BUILDING A SMARTER AI POWER SPAM CLASSIFIER

#### Phase 5 Project documentation

**and submission**

# Content for project phase5

# Problem statement

1. Design thinking process
2. phases of development part(1)
3. dataset load
4. Data preprocessing Iii)Feature extraction
5. phases of development part(2)
6. Model selection
7. Model Training
8. Model Evaluation Iv)Model prediction
9. Innovative technique used
10. Bert language for feature extraction
11. Heatmap
12. Word cloud to see spam and ham messages iv)Cofusion matrix

**Data source:**

Data link:( https://[www.kaggle.com/datasets/uciml/sms-spam-collection-dataset)](http://www.kaggle.com/datasets/uciml/sms-spam-collection-dataset))

v1 v2

ham Go until jurong point, crazy.. Available only in bugis n great world la e buffet...

Cine there got amore wat...

ham Ok lar... Joking wif u oni...

spam Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text FA to 87121 to receive entry question(std txt rate)T&C's apply 08452810075over18's

ham U dun say so early hor... U c already then say...

ham Nah I don't think he goes to usf, he lives around here though

spam FreeMsg Hey there darling it's been 3 week's now and no word back! I'd like som e fun you up for it still? Tb ok! XxX std chgs to send, 螢 1.50 to rcv

ham Even my brother is not like to speak with me. They treat me like aids patent.

ham As per your request 'Melle Melle (Oru Minnaminunginte Nurungu Vettam)' has been set as your callertune for all Callers. Press \*9 to copy your friends Callertune

WINNER!! As a valued network customer you have been selected to receivea 螢

spam

900 prize reward! To claim call 09061701461. Claim code KL341. Valid 12 hours only.

spam Had your mobile 11 months or more? U R entitled to Update to the latest colour mobiles with camera for Free! Call The Mobile Update Co FREE on 08002986030

ham I'm gonna be home soon and i don't want to talk about this stuff anymore tonigh t, k? I've cried enough today.

SIX chances to win CASH! From 100 to 20,000 pounds txt> CSH11 and send to

spam

spam

ham

875

1. Cost 150p/day, 6days, 16+ TsandCs apply Reply HL 4 info

URGENT! You have won a 1 week FREE membership in our 螢 100,000 Prize Jackp ot! Txt the word: CLAIM to No: 81010 T&C [www.dbuk.net](http://www.dbuk.net/) LCCLTD POBOX 4403LD NW1A7RW18

I've been searching for the right words to thank you for this breather. I promise i wont take your help for granted and will fulfil my promise. You have been wonde rful and a blessing at all times.

ham I HAVE A DATE ON SUNDAY WITH WILL!!

spam XXXMobileMovieClub: To use your credit, click the WAP link in the next txt messaGe or click here>> [http://wap.](http://wap/) xxxmobilemovieclub.com?n=QJKGIGHJJGCBL

ham Oh k...i'm watching here:)

ham Eh u remember how 2 spell his name... Yes i did. He v naughty make until i v wet.

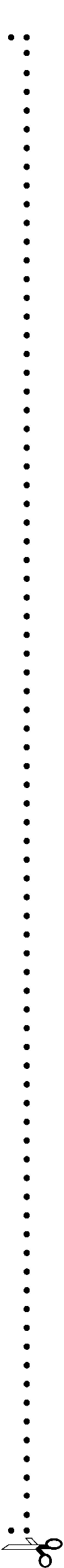
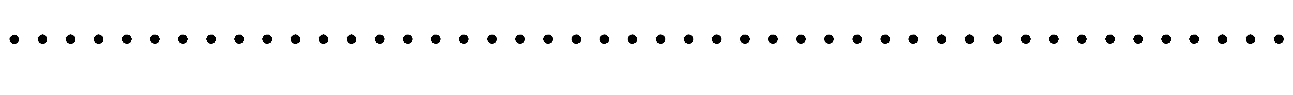
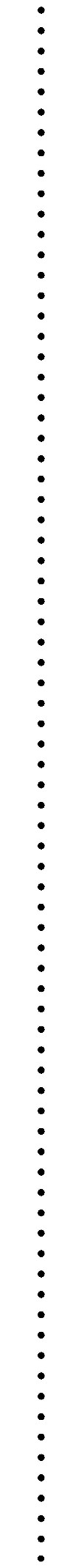
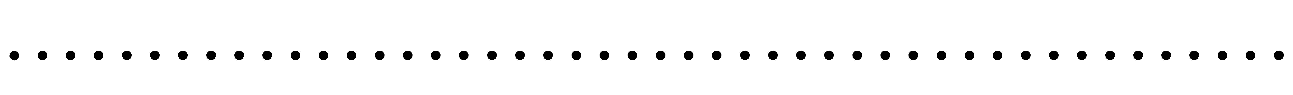
ham Fine if that 袗s the way u feel. That 袗s the way its gota b

