OPTIMIZING SPAM FILTERING WITH MACHINE LEARNING

1. INTRODUCTION

1.1 OVERVIEW

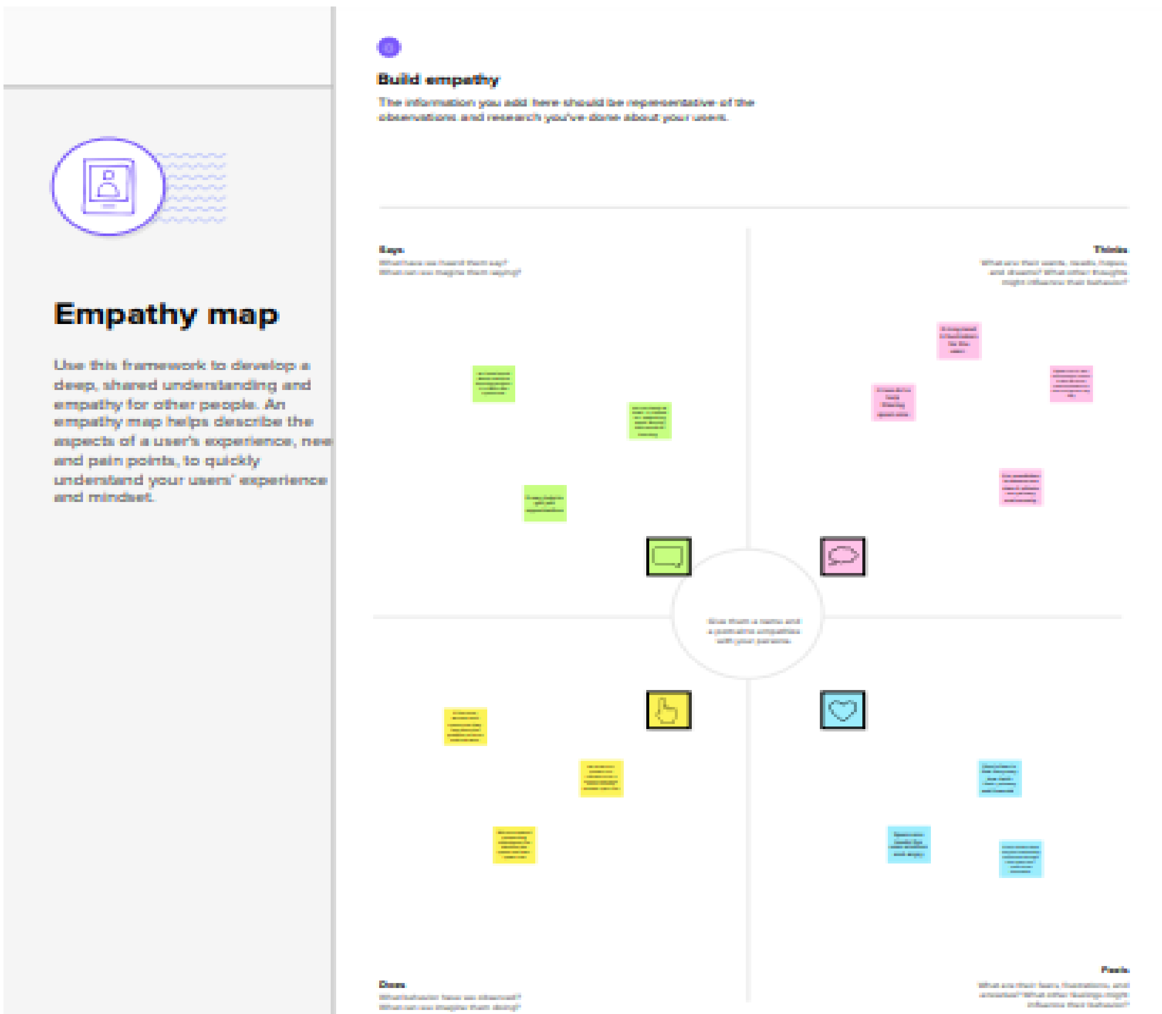
Over recent years, as the popularity of mobile phone devices has increased, Short Message Service (SMS) has grown into a multi-billion dollar industry. At the same time, reduction in the cost of messaging services has resulted in growth in unsolicited commercial advertisements (spams) being sent to mobile phones. Due to Spam, Mobile services providers suffer from some sort of financial problems as well as it reduces calling time for users. Unfortunately, if the user accesses such Spam SMS they may face the problem of virus or malware. When SMS arrives at mobile it will disturb mobile user privacy and concentration. It may lead to frustration for the user. So Spam SMS is one of the major issues in the wireless communication world and it grows day by day.

