classification learner L is as follows:

Inputs:

- L, a learner (training algorithm for binary classifiers)
- samples X
- labels y where $y_i \in \{1, \ldots K\}$ is the label for the sample X_i

Output:

- a list of classifiers f_k for $k \in \{1, \ldots, K\}$

Procedure:

- For each k in {1, …, K}
    - Construct a new label vector z where $z_i = 1$ if $y_i = k$ and $z_i = 0$ otherwise
    - Apply L to X, z to obtain f_k

Making decisions means applying all classifiers to an unseen sample x and predicting the label k for which the corresponding classifier reports the highest confidence score:

Although this strategy is popular, it is a heuristic that suffers from several problems. Firstly, the scale of the confidence values may differ between the binary classifiers. Second, even if the class distribution is balanced in the training set, the binary classification learners see unbalanced distributions because typically the set of negatives they see is much larger than the set of positives.

PIPELINE:

Pipeline of transforms with a final estimator.

Sequentially apply a list of transforms and a final estimator. Intermediate steps of the pipeline must be 'transforms', that is, they must implement fit and transform methods. The final estimator only needs to implement fit. The transformers in the pipeline can be cached using memory argument.

The purpose of the pipeline is to assemble several steps that can be cross-validated together while setting different parameters. For this, it enables setting parameters of the various steps using their names and the parameter name separated by a '__', as in the example below. A step's estimator may be replaced entirely by setting the parameter with its name to another estimator, or a transformer removed by setting to None.

TFIDF Vectorizer:

Word counts are a good starting point, but are very basic.

One issue with simple counts is that some words like "*the*" will appear many times and their large counts will not be very meaningful in the encoded vectors.

An alternative is to calculate word frequencies, and by far the most popular method is called TF-IDF. This is an acronym than stands for "*Term Frequency – Inverse Document*" Frequency which are the components of the resulting scores assigned to each word.

- **Term Frequency**: This summarizes how often a given word appears within a document.
- **Inverse Document Frequency**: This downscales words that appear a lot across documents.

Without going into the math, TF-IDF are word frequency scores that try to highlight words that are more interesting, e.g. frequent in a document but not across documents.

The TfidfVectorizer will tokenize documents, learn the vocabulary and inverse document frequency weightings, and allow you to encode new documents. Alternately, if you already have a learned CountVectorizer, you can use it with a TfidfTransformer to just calculate the inverse document frequencies and start encoding documents.

The same create, fit, and transform process is used as with the CountVectorizer.

Below is an example of using the TfidfVectorizer to learn vocabulary and inverse document frequencies across 3 small documents and then encode one of those documents.

3.4.FEATURE EXTRACXTION AND INTIAL ANALYSIS

As mentioned earlier, our dataset consists of one large text file in which each line corresponds to a text message. Therefore, **preprocessing** of the data, **extraction** of features, and **tokenization** of each message is required. After all the process the data is passed through TFIDF vectorizer to count the term frequency and inverse data frequency, after this the pipeline algorithm is combinedly used with One Vs Rest classifier algorithm to classify the model together for multiple class

For the initial analysis of the data, each message in dataset is split into tokens of alphabetic characters. Any space, comma, dot, or any special characters are removed from feature space for now, and alphabetic strings are stored as a token as long as they do not have any non-alphabetic characters in between. The effect of abbreviations and misspellings in the messages are ignored, and no word stemming algorithm is used. Additionally, three more tokens are generated based on the number of dollar signs ($), the number of numeric strings, and the overall number of characters in the message. The intuition behind entering the length of message as a feature is that the cost of sending a text message is the same as long as it is contained below characters, so marketers would prefer to use most of the space available to them as long as it doesn't exceed the limit. To train the model a data set 1.53 lakh records is provided to the model out which on splitting the dataset for test and train the 70% of the data is used for the training purpose and the rest 30% is used for the test

purpose. For initial analysis one vs rest classifier is used and the multiple class are predicted together for a single message passed by the user and before all this process several pre-processing tasks are performed over the text .

3.5 Classification Method

Gaussian naïve base algorithm

Naive Bayes classifier is a straightforward and powerful algorithm for the classification task. Even if we are working on a data set with millions of records with some attributes, it is suggested to try Naive Bayes approach.

Naive Bayes classifier gives great results when we use it for textual data analysis. Such as Natural Language Processing.Naive Bayes is a kind of classifier which uses the Bayes Theorem. It predicts membership probabilities for each class such as the probability that given record or data point belongs to a particular class. The class with the highest probability is considered as the most likely class. This is also known as **Maximum A Posteriori (MAP)**.