England v Macedonia - dont miss the goals/team news. Txt ur national team to

spam

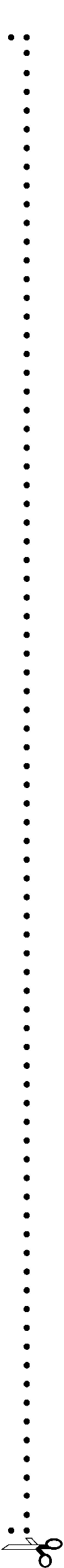
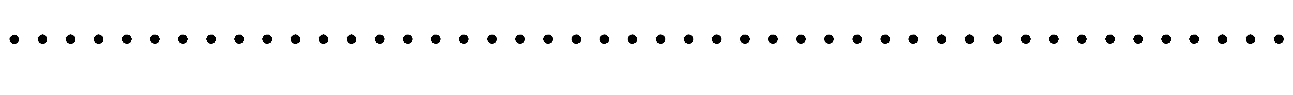
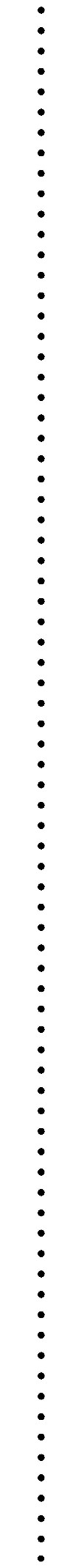
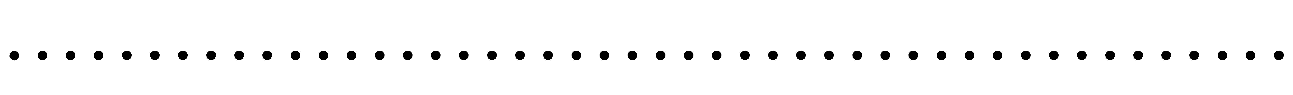
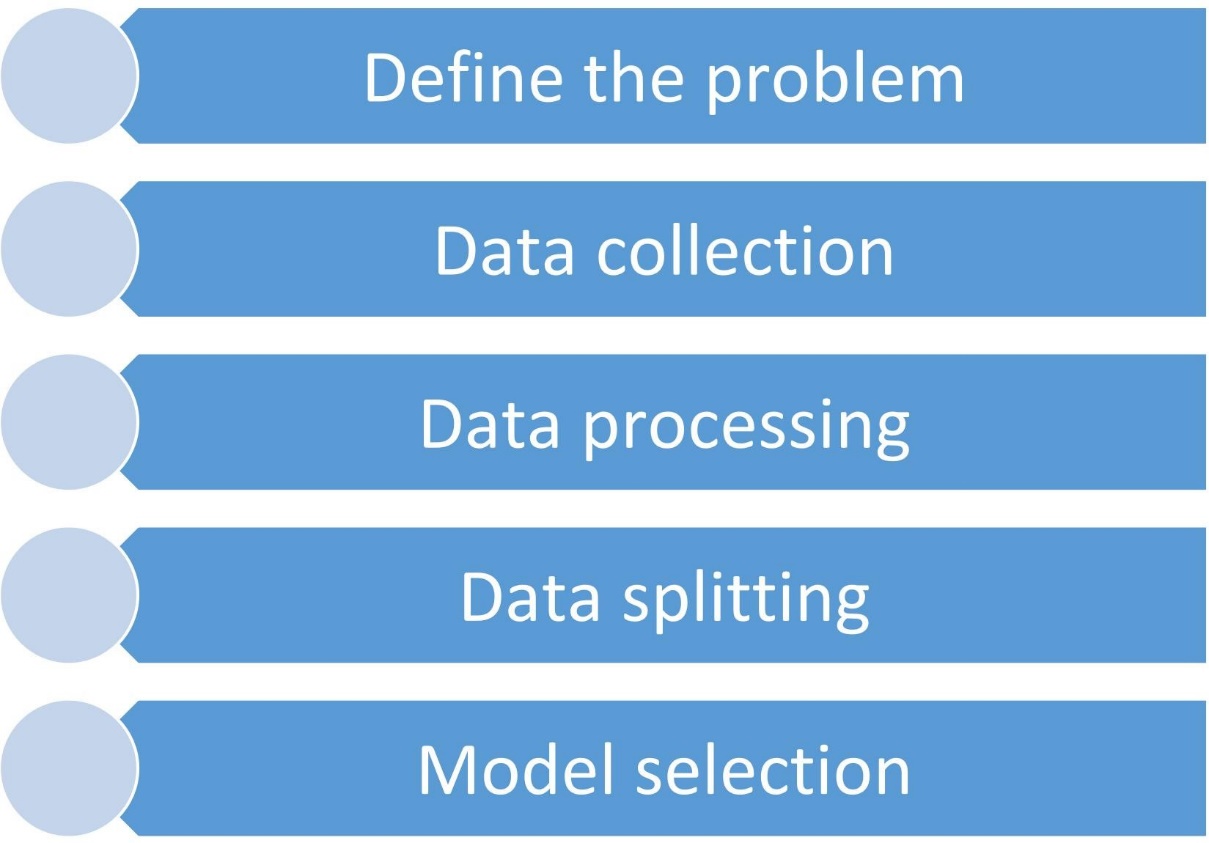
87077 eg ENGLAND to 87077 Try:WALES, SCOTLAND 4txt/ﾌｼ1.20 POBOXox36504W45WQ 16+