1.2 PURPOSE

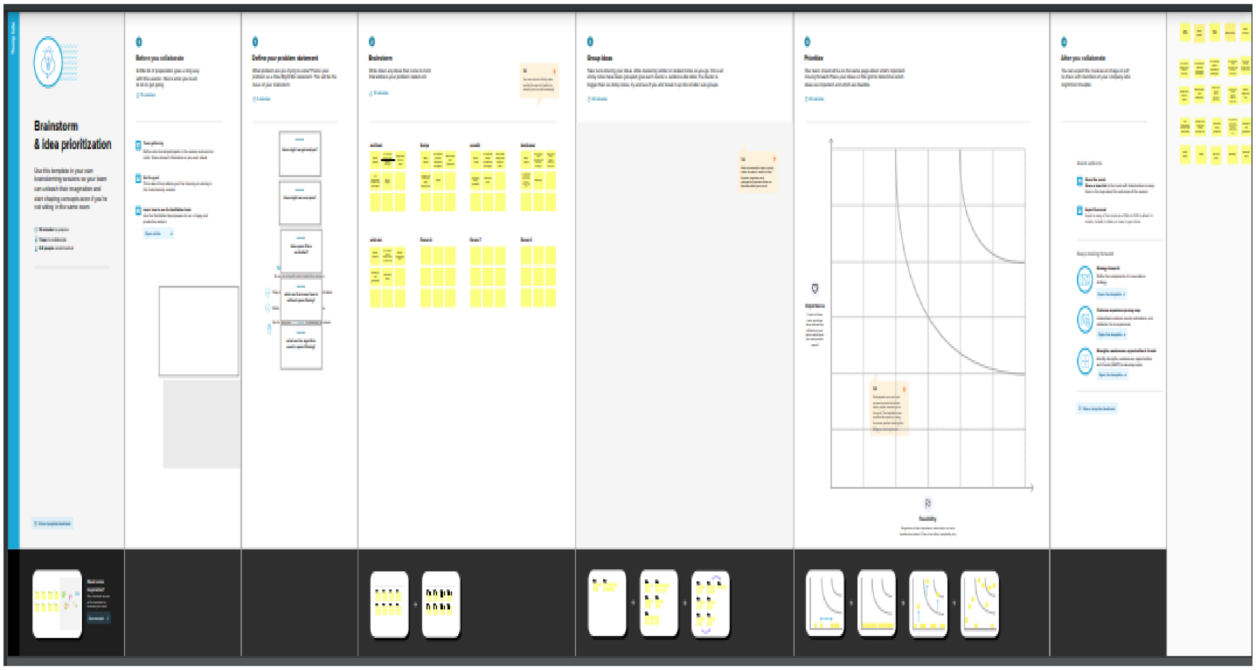
To avoid such Spam SMS people use white and black list of numbers. But this technique is not adequate to completely avoid Spam SMS. To tackle this problem it is needful to use a smarter technique which correctly identifies Spam SMS. Natural language processing technique is useful for Spam SMS identification. It analyses text content and finds patterns which are used to identify Spam and Non-Spam SMS.

2. PROBLEM DEFINITION & DESIGN THINKING

2.1 EMPATHY MAP



2.2 IDEATION & BRAINSTORMING MAP



3. RESULT

	v1	v2	Unnamed: 2	Unnamed: 3	Unnamed: 4
0	ham	Go until jurong point, crazy.. Available only ...	NaN	NaN	NaN
1	ham	Ok lar... Joking wif u oni...	NaN	NaN	NaN
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	NaN	NaN	NaN
3	ham	U dun say so early hor... U c already then say...	NaN	NaN	NaN
4	ham	Nah I don't think he goes to usf, he lives aro...	NaN	NaN	NaN

```
<class 'pandas.core.frame.DataFrame'>
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
```

```
df.isna().sum()
```

```
v1               0
v2               0
Unnamed: 2    5522
Unnamed: 3    5560
Unnamed: 4    5566
dtype: int64
```

	v1	v2
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...

	label	text
5567	spam	This is the 2nd time we have tried 2 contact u...
5568	ham	Will Ì_ b going to esplanade fr home?
5569	ham	Pity, * was in mood for that. So...any other s...
5570	ham	The guy did some bitching but I acted like i'd...
5571	ham	Rofl. Its true to its name