The MAP for a hypothesis is:

MAP(H)=max(P(H|E))= max((P(E|H)*P(H))/P(E))= max(P(E|H)*P(H))

P(E) is evidence probability, and it is used to normalize the result. It remains same so, removing it won't affect.Naive Bayes classifier assumes that all the features are **unrelated** to each other. Presence or absence of a feature does not influence the presence or absence of any other feature.

K-NEAREST NEIGHBOUR ALGORITHM

In pattern recognition, the ***k*-nearest neighbors algorithm** (***k*-NN**) is a non-parametric method used for classification and regression.In both cases, the input consists of the *k* closest training examples in the feature space. The output depends on whether *k*-NN is used for classification or regression:

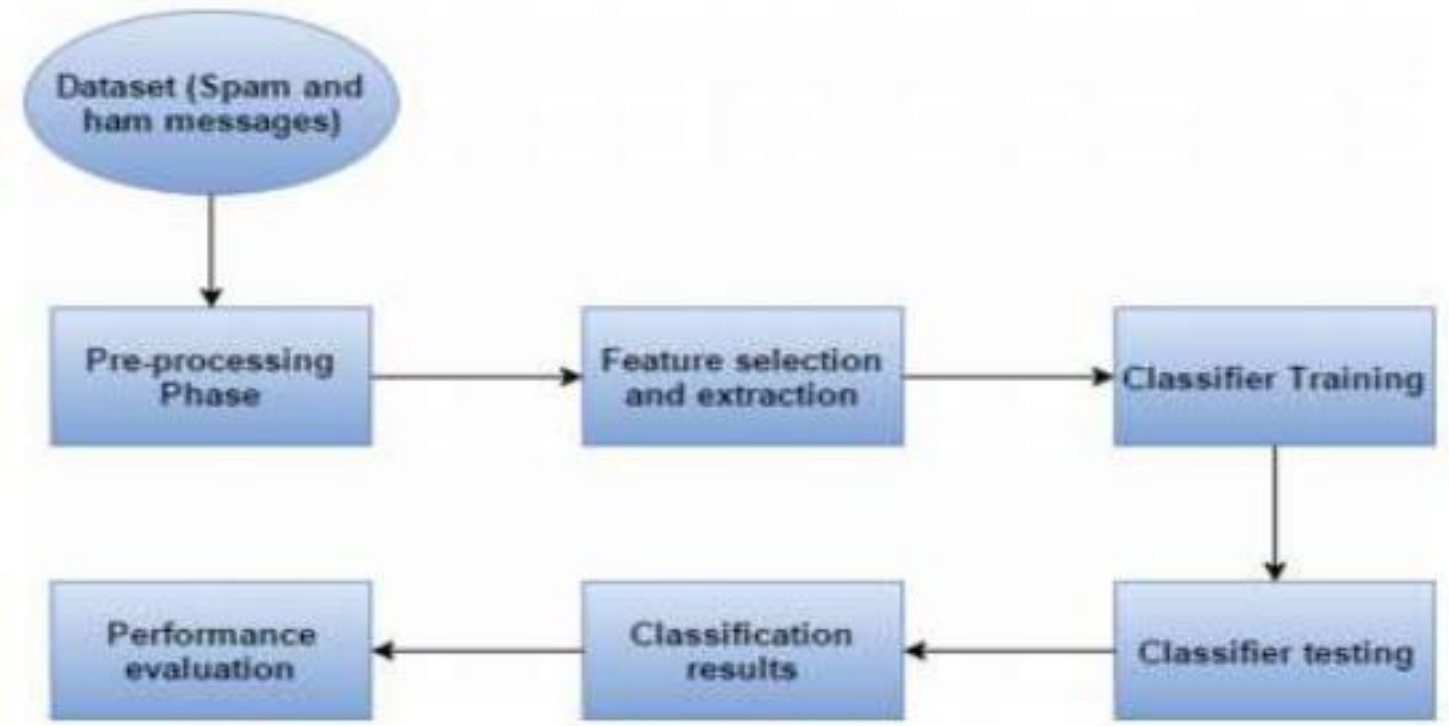
In *k-NN classification*, the output is a class membership. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its *k*

nearest neighbors (*k* is a positive integer, typically small). If *k* = 1, then the object is simply assigned to the class of that single nearest neighbor.

In *k-NN regression*, the output is the property value for the object. This value is the average of the values of its *k* nearest neighbors.*k*-NN is a type of instance-based learning, or lazy learning, where the function is only approximated locally and all computation is deferred until classification. The *k*-NN algorithm is among the simplest of all machine learning algorithms.