ham Is that seriously how you spell his name?

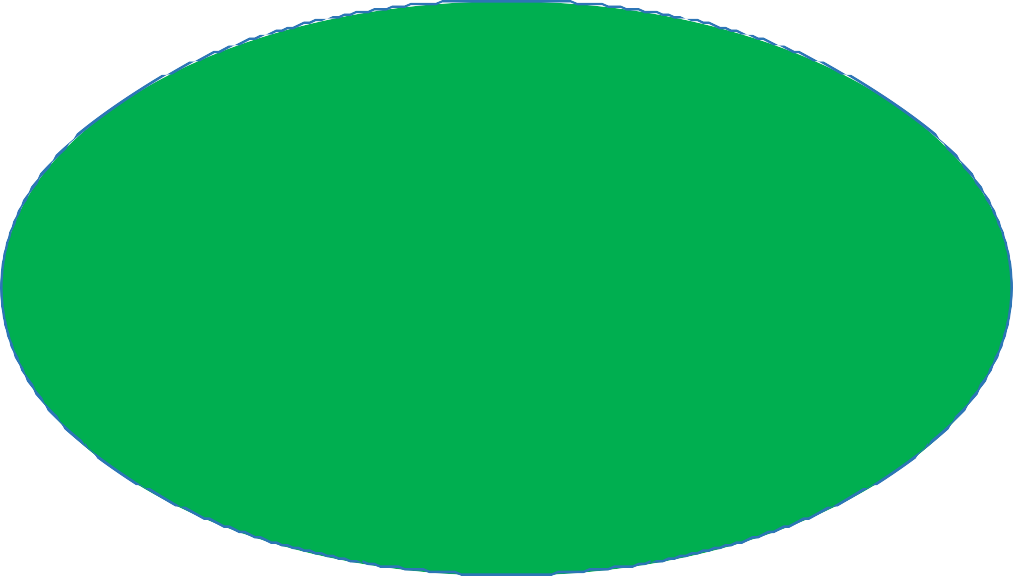


ham I 課 going to try for 2 months ha ha only joking ham So ﾌ\_ pay first lar... Then when is da stock comin...

ham Aft i finish my lunch then i go str down lor. Ard 3 smth lor. U finish ur lunch already?