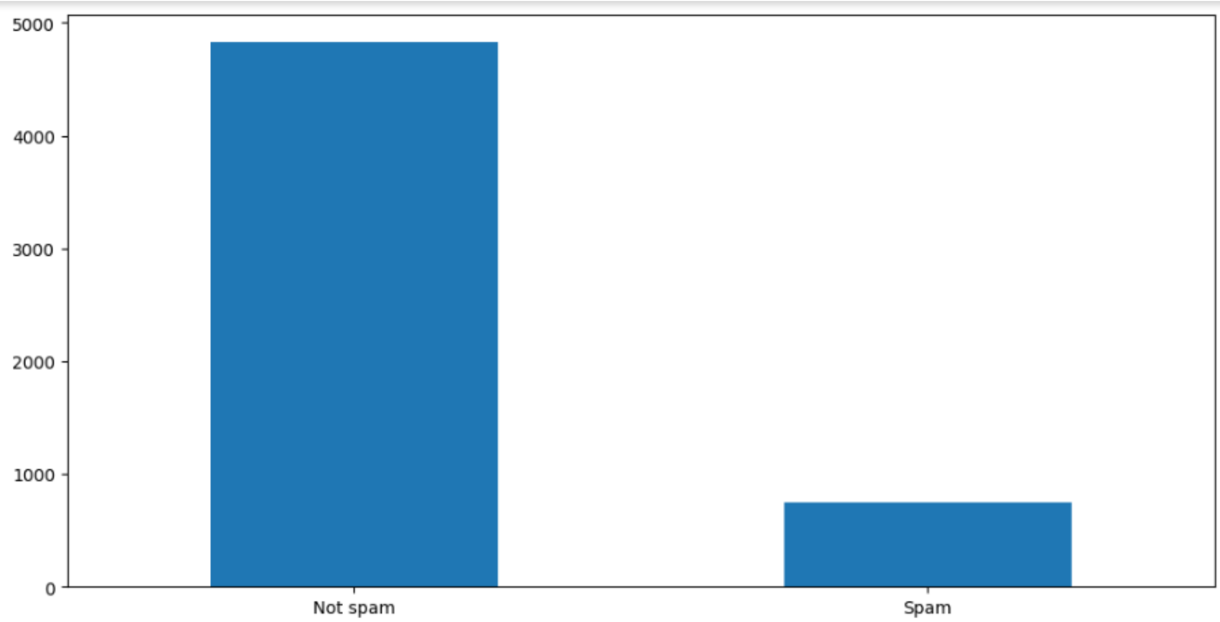
```
Before oversampling, counts of label'1': 581
Before oversampling, counts of label'0': 3876


after oversampling, counts of label '1': 581
after oversampling, counts of label '0': 3876
```

```
['go jurong point crazi avail bugi n great world la e buffet cine got amor wat',
 'ok lar joke wif u oni',
 'free entri 2 wkli comp win fa cup final tkt 21st may 2005 text fa 87121 receiv entri question std txt
rate c appli 08452810075over18',
 'u dun say earli hor u c alreadi say',
 'nah think goe usf live around though',
 'freemsg hey darl 3 week word back like fun still tb ok xxx std chg send 1 50 rcv',
 'even brother like speak treat like aid patent',
 'per request mell mell oru minnaminungint nurungu vettam set callertun caller press 9 copi friend
callertun',
 'winner valu network custom select receivea 900 prize reward claim call 09061701461 claim code kl341
valid 12 hour',
 'mobil 11 month u r entitl updat latest colour mobil camera free call mobil updat co free 08002986030',
 'gonna home soon want talk stuff anymor tonight k cri enough today',
 'six chanc win cash 100 20 000 pound txt csh11 send 87575 cost 150p day 6day 16 tsandc appli repli hl 4
info',
 'urgent 1 week free membership 100 000 prize jackpot txt word claim 81010 c www dbuk net lccltd pobox
4403ldnw1a7rw18',
 'search right word thank breather promis wont take help grant fulfil promis wonder bless time',
 'date sunday',
 'xxxmobilemovieclub use credit click wap link next txt messag click http wap xxxmobilemovieclub com n
qjkgighjjgcbl',
 'oh k watch',
```

	label
count	5572.000000
mean	0.134063
std	0.340751
min	0.000000
25%	0.000000
50%	0.000000
75%	0.000000
max	1.000000

```
df.shape
```

(5572, 2)



▾ DecisionTreeClassifier

DecisionTreeClassifier()

```
▾ RandomForestClassifier
RandomForestClassifier()
```

```
▾ MultinomialNB
MultinomialNB()
```

```
(4457, 7163)
```

```
[[948   1]
 [ 13 153]]
Accuracy score is:- 98.7443946188341
```

```
Epoch 1/10
69/69 [==============================] - 84s 1s/step - loss: 0.1729 - accuracy: 0.9441
Epoch 2/10
69/69 [==============================] - 78s 1s/step - loss: 0.0082 - accuracy: 0.9987
Epoch 3/10
69/69 [==============================] - 78s 1s/step - loss: 0.0016 - accuracy: 0.9998
Epoch 4/10
69/69 [==============================] - 78s 1s/step - loss: 3.2846e-04 - accuracy: 1.0000
Epoch 5/10
69/69 [==============================] - 78s 1s/step - loss: 1.3045e-04 - accuracy: 1.0000
Epoch 6/10
69/69 [==============================] - 78s 1s/step - loss: 8.4569e-05 - accuracy: 1.0000
Epoch 7/10
69/69 [==============================] - 81s 1s/step - loss: 6.1553e-05 - accuracy: 1.0000
Epoch 8/10
69/69 [==============================] - 78s 1s/step - loss: 4.7031e-05 - accuracy: 1.0000
Epoch 9/10
69/69 [==============================] - 78s 1s/step - loss: 3.6900e-05 - accuracy: 1.0000
Epoch 10/10
69/69 [==============================] - 79s 1s/step - loss: 2.9689e-05 - accuracy: 1.0000
```

```
array([[2.2721011e-12],
       [1.4128274e-05],
       [2.6681055e-12],
       ...,
       [3.3437507e-07],
       [1.1418366e-13],
       [6.5217437e-10]], dtype=float32)
```

```
array([0, 0, 0, ..., 0, 0, 0], dtype=uint8)
```

```
array([[2.2721011e-12],
       [1.4128274e-05],
       [2.6681055e-12],
       ...,
       [3.3437507e-07],
       [1.1418366e-13],
       [6.5217437e-10]], dtype=float32)
```