Both for classification and regression, a useful technique can be used to assign weight to the contributions of the neighbors, so that the nearer neighbors contribute more to the average than the more distant ones. For example, a common weighting scheme consists in giving each neighbor a weight of 1/*d*, where *d* is the distance to the neighbor.

The neighbors are taken from a set of objects for which the class (for *k*-NN classification) or the object property value (for *k*-NN regression) is known. This can be thought of as the training set for the algorithm, though no explicit training step is required. A peculiarity of the *k*-NN algorithm is that it is sensitive to the local structure of the data. The algorithm is not to be confused with *k*-means, another popular machine learning technique.



3.1 Architecture of spam message

3.6 FEATURE EXTRACXTION AND INTIAL ANALYSIS

As mentioned earlier, our dataset consists of one large text file in which each line corresponds to a text message. Therefore, **pre-processing** of the data, **extraction** of features, and **tokenization** of each message is required. After the feature extraction, an initial analysis on the data is done using naive Bayes (NB) algorithm with multinomial event model and Laplace smoothing, and based on the results, next steps are determined.

For the initial analysis of the data, each message in dataset is split into tokens of alphabetic characters. Any space, comma, dot, or any special characters are removed from feature space for now, and alphabetic strings are stored as a token as long as they do not have any non-alphabetic characters in between. The effect of abbreviations and misspellings in the messages are ignored, and no word stemming algorithm is used.

Additionally, three more tokens are generated based on the number of dollar signs ($), the number of numeric strings, and the overall number of characters in the message. The intuition behind entering the length of message as a feature is that the cost of sending a text message is the same as long as it is contained below 160 characters, so marketers would

prefer to use most of the space available to them as long as it doesn't exceed the limit. For the initial analysis of data, we have used the multinomial event model with Laplace smoothing. Extracting tokens for all messages in the dataset will result in 7,789 features.

However, not all of these features are useful in the classification. Going through the extracted tokens, we removed the ones with less than five and more than 500 times frequency in the dataset, since those tokens are either too rare or too common, and do not contribute to the content of the messages. These two thresholds are set by exploring different values and checking the performance of NB classification algorithm on results. Finally, the remaining tokens result in 1,552 features.

CHAPTER 4

REQUIREMENT SPECIFICATION

4.1 NON-FUNCTIONAL REQUIREMENTS

In systems engineering and needs engineering, a non-functional demand may be a demand that specifies criteria which will be wont to decide the operation of a system, instead of specific behaviors.This should be contrasted with useful needs that outline specific behavior or functions. The arrange for implementing useful needs is elaborate within the system style.

The arrange for implementing non-functional needs is elaborate within the system design.Other terms for non-functional needs ar "constraints", "quality attributes", "quality goals", "quality of service requirements" and "non-behavioral requirements".

Some of the quality attributes are as follows:

ACCESSIBILITY:

Accessibility may be a general term wont to describe the degree to that a product, device, service, or surroundings is accessible by as many folks as potential.

MAINTAINABILITY:

In software engineering, maintainability is the ease with which a software product can be modified in order to:

- Correct defects
- Met new requirements
- New functionalities can be added in the project based on the user requirements.

Since the programming is incredibly easy, it is easier to find and correct the defects and to make the changes in the project.

SCALABILITY:

System is capable of handling increase total throughput under an increased load when resources are added.System can work normally under situations such as low bandwidth and large number of users.

PORTABILITY:

Portability is one in every of the key ideas of high-level programming.Portability is that the code base feature to be ready to apply the prevailing code rather than making new code once moving code from Associate in Nursing surroundings to a different.

Project can be executed under different operation conditions provided it meet its minimum configurations. Only system files and dependant assemblies would have to be configured in such case.

4.2 HARDWARE REQUIREMENTS

CPU: Intel 2.1 GHZ

Memory: 4GB Disk: 100GB

Display: 15 inch color

4.3 SOFTWARE REQUIREMENTS

Coding: Python

Platform: Machine Learning Tool: Spyder

4.4 Software Used

Anaconda:

Anaconda is a free and open-source distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment. Package versions are managed by the package management system conda.

The Anaconda distribution is used by over 13 million users and includes more than 1400 popular data-science packages suitable for Windows, Linux, and MacOS.

Anaconda distribution comes with more than 1,400 packages as well as the Conda package and virtual environment manager, called Anaconda Navigator, so it eliminates the need to learn to install each library independently.

The open source packages can be individually installed from the Anaconda repository with the conda install command or using the pip install command that is installed with Anaconda. Pip packages provide many of the features of conda packages and in most cases they can work together.

Custom packages can be made using the conda build command, and can be shared with others by uploading them to Anaconda Cloud,PyPI or other repositories.