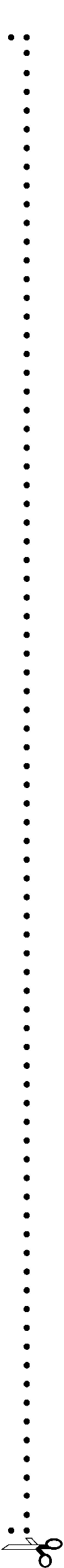
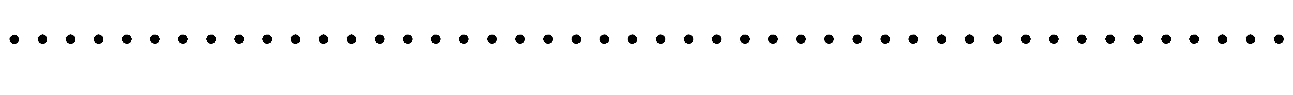
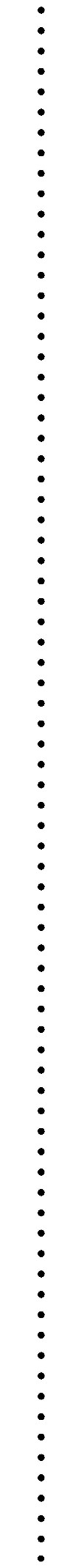
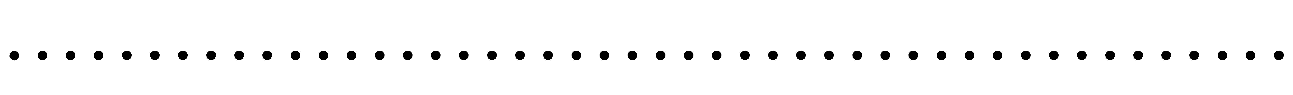
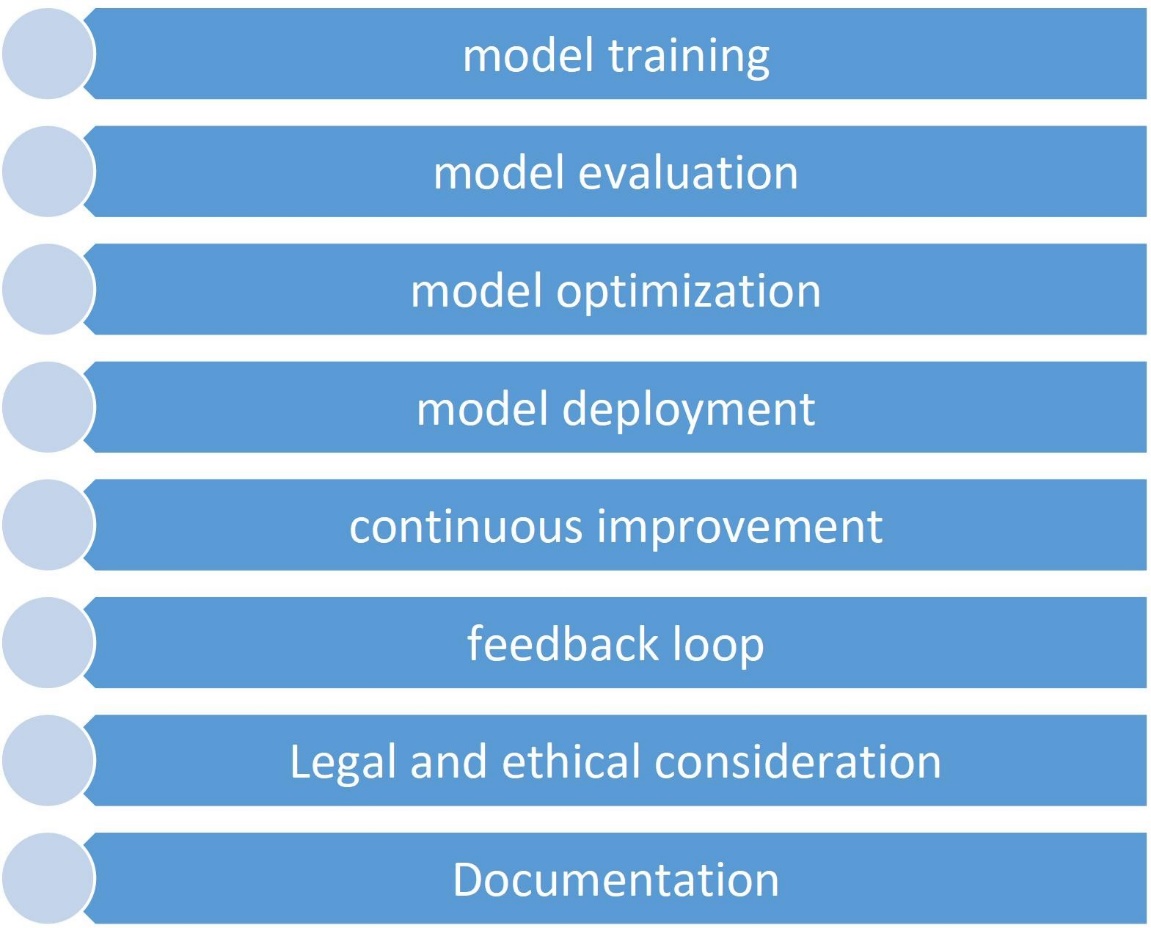


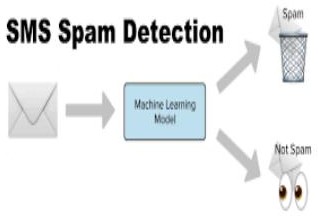
* 1. PROBLEM STATEMENT



A "Smarter AI Power Spam Classifier" refers to a sophisticated artificial intelligence system designed to effectively and accurately identify and filter out spam or

2.DESIGN THINKING







# 3. Data processing steps:

##### Text Cleaning:

Clean and preprocess the text data. This includes: Removing special characters, punctuation, and numbers.

Lowercasing all text.

Tokenization: Splitting text into words or tokens.

Removing stop words (common words like "the," "and," "is" that don't carry much meaning).

##### Feature Extraction:

Convert text data into numerical features that machine learning algorithms can understand.

Common techniques include TF-IDF (Term Frequency-Inverse Document Frequency

) and word embeddings like Word2Vec or GloVe.

##### Data Split:

Split your dataset into two parts:

a training set and a testing set.

The training set will be used to train your model, while the testing set will be used to evaluate its performance.

##### Select a Machine Learning Algorithm:

Choose a suitable machine learning algorithm for text classification.