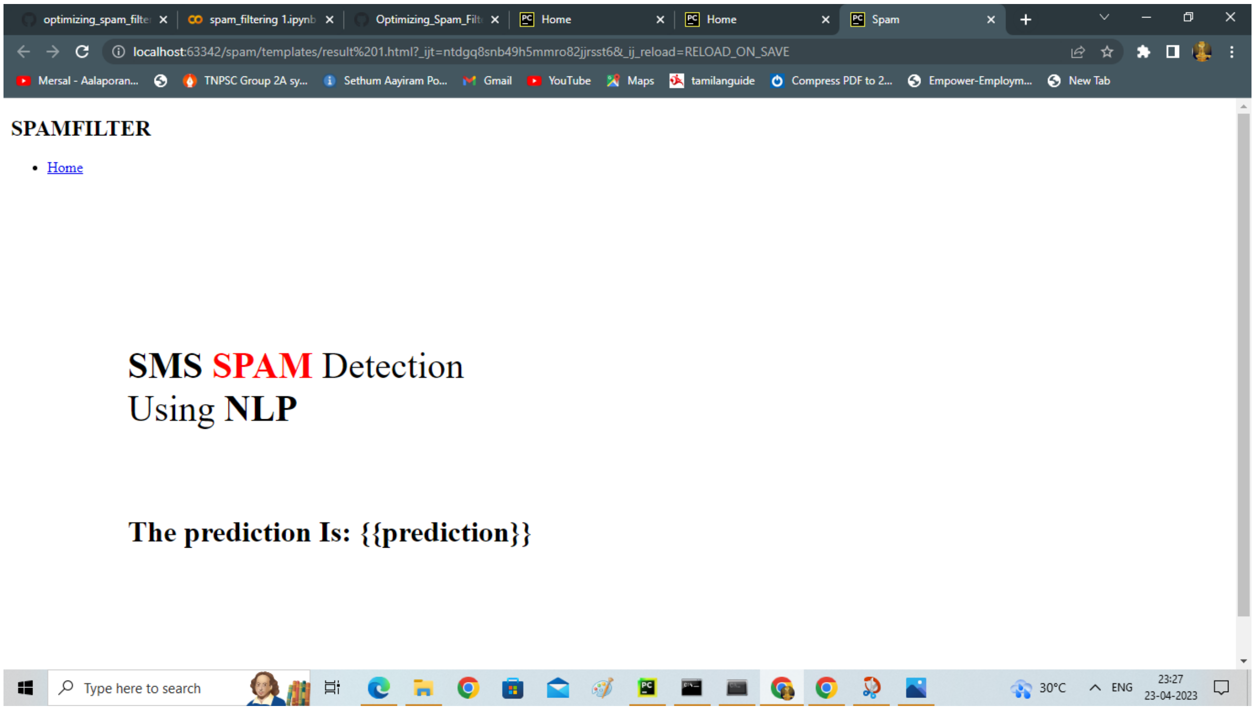
```
Enter new review...how are you
1/1 [==============================] - 0s 42ms/step
NOT SPAM
```