The default installation of Anaconda2 includes Python 2.7 and Anaconda3 includes Python 3.7. However, you can create new environments that include any version of Python packaged with conda.

Anaconda Navigator

Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda distribution that allows users to launch applications and manage conda packages, environments and channels without using command-line commands. Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository, install them in an environment, run the packages and update them. It is available for Windows, macOS and Linux.

The following applications are available by default in Navigator :

- JupyterLab
- Notebook
- QtConsole
- Spyder

- Glueviz
- Orange
- Rstudio
- Visual Studio Code

Conda

Conda is an open source, cross-platform,language-agnostic package manager and environment management system that installs, runs, and updates packages and their dependencies. It was created for Python programs, but it can package and distribute software for any language (e.g., R), including multi-language projects. The Conda package and environment manager is included in all versions of Anaconda, Miniconda Anaconda Repository.

Anaconda Cloud

Anaconda Cloud is a package management service by Anaconda where you can find, access, store and share public and private notebooks, environments, and conda and PyPI packages. Cloud hosts useful Python packages, notebooks and environments for a wide variety of applications. You do not need to log in or to have a Cloud account, to search for public packages, download and install them.

You can build new packages using the Anaconda Client command line interface (CLI), then manually or automatically upload the packages to Cloud.

Spyder :

Spyder is an open source cross-platform integrated development environment (IDE) for scientific programming in the Python language. Spyder integrates with a number of prominent packages in the scientific Python stack, including NumPy, SciPy, Matplotlib, pandas, IPython, SymPy and Cython, as well as other open source software. It is released under the MIT license.

Initially created and developed by Pierre Raybaut in 2009, since 2012 Spyder has been maintained and continuously improved by a team of scientific Python developers and the community.

Spyder is extensible with first- and third-party plugins, includes support for interactive tools for data inspection and embeds Python-specific code quality assurance and introspection instruments, such as Pyflakes, Pylint and Rope. It is available cross-platform through Anaconda, on Windows, on macOS through MacPorts, and on major Linux distributions such as Arch Linux, Debian, Fedora, Gentoo Linux, openSUSE and Ubuntu.

Spyder uses Qt for its GUI, and is designed to use either of the PyQt or PySide Python bindings.[10] QtPy, a thin abstraction layer developed by the Spyder project and later adopted by multiple other packages, provides the flexibility to use either backend.

Features

Features include:

- An editor with syntax highlighting, introspection, code completion
- Support for multiple IPython consoles
- The ability to explore and edit variables from a GUI
- A Help pane able to retrieve and render rich text documentation on functions, classes and methods automatically or on-demand.
- A debugger linked to IPdb, for step-by-step execution.
- Static code analysis, powered by Pylint.
- A run-time Profiler, to benchmark code.
- Project support, allowing work on multiple development efforts simultaneously.

- A built-in file explorer, for interacting with the filesystem and managing projects.
- A "Find in Files" feature, allowing full regular expression search over a specified scope.
- An online help browser, allowing users to search and view Python and package documentation inside the IDE.
- A history log, recording every user command entered in each console.
- An internal console, allowing for introspection and control over Spyder's own operation

Plugins

Available plugins include:

Spyder-Unittest, which integrates the popular unit testing frameworks Pytest, Unittest and Nose with Spyder

Spyder-Notebook, allowing the viewing and editing of Jupyter Notebooks within the IDE

Spyder-Reports, enabling use of literate programming techniques in Python

Spyder-Terminal, adding the ability to open, control and manage cross-platform system shells within Spyder