Common choices include: Naive Bayes

Support Vector Machines (SVM) Logistic Regression

Decision Trees

Neural Networks (e.g., LSTM, CNN)

##### Training the Model

Train your selected model using the

training data. During training,



the model learns to recognize patterns and features in the

data that distinguish

spam from non-spam messages.

##### Evaluate Model Performance:

Use the testing dataset to evaluate the model's performance. Common evaluation metrics for spam classification include accuracy, precision, recall, F1-**score, a**nd ROC AUC.

##### Model optimization:

Hyperparameter Tuning:

Experiment with different hyperparameters of your chosen algorithm

to optimize performance. This may involve grid search or random search. Feature Engineering:

Experiment with different text features and representations to improve model accuracy.

##### Model Deployment:

Once you are satisfied with

your model's performance, deploy it in a production environment.

This may involve integrating it into an application or system

that can classify messages in real-time

**Continuous improvement** in the context of spam classification refers to the ongoing process of enhancing the performance and effectiveness of a spam classifierover time

**Collecting feedback** from users or system administrators who interact with the spam classifier. This feedback can help identify false positives and false negatives, allowing for model adjustments and

training data improvements.



**Legal and ethical considerations**: in AI-powered spam detection refer to the principles, rules, and guidelines that govern the development, deployment, and use of spam classification systems to ensure they align

with legal frameworks and ethical standards. These considerations are crucial to safeguard individuals' rights, protect privacy, and prevent potential misuse of AI technology. Here are definitions for legal and ethical considerations in this context:

**Program:**

##### Import the required packages

import numpy as np import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.feature\_extraction.text import TfidfVectorizer from sklearn.linear\_model import LogisticRegression from sklearn.metrics import accuracy\_score

1. **Loading the dataset** raw\_spam=pd.read\_csv('/content/spam.csv',encoding='latin-1') print(raw\_spam)

##### Output:

v1

v2 Unnamed: 2 \ 0

ham Go until jurong point, crazy.. Available only ... NaN

1

ham

Ok lar... Joking wif u oni... NaN

2

spam Free entry in 2 a wkly comp to win FA Cup fina... NaN

3

ham U dun say so early hor... U c already then say... NaN



4

ham Nah I don't think he goes to usf, he lives aro... NaN

... ...

...

...

5567 spam This is the 2nd time we have tried 2 contact u... NaN

5568

ham

Will Ì\_ b going to esplanade fr home? NaN

5569

ham Pity, \* was in mood for that. So...any other s... NaN

5570

ham The guy did some bitching but I acted like i'd... NaN

5571 ham

Rofl. Its true to its name NaN

Unnamed: 3 Unnamed: 4

0

NaN NaN 1

NaN NaN 2

NaN NaN 3

NaN NaN 4



NaN NaN

...

...

... 5567

NaN NaN 5568

NaN NaN 5569

NaN NaN 5570

NaN NaN 5571

NaN NaN

[5572 rows x 5 columns] error

0scompleted at 12:43 PM**3.Removing the unwanted colomns** raw\_spam.rename(columns = {'v1':'class\_label', 'v2':'message'}, inplace = True)

raw\_spam.drop(['Unnamed: 2', 'Unnamed: 3', 'Unnamed: 4'], axis = 1, inplace

= True) raw\_spam[1990:2000] **Output:**

##### class\_label message 1990

ham HI DARLIN IVE JUST GOT BACK AND I HAD A REALLY...



##### 1991

ham

No other Valentines huh? The proof is on your ...

##### 1992

spam

Free tones Hope you enjoyed your new content. ...

##### 1993

ham

Eh den sat u book e kb liao huh...

##### 1994

ham

Have you been practising your curtsey?

##### 1995

ham

Shall i come to get pickle

##### 1996

ham

Lol boo I was hoping for a laugh

##### 1997

ham

\YEH I AM DEF UP4 SOMETHING SAT

##### 1998

ham

Well, I have to leave for my class babe ... Yo...

##### 1999

ham

LMAO where's your fish memory when I need it? **4.Exploring the dataset:** raw\_spam['class\_label'].value\_counts()

##### Output:

ham 4825

spam 747

Name: class\_label, dtype: int64

##### Print spam messages



raw\_spam = raw\_spam[raw\_spam.class\_label=='spam'] raw\_spam

##### Output:

class\_label message 2

spam

Free entry in 2 a wkly comp to win FA Cup fina… 5

spam

FreeMsg Hey there darling it's been 3 week's n... 8

spam WINNER!! As a valued network customer you have… 9

spam

Had your mobile 11 months or more? U R entitle… 11

spam SIX chances to win CASH! From 100 to 20,000 po 5537 spam

Want explicit SEX in 30 secs? Ring 02073162414...

5540

spam

ASKED 3MOBILE IF 0870 CHATLINES INCLU IN FREE ...

5547 spam

Had your contract mobile 11 Mnths? Latest Moto...