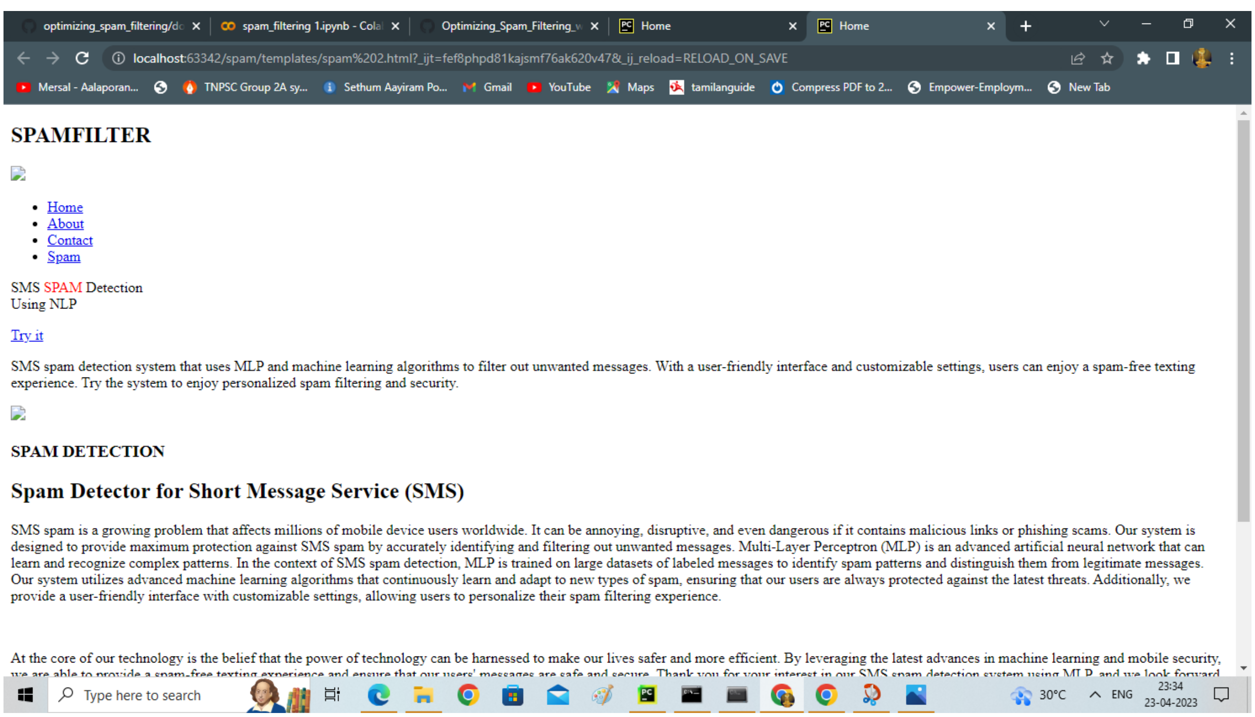
```
0.9730941704035875
```

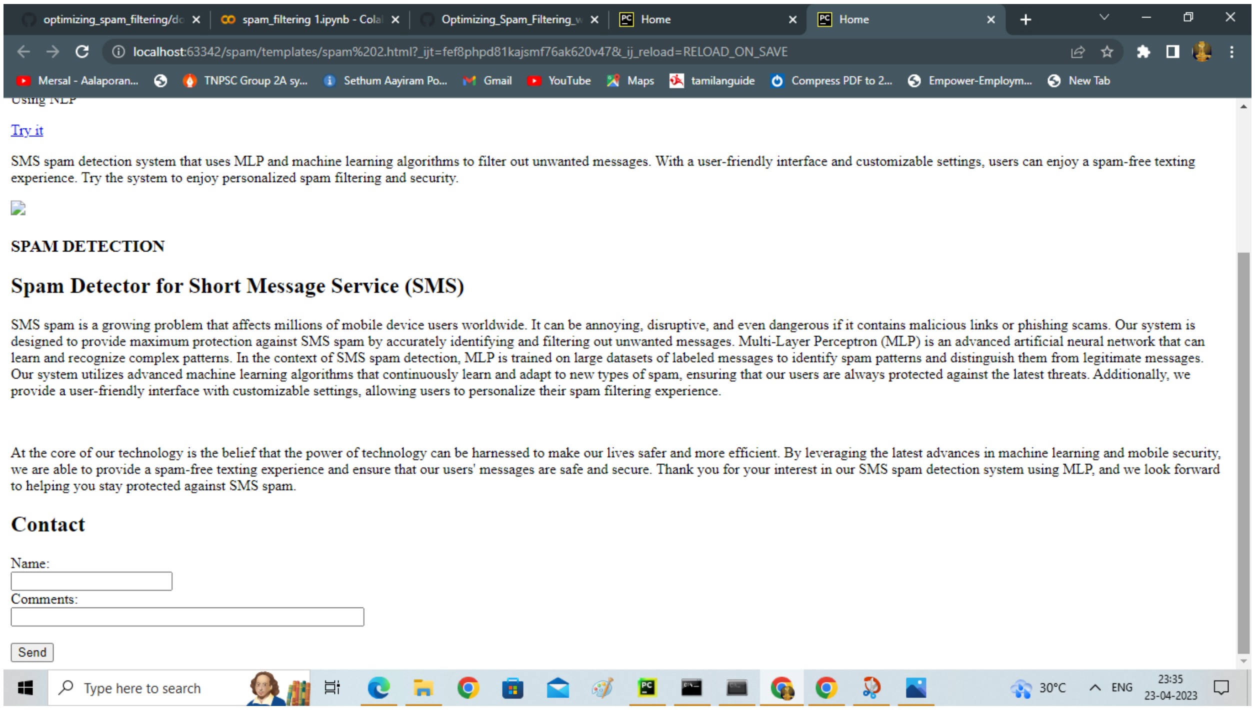
	Model	Test Score
0	MultinomialNM	0.987444
3	Decision Tree	0.976682
2	SVM-sigmoid	0.975785
1	SVM-rbf	0.973094