Spyder-Vim, containing commands and shortcuts emulating the Vim text editor

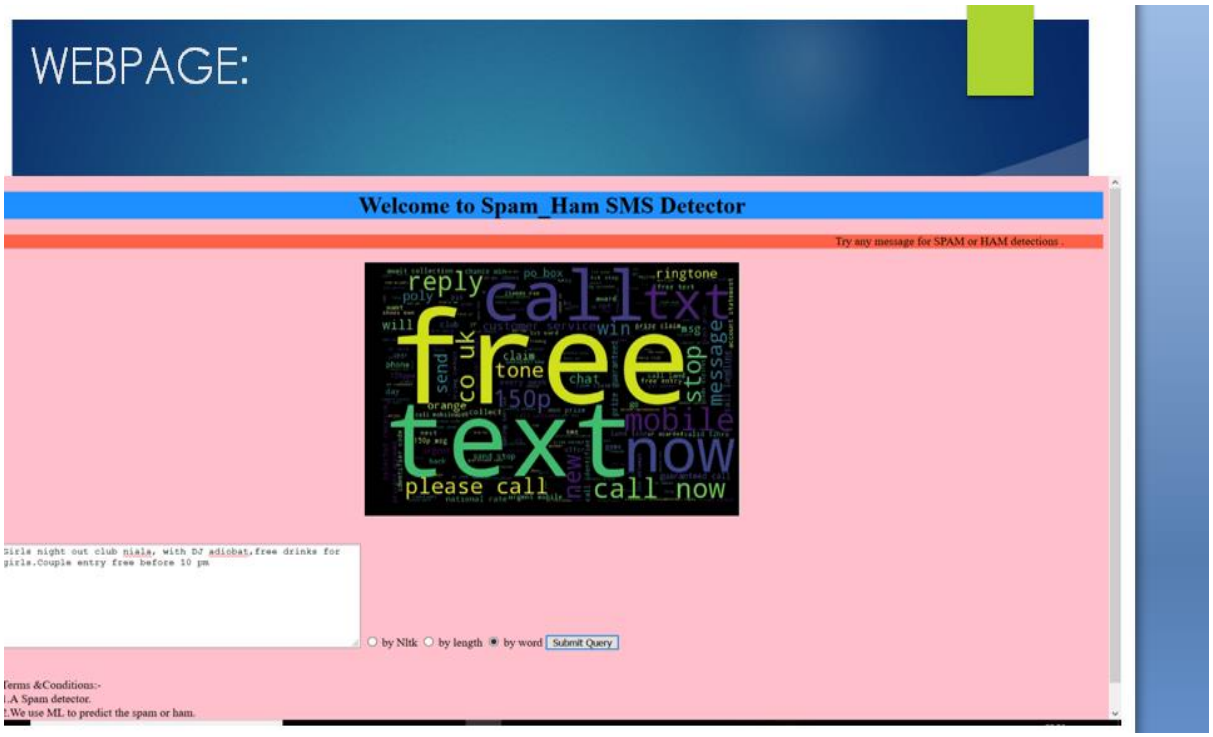
Spyder-AutoPEP8, which can automatically conform code to the standard PEP 8 code style

Spyder-Line-Profiler and Spyder-Memory-Profiler, extending the built-in profiling functionality to include testing an individual line, and measuring memory usage

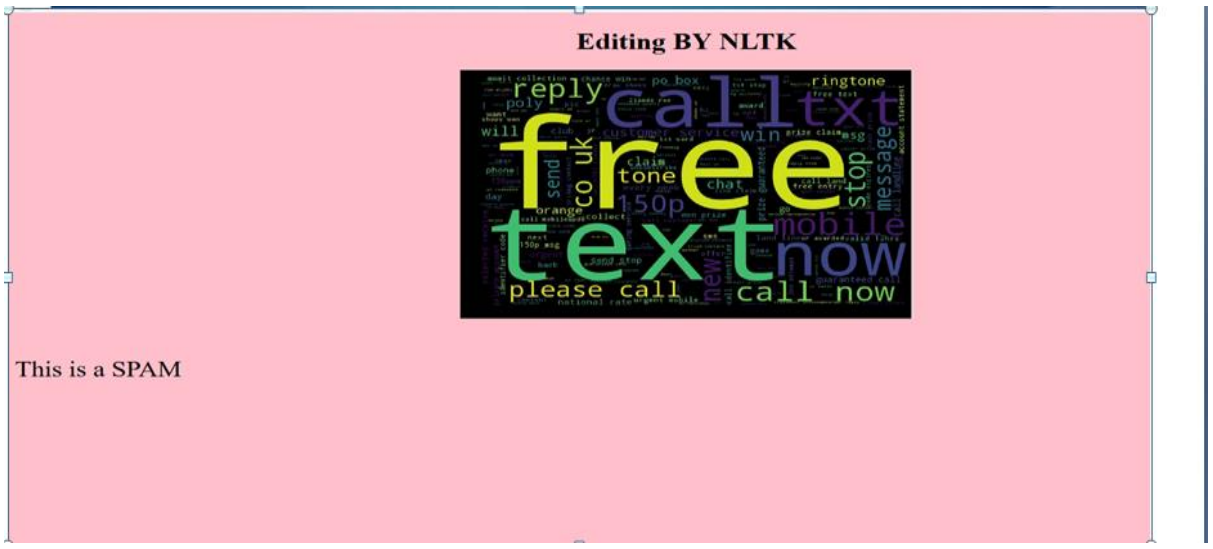
CHAPTER 5

OUTPUT

Screenshot1



Screenshot2



Code Screenshots

Creatingdataset:

```
import pandas as pd
import warnings
warnings.filterwarnings('ignore')
def ham(check_spam_ham):
    data = pd.read_csv("spam_ham.csv",encoding='latin-1')


    #Drop column and name change
    data = data.rename(columns={"v1":"label", "v2":"text"})
    #Count observations in each label
    data.label.value_counts()
```