5566 spam

REMINDER FROM O2: To get 2.50 pounds free call… 5567 spam

This is the 2nd time we have tried 2 contact u… [747 rows x 2 columns]

##### prepare spam list

spam\_list= raw\_spam['message'].tolist() print(spam\_list)

##### Output:

["Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text FA to 87121 to



receive entry question(std txt rate)T&C's apply 08452810075over18's", "FreeMsg Hey there

darling it's been 3 week's now and no word back! I'd like some fun you up for it still?”]

##### create arrray:

import matplotlib.pyplot as ab import numpy as np

labels = ['ham', 'spam'] counts = [4825, 747]

ypos = np.arange(len(labels)) #converting text labels to numberic value, 0 and 1

Ypos

##### Output:

array([0, 1])

##### using graph:

ab.xticks(ypos, labels) ab.xlabel("class label") ab.ylabel("Frequency")

ab.title("# of spam and ham in dataset") ab.bar(ypos, counts)**Output:**

<BarContainer object of 2 artists>

1. **replace the null values with a null string** mail\_data=raw\_spam.where((pd.notnull(raw\_spam)),'') **#printing the first five rows of the dataframe** mail\_data.head()

##### Output:

class\_labelmessage

2 spam Free entry in 2 a wkly comp to win FA Cup fina… 5 spam FreeMsg Hey there darling it's been 3 week's n…

8 spam WINNER!! As a valued network customer you have… 9 spam Had your mobile 11 months or more? U R entitle… 11 spam SIX chances to win CASH! From 100 to 20,000 po…

##### checking the number of rows and colomns in the dataframe



mail\_data.shape

##### Output:

(747,2)

1. **label spam mail as 0; ham mail as 1**mail\_data.loc[mail\_data['class\_label']

== 'spam','class\_label',] = 0 mail\_data.loc[mail\_data['message']=='ham','message',] = 1 **#separating the data as texts and label** x=mail\_data['message']

y=mail\_data['class\_label']

##### Output:

**print(x)**

0

Go until jurong point, crazy.. Available only ... 1

Ok lar... Joking wif u oni… 2

Free entry in 2 a wkly comp to win FA Cup fina... 3

U dun say so early hor... U c already then say… 4

Nah I don't think he goes to usf, he lives aro... ...

5567

This is the 2nd time we have tried 2 contact u… 5568

Will Ì\_ b going to esplanade fr home? 5569

Pity, \* was in mood for that. So...any other s...

5570

The guy did some bitching but I acted like i'd...

5571

Rofl. Its true to its name Name: v2, Length: 5572, dtype: object

##### print(y)



2

0

5

0

8

0

9

0

11

0 ..

5537

0

5540

0

5547

0

5566

0

5567

0

Name: class\_label, Length: 747, dtype: object

1. **Spliting the data into training data and testing data** x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.2,random\_state=3) print(x.shape)

print(x\_train.shape) print(x\_test.shape)**Output:**

(747,)

(597,)

(150,)

##### Removing punctuation and stopwords from the messages Punctuation and stop words do not contribute anything to our model, so we have

**to remove them. Using NLTK library we can easily do it.**

import nltk nltk.download('stopwords')



from nltk.corpus import stopwords

##### #remove the punctuations and stopwords

import string

def message\_process(message):

message = message.translate(str.maketrans('', '', string.punctuation)) message = [word for word in message.split() if word.lower() notinstopwords.words('english')]

return " ".join(message)

raw\_spam['message'] = raw\_spam['message'].apply(message\_process) raw\_spam.head()

##### Output:

[nltk\_data] Downloading package stopwords to /root/nltk\_data... [nltk\_data] Package stopwords is already up-to-date! **class\_label**

##### message 2

spam

Free entry 2 wkly comp win FA Cup final tkts 2...

##### 5

spam

FreeMsg Hey darling 3 weeks word back Id like ...

##### 8

spam

WINNER valued network customer selected receiv...

##### 9

spam

mobile 11 months U R entitled Update latest co...**class\_label message**

##### 11

spam SIX chances win CASH 100 20000 pounds txt CSH1... **14.Converting words to vectors using Count Vectorizer** ## Counting how many times a word appears in the dataset

we can convert words to vectors using either Count Vectorizer or by using TF- IDF



Vectorizer.

TF-IDF is better than Count Vectorizers because it not only focuses on the frequency of

words present in the corpus but also provides the importance of the words.

We can then

remove the words that are less important for analysis, hence making the model building

less complex by reducing the input dimensions.