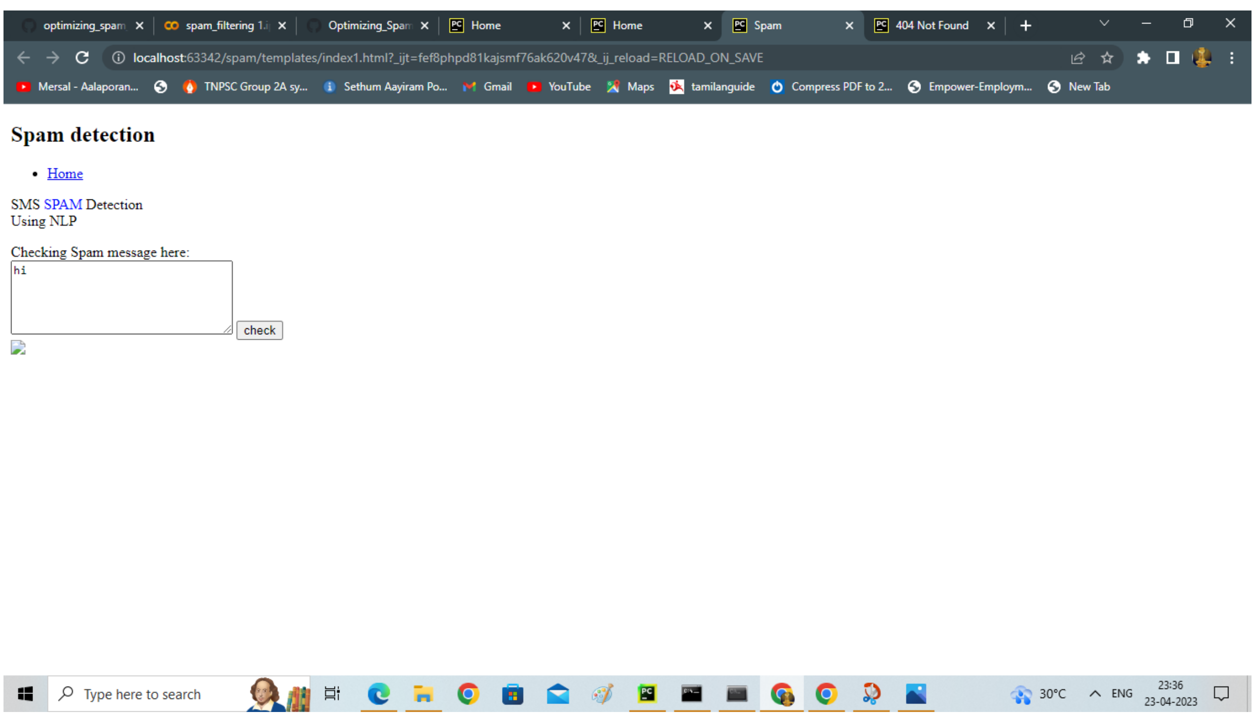
```
] model.save('spam.h5')
```

```
2023-04-23 07:22:41.128787: W tensorflow/tsl/framework/cpu_allocato
2023-04-23 07:22:51.492462: W tensorflow/tsl/framework/cpu_allocato
2023-04-23 07:23:45.694278: W tensorflow/tsl/framework/cpu_allocato
2023-04-23 07:23:45.959088: W tensorflow/tsl/framework/cpu_allocato
 * Serving Flask app 'main'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a productio
```


localhost:63342/spam/templates/result%201.html?_ijt=ntdgq8snb49h5mmro82jjrsst6&_ij_reload=RELOAD_ON_SAVE
SPAMFILTER
Home
SMS SPAM Detection
Using NLP
The prediction Is: {{prediction}}
Type here to search
30°C
ENG
23:27
23-04-2023


localhost:63342/spam/templates/spam%202.html?_ijt=fef8phpd81kajsmf76ak620v47&_ij_reload=RELOAD_ON_SAVE
SPAMFILTER
Home
About
Contact
Spam
SMS SPAM Detection
Using NLP
Try it
SMS spam detection system that uses MLP and machine learning algorithms to filter out unwanted messages. With a user-friendly interface and customizable settings, users can enjoy a spam-free texting experience. Try the system to enjoy personalized spam filtering and security.
SPAM DETECTION
Spam Detector for Short Message Service (SMS)
SMS spam is a growing problem that affects millions of mobile device users worldwide. It can be annoying, disruptive, and even dangerous if it contains malicious links or phishing scams. Our system is designed to provide maximum protection against SMS spam by accurately identifying and filtering out unwanted messages. Multi-Layer Perceptron (MLP) is an advanced artificial neural network that can learn and recognize complex patterns. In the context of SMS spam detection, MLP is trained on large datasets of labeled messages to identify spam patterns and distinguish them from legitimate messages. Our system utilizes advanced machine learning algorithms that continuously learn and adapt to new types of spam, ensuring that our users are always protected against the latest threats. Additionally, we provide a user-friendly interface with customizable settings, allowing users to personalize their spam filtering experience.
At the core of our technology is the belief that the power of technology can be harnessed to make our lives safer and more efficient. By leveraging the latest advances in machine learning and mobile security,
Type here to search
30°C
ENG
23:34
23-04-2023


localhost:63342/spam/templates/spam%202.html?_ijt=fef8phpd81kajsmf76ak620v47&_ij_reload=RELOAD_ON_SAVE
Using NLP
Try it
SMS spam detection system that uses MLP and machine learning algorithms to filter out unwanted messages. With a user-friendly interface and customizable settings, users can enjoy a spam-free texting experience. Try the system to enjoy personalized spam filtering and security.
SPAM DETECTION
Spam Detector for Short Message Service (SMS)
SMS spam is a growing problem that affects millions of mobile device users worldwide. It can be annoying, disruptive, and even dangerous if it contains malicious links or phishing scams. Our system is designed to provide maximum protection against SMS spam by accurately identifying and filtering out unwanted messages. Multi-Layer Perceptron (MLP) is an advanced artificial neural network that can learn and recognize complex patterns. In the context of SMS spam detection, MLP is trained on large datasets of labeled messages to identify spam patterns and distinguish them from legitimate messages. Our system utilizes advanced machine learning algorithms that continuously learn and adapt to new types of spam, ensuring that our users are always protected against the latest threats. Additionally, we provide a user-friendly interface with customizable settings, allowing users to personalize their spam filtering experience.
At the core of our technology is the belief that the power of technology can be harnessed to make our lives safer and more efficient. By leveraging the latest advances in machine learning and mobile security, we are able to provide a spam-free texting experience and ensure that our users' messages are safe and secure. Thank you for your interest in our SMS spam detection system using MLP, and we look forward to helping you stay protected against SMS spam.
Contact
Name:
Comments:
Send
Type here to search
30°C ENG 23:35 23-04-2023