Split dataset:

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(data["text"],
            data["label"], test_size = 0.2, random_state = 10)


from sklearn.feature_extraction.text import CountVectorizer
vect = CountVectorizer()
vect.fit(X_train)
```

ApplyNLP:

```
import nltk
for val in spam.text:
    text = val.lower()
    tokens = nltk.word_tokenize(text)
    #tokens = [word for word in tokens if word not in stopwo
    for words in tokens:
        spam_words = spam_words + words + ' '
   for val in ham.text:
    text = val.lower()
    tokens = nltk.word_tokenize(text)
    for words in tokens:
        ham_words = ham_words + words + ' '
```

```
test_str_series.append(check_spam_ham)
    test_str_df = vect.transform(test_str_series)

    predict_result=model.predict(test_str_df)

    return predict_result[0]
```

Prediction of results:

CONFUSION MATRICS

	0	1
0	900	49
1	103	63

CONFUSION MATRIX(BYLENGTH)

	0	1
0	822	127
1	24	142

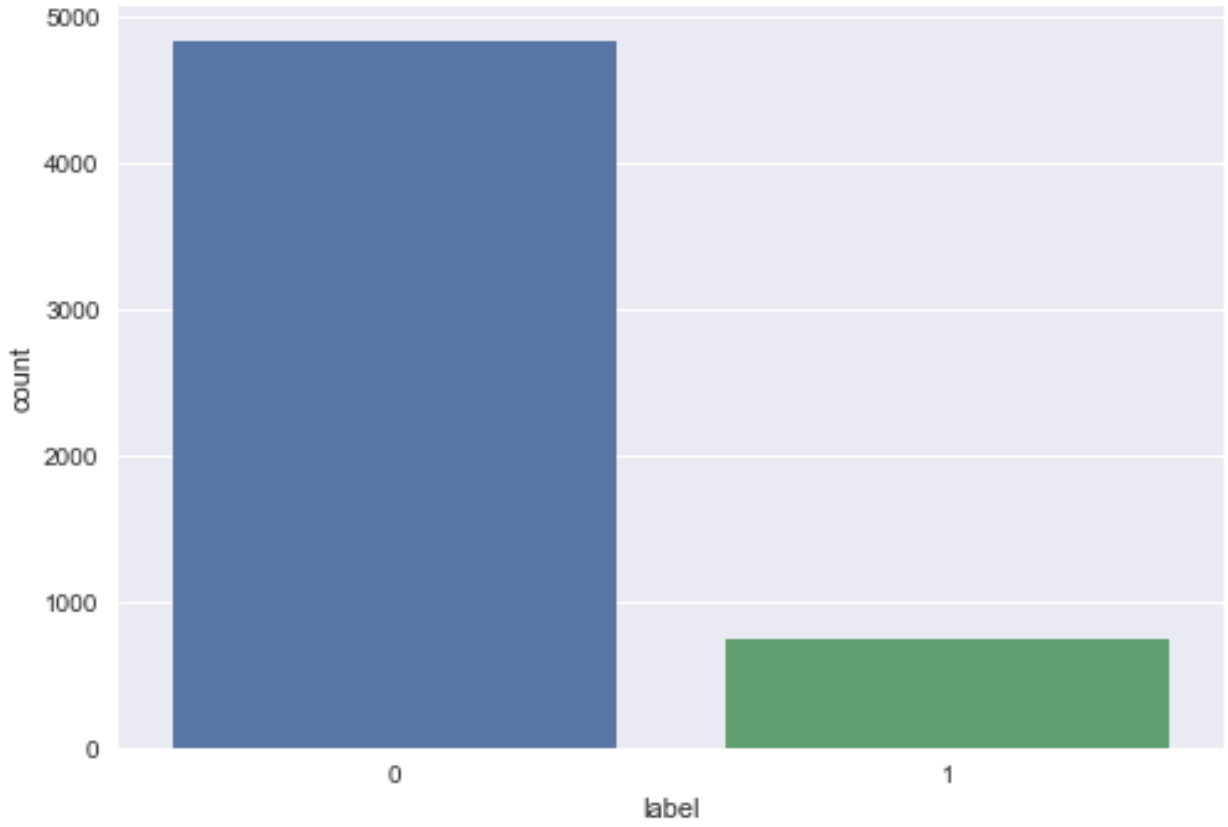
CONFUSION MATRIX(BYWORDCOUNT)

	0	1
0	925	24
1	151	15

CONFUSIONMATRIX(BY NATURAL LANGUAGE PROCESSING)

5.1 RESULTSANALYSIS

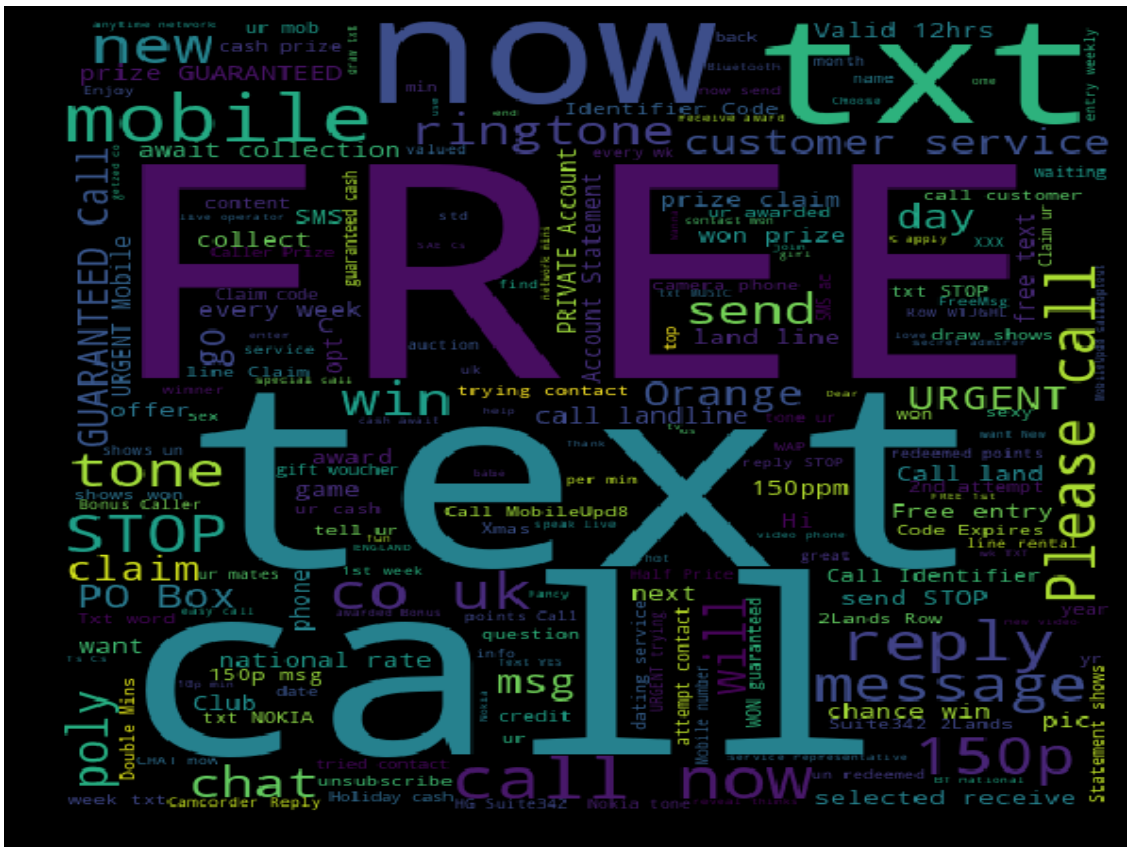
Visualizing number of spam SMS and ham SMS



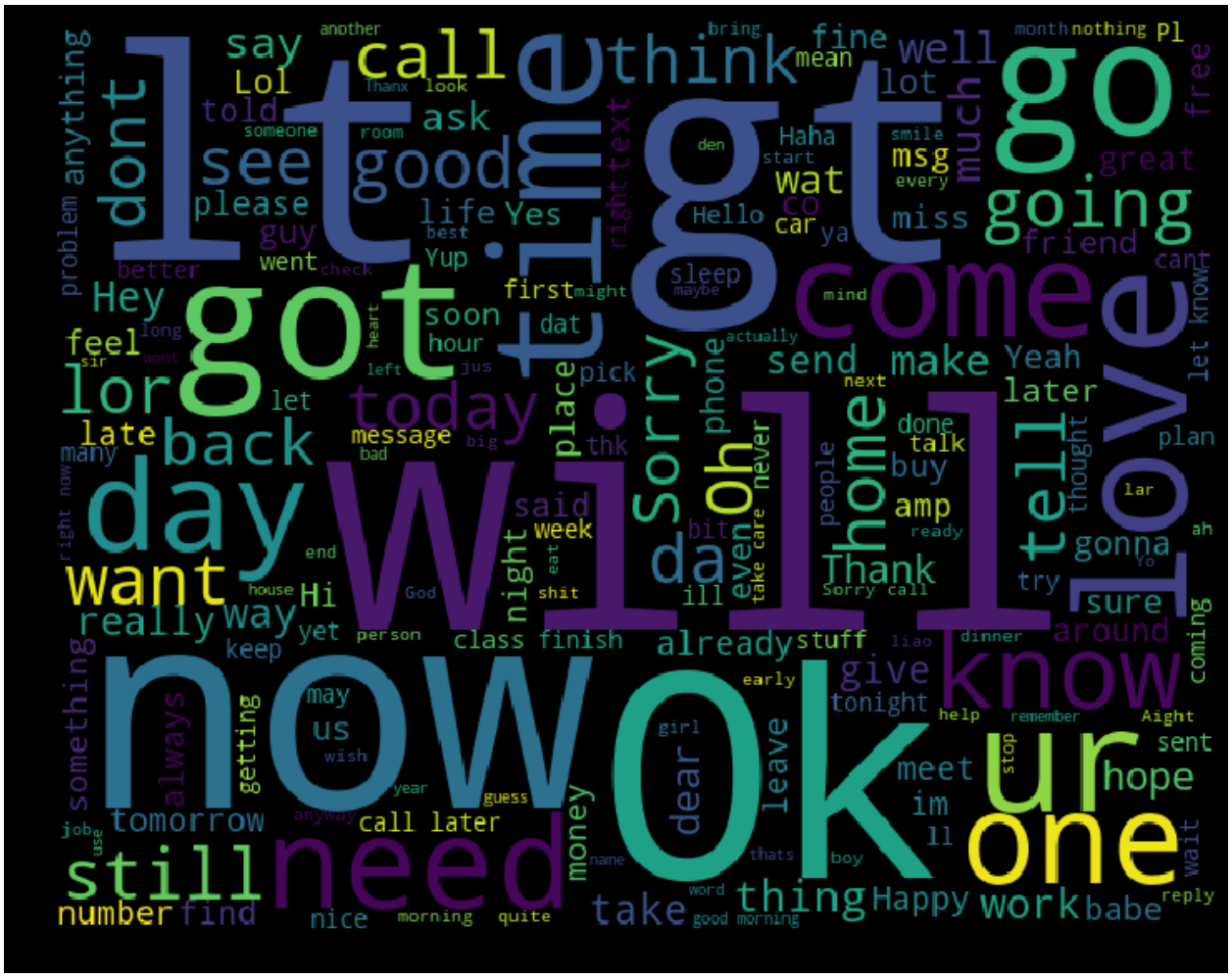
Visualizing descriptions such as count (total no. Of messages), minimum length of message, maximum length of message etc.

Count	5572.000000
Mean	83.607143
Standard Deviation	59.912059
Min	5.000000
Max	913.000000

Visualizing Word-cloud for spam messages:



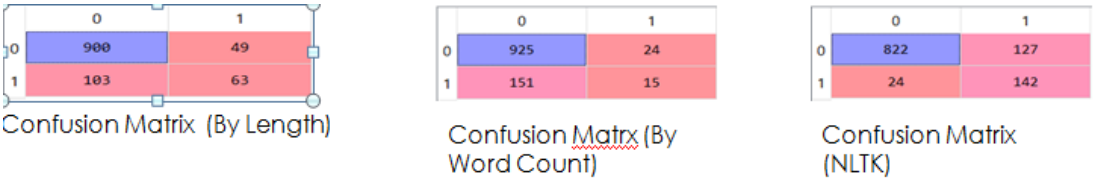
Visualizing Word-cloud for spam messages:



CHAPTER 6

CONCLUSION

It has been a great pleasure for me to work on this exciting and challenging project. This project proved good for me as it provided practical knowledge of not only programming in PYTHON and developing Machine Learning models but also about all handling procedure related with **"FORSK TECHNOLOGIES".** It also provides knowledge about the latest technology used in machine learning technology that will be great demand in future. This will provide better opportunities and guidance in future in developing projects independently.



6.1 Confusion Matrixes

CHAPTER 6

FUTURE SCOPE

- In future we will try to connect it with of inboxes of mobile phones and other messengers present.
- Try to improve the accuracy which is 89.2% till now
- A new dataset will be added to train this model with latest type of spam messages.
- Creating new application related to the project such as android applications etc ,so that we can run this on mobile phones etc also.

REFERENCES

- for dataset

www.kaggle.com

- for deployment and packing on server

www.forsk.inopened.forsk.in