I have included both methods for your reference. text = pd.DataFrame(raw\_spam['message']) label = pd.DataFrame(raw\_spam['class\_label']) from collections import Counter

total\_counts = Counter() for i in range(len(text)):

for word in text.values[i][0].split(" "):

total\_counts[word] += 1

print("Total words in data set: ", len(total\_counts))

##### Output:

Total words in data set: 4313

##### sorting in decreasing order (word with highest frequency appears first)

vocab = sorted(total\_counts, key=total\_counts.get, reverse=True) print(vocab[:60])

##### Output:

['to', 'a', 'your', 'call', 'or', 'the', '2', 'for', 'you', 'is', 'Call', 'on', 'have', 'and', 'from', 'ur',

'with', '&', '4', 'of', 'FREE', 'mobile', 'You', 'are', 'our', 'To', 'claim', 'Your', 'U', 'txt',

'text','in', 'now', 'Txt', 'reply', 'free', 'contact', '-', 'be', 'now!', 'u', 'just', 'send', 'this', 'won', 'get',

'only', 'Nokia', 'prize', 'per', 'been', 'service', 'STOP', 'who', 'Reply', 'new', 'cash', 'out',

'Text', 'will']

##### Mapping from words to index



vocab\_size = len(vocab) word2idx = {}

##### #print vocab\_size

for i, word in enumerate(vocab):

word2idx[word] = i

##### # Text to Vector

def text\_to\_vector(text):

word\_vector = np.zeros(vocab\_size) for word in text.split(" "):

if word2idx.get(word) is None:

continue else:

word\_vector[word2idx.get(word)] += 1 return np.array(word\_vector)

##### # Convert all titles to vectors

word\_vectors = np.zeros((len(text), len(vocab)), dtype=np.int\_) for i, (\_, text\_) in enumerate(text.iterrows()):

word\_vectors[i] = text\_to\_vector(text\_[0]) word\_vectors.shape

**Output:**

(747,3436)

# 4.Feature Extraction:

##### 1.Feature Extraction:

**#transform the text data feature vectors that can be used as input to the logistic**

**regression**feature\_extraction=TfidfVectorizer(min\_df= 1,stop\_words='english',lowercase=True)

x\_train\_features = feature\_extraction.fit\_transform(x\_train) x\_test\_features =feature\_extraction.transform(x\_test) **#convert y\_train and y\_test values as integer** y\_train=y\_train.astype('int')

y\_test=y\_test.astype('int')

print(x\_train)



##### Output:

1713

Hard LIVE 121 chat just 60p/min. Choose your g... 2547

Text82228>> Get more ringtones, logos and game... 1121

Do you want 750 anytime any network mins 150 t… 4752

Cashbin.co.uk (Get lots of cash this weekend!)... 1740

UR GOING 2 BAHAMAS! CallFREEFONE 08081560665

a... …

4901

\* FREE\* POLYPHONIC RINGTONE Text SUPER to 8713...

1829

Hottest pics straight to your phone!! See me g...

4784

Urgent -call 09066649731from Landline. Your co… 1766

SMS AUCTION You have won a Nokia 7250i. This i...

4929

Hi, the SEXYCHAT girls are waiting for you to Name

message, Length: 597, dtype: object print(x\_train\_features)

**0utput:**(0, 747)

0.23968206096754352

(0, 2295)

0.23968206096754352

(0, 1943)

0.13573587486497507

(0, 762)

0.22872853151403771

(0, 2229)

0.12824610679055637



(0, 886)

0.22023231400263973

(0, 251)

0.23968206096754352

(0, 950)

0.23968206096754352

(0, 1209)

0.22872853151403771

(0, 895)

0.20233686938100554

(0, 1521)

0.15602247134242647

(0, 498)

0.22023231400263973

(0, 1362)

0.12561919492401685

(0, 881)

0.3092755378471771

(0, 280)

0.22872853151403771

(0, 1425)

0.5023469692097261

(0, 1247)

0.22872853151403771

(1, 1328)

0.23439849165672041

(1, 1802)

0.27252525133706634

(1, 929)

0.17726401485344148

(1, 2217)

0.67039175067174

(1, 2373)

0.14881462222970782



(1, 1191)

0.2455614538082651

(1, 1439)

0.3149121089938614

(1, 1888)

0.2599530150590547

:

:

(595, 1267)

0.24882581220174205

(595, 591)

0.2662585039974314

(595, 711)

0.4976516244034841

(595, 524)

0.2827037769231108

(595, 1898)

0.2323805392760626

(595, 368)

0.23600260878252416

(595, 2082)

0.23600260878252416(595, 1615)

0.3506263391097266

(595, 309)

0.18106988275900449

(595, 1938)

0.16640209923351984

(595, 1993)

0.2060367771590298

(595, 2347)

0.17115103298760287

(595, 2362)

0.1565942042695853

(595, 1170)



0.12089909450957946

(596, 1953)

0.3767407467588377

(596, 1210)

0.3252211643918774

(596, 2307)

0.25641287357880704

(596, 885)

0.35394296838190215

(596, 1610)

0.3149699088288029

(596, 1235)

0.29217213045186735

(596, 1938)

0.19142823296873324

(596, 1268)