localhost:63342/spam/templates/index1.html?_ijt=fef8phpd81kajsmf76ak620v47&_ij_reload=RELOAD_ON_SAVE
Spam detection
Home
SMS SPAM Detection
Using NLP
Checking Spam message here:
hi
check
Type here to search
30°C ENG 23:36 23-04-2023

4. ADVANTAGES & DISADVANTAGES

ADVANTAGES:

- The proposed model is trained well to identify the SMS category in terms of Ham or Spam with TF-IDF features and oversampling technique. The performance of the proposed approach was also evaluated on the spam email dataset with significant 99% accuracy.
- It is very effective and is also adaptive, so hard to fool.
- Based on text classification methods: Decision tree model, Random forest model, Naïve Bayes model, ANN model.
- Phenomenally accurate.
- Learns new spammer tactics automatically.
- Adapt to changing spam.

DISADVANTAGES:

- Need lots of training data.
- Spammers are learning too-Images, synonyms, misspellings,…
- Hard to get good email corpora.
- Need huge attributes.
- Functions best with individual user settings.
- Accuracy dramatically decreases when deployed as a generic gateway solution.
- Requires more processing power.

5. APPLICATIONS

Since having a good term representation is one of the most important parts for getting a good classifier, we have to face the fact that SMS messages have not the same structure and characteristics than email messages. We have described techniques used to filter spam email messages, but we cannot state they can be also effective filtering SMS.

SMS are usually shorter than email messages. Only 160 characters are allowed in a standard SMS text, and that could be a problem because using fewer words means less information to work with.

Also, due to the above constraint, people tend to use acronyms when writing SMS. Moreover, the abbreviations used by SMS users are not standard for a language, but they depend on the users communities. Such language variability provides more terms or features, and a more sparse representation. We have to test if the state of the art methods used to extract terms from email messages are also suitable for SMS texts.

- Spam Titan
- Maliwasher
- SpamSieve
- Comodo Dome Antispam
- Spamfighter
- MX Guarddog

6. CONCLUSION

SMS Spam identification is one of the important task in present world, which is wasting user’s valuable time as well as money. Present algorithm tackles this issues.

Present Work is useful to identify Spam SMS from SMS dataset. Experimental work shows that 98.12% SMS are identified correctly as Spam SMS’s from the dataset.

It also checks algorithm errors by most important error checking technique MAE and RMSE. MAE of current algorithm is 0.091 and RMSE is 0.3 which is very less. Therefore present study correctly identifies Spam SMS’s as compared to other algorithms. There is more scope to increase accuracy in identifying Spam SMS. The merit of our approach which lies in the various machine recognizable statistics derived from the skeleton of the document (HTML tags).

7. FUTURE SCOPE

The proposed approach is helpful because it can automatically detect SMS categories. So, there is no need for human interaction for categorical purposes, and the proposed model will automatically detect the SMS category.

This research can be further explored by hybrid machine learning techniques to enhance the accuracy of results, which will be beneficial in categorizing SMS.

8. APPENDIX

SOURCE CODE:

```
import numpy as np #scientific computation
import pandas as pd #loading dataset file
import matplotlib.pyplot as plt #visualization
from sklearn.model_selection import train_test_split #train dataset
import nltk #preprocessing
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer

# read the dataset
df = pd.read_csv("/content/spam.csv",encoding="latin")
df.head()


#Give consise  summary of the dataframe
df.info()

#return the sum of all no values
df.isna().sum()

#rename the dataset
df.rename({"v1":"label","v2":"text"},inplace=True,axis=1)
df.tail()

#HANDILING CATEGROICAL VALUES
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
df['label'] = le.fit_transform(df['label'])

#CLEANING THE TEXT DATA
nltk.download("stopwords")

import nltk
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer

import re
corpus = []
length = len(df)
```

```
for i in range(0,length):
  text = re.sub("^a-Za-Z0-9]"," " ,df["text"][i])
  text = text.lower()
  text = text.split()
  pe = PorterStemmer()
  stopword = stopwords.words("english")
  text = [pe.stem(word) for word in text if not word in set(stopword)]
  text = " ".join(text)
  corpus.append(text)

corpus

# splting datq into train and validatiob sets using train_test_split
from sklearn.feature_extraction.text import CountVectorizer
cv = CountVectorizer(max_features=35000)
x =cv.fit_transform(corpus).toarray()

y = pd.get_dummies(df['label'])
y = y.iloc[:, 1].values

import pickle
pickle.dump(cv, open('cv1.pkl','wb'))

from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.20,
random_state=0)

print("Before OverSampling, counts of label '1': {}".format(sum(y_train ==
 1)))
print("Before OverSampling, counts of label '0': {}  \n".format(sum(y_trai
n == 0)))

from imblearn.over_sampling import SMOTE
sm = SMOTE(random_state = 2)
x_train_res, y_train_res = sm.fit_resample(x_train, y_train.ravel())

print('After OverSampling, the shape of train_x: {}'.format(x_train_res.sh
ape))
print('After OverSampling, the shape of train_y: {} \n'.format(y_train_res
.shape))

print("After OverSampling, counts of label '1': {}".format(sum(y_train ==
1)))
print("After OverSampling, counts of label '0': {}".format(sum(y_train ==
0)))
```

```
df.describe()

df.shape

df["label"].value_counts().plot(kind="bar",figsize=(12,6))
plt.xticks(np.arange(2),  ('Non spam', 'spam'),rot

from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.20,
random_state=0)


from sklearn.tree import DecisionTreeClassifier



model = DecisionTreeClassifier()
model.fit(x_train_res, y_train_res)

from sklearn.ensemble import RandomForestClassifier

model = RandomForestClassifier()
model.fit(x_train_res, y_train_res)

from sklearn.naive_bayes import MultinomialNB
model = MultinomialNB()

model.fit(x_train_res, y_train_res

model.fit(x_train_res, y_train_res)


from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense

model =  Sequential()


x_train.shape

model.add(Dense(units =
x_train_res.shape[1],activation="relu",kernel_initializer="random_uniform"
))
```

```
model.add(Dense(units
=100,activation="relu",kernel_initializer="random_uniform"))


model.add(Dense(units=1,activation="sigmoid"))

model.compile(optimizer="adam",loss="binary_crossentropy",metrics=['accura
cy'])


generator =
model.fit(x_train_res,y_train_res,epochs=10,steps_per_epoch=len(x_train_re
s)//64)

generator =
model.fit(x_train_res,y_train_res,epochs=10,steps_per_epoch=len(x_train_re
s)//64)


y_pred=model.predict(x_test)
y_pred


y_pred1=model.predict(x_train)
y_pred1

y_pred1 = np.where(y_pred>0.5,1,0)

y_pr = np.where(y_pred>0.5,1,0)

y_test

y_pred = np.where(y_pred>0.5,1,0)

y_pred1 = np.where(y_pred>0.5,1,0)

from sklearn.metrics import confusion_matrix,accuracy_score
cm = confusion_matrix(y_test, y_pr)
score = accuracy_score(y_test,y_pr)
print(cm)
print('Accuracy Score Is:- ' ,score*100)


def new_review(new_review):
```

```
    new_review = new_review
    new_review = re.sub('[^a-zA-Z]', ' ', new_review)
    new_review = new_review.lower()
    new_review = new_review.split()
    ps = PorterStemmer()
    all_stopwords = stopwords.words('english')
    all_stopwords.remove('not')
    new_review = [ps.stem(word) for word in new_review if not word in
set(all_stopwords)]
    new_review = ' '.join(new_review)
    new_corpus = [new_review]
    new_X_test = cv.transform(new_corpus).toarray()
    new_y_pred = model.predict(new_X_test)
    print(new_y_pred)
    new_X_pred = np.where(new_y_pred>0.5,1,0)
    return new_review
new_review = new_review(str(input("Enter new review...")))

from sklearn.metrics import confusion_matrix,accuracy_score
cm=confusion_matrix(y_test,y_pr)
score = accuracy_score(y_test,y_pr)
print(cm)
print('Accuracy Score Is Naive Bayes:- ' ,score*100)


#COMPARE THE MODEL
from sklearn.metrics import confusion_matrix,accuracy_score
cm=confusion_matrix(y_test,y_pred)
score = accuracy_score(y_test,y_pred)
print(cm)
print('Accuracy Score Is Naive Bayes:- ' ,score*100)


from sklearn.metrics import confusion_matrix,accuracy_score
cm1=confusion_matrix(y_test,y_pred1)
score = accuracy_score(y_test,y_pred1)
print(cm1)
print('Accuracy Score Is Naive Bayes:- ' ,score*100)

model.save('spam.h5')

from sklearn.svm import SVC

svm1=SVC(kernel='rbf')
```

```
svm1.fit(x_train_res, y_train_res)


y_pred4=svm1.predict(x_test)
from sklearn.metrics import accuracy_score
svm_rbf=accuracy_score(y_test,y_pred4)
svm_rbf

svm2=SVC(kernel='sigmoid')
svm2.fit(x_train, y_train)


y_pred5=svm2.predict(x_test)
from sklearn.metrics import accuracy_score
svm_sig=accuracy_score(y_test,y_pred5)
svm_sig

from sklearn.tree import DecisionTreeClassifier
model = DecisionTreeClassifier()
model.fit(x_train, y_train)

y_pred6=model.predict(x_test)
from sklearn.metrics import accuracy_score
dec_tree=accuracy_score(y_test,y_pred6)
dec_tree

!pip install nbconvert

from google.colab import drive
drive.mount('/content/drive')

! pwd
```