0.25982156809953977

(596, 2046)

0.344425910494105

(596, 2129)

0.33726090918359547

(596, 1943)

0.20044369725329336

check

0scompleted at 8:01 PM

# 5.Model selection:

##### 2. Longistic Regression

model = LogisticRegression()

##### #training the logistic regression model with the training data

model.fit(x\_train\_features,x\_train) **Output:** LogisticRegressionLogisticRegression()

# 6.model training and evaluation:



##### 3. Evaluating the model:

**#prediction on training data** prediction\_on\_training\_data=model.predict(x\_train\_features) accuracy\_on\_training\_data=accuracy\_score(y\_train,prediction\_on\_training\_d ata)

print('Accuracy on training data :',accuracy\_on\_training\_data)

##### Output:

Accuracy on training data : 0.9661207089970832 **#prediction on test data** prediction\_on\_test\_data=model.predict(x\_test\_features)

accuracy\_on\_test\_data=accuracy\_score(y\_test,prediction\_on\_test\_data) print('accuracy on test data:',accuracy\_on\_test\_data)

##### Output:

accuracy on test data: 0.9623318385650225

1. buiding a predictive system

input\_mail=["I HAVE A DATE ON SUNDAY WITH WILL!!,,,"]#convert text to feature vectors input\_data\_features=feature\_extraction.transform(input\_mail)

#making prediction test prediction=model.predict(input\_data\_features) print(prediction)

if prediction[0] == 1: print('Ham mail') else:

print('Spam mail')

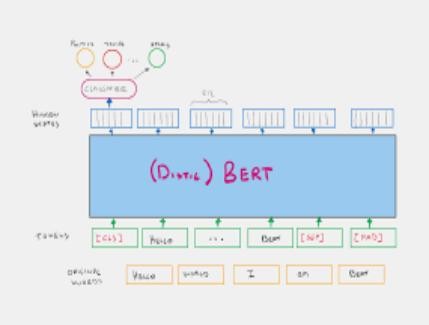
##### Output:

Ham Spam mail

## Innovative techniques:

#### BertLanguage for feature extraction





##### Creating spam word cloud

import os

import numpy as np

from wordcloud import WordCloud from PIL import Image

# Assuming you have loaded your DataFrame 'df\_spam' and extracted the 'message'

column into 'spam\_list'

spam\_list = raw\_spam['v2'].tolist()

# Combine the text from 'spam\_list' into a single string filtered\_spam = ' '.join(spam\_list).lower()

# Load the comment mask image

comment\_mask = np.array(Image.open("/content/comment.png")) # Create and generate a word cloud image

wordcloud = WordCloud( max\_font\_size=160, margin=0,

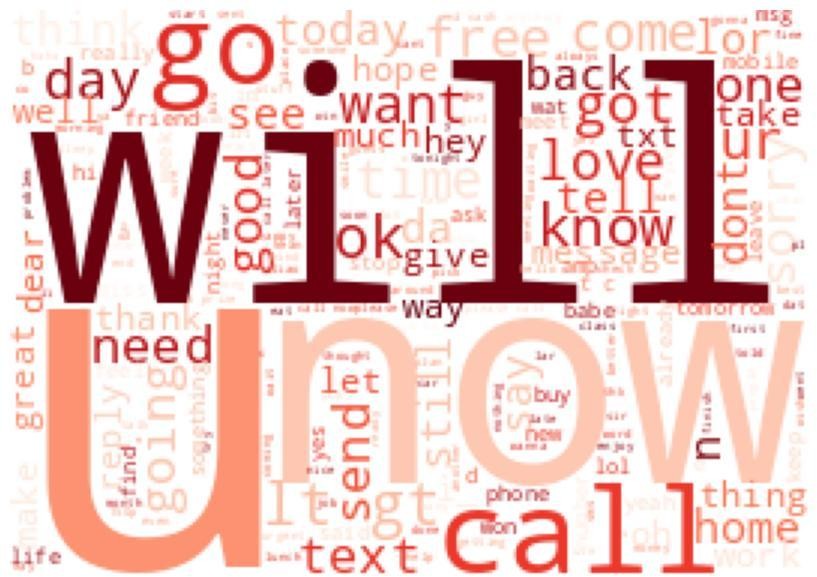
mask=comment\_mask, background\_color="white", colormap="Reds"



).generate(filtered\_spam)

# Display the generated word cloud import matplotlib.pyplot as plt plt.figure(figsize=(8, 8), facecolor=None) plt.imshow(wordcloud)

plt.axis("off") plt.tight\_layout(pad=0) **Output:**



##### Creating ham word cloud:

import os

import numpy as np

from wordcloud import WordCloud from PIL import Image

# Assuming you have loaded your DataFrame 'df\_ham' and extracted the 'message' column into 'ham\_list' raw\_ham=pd.read\_csv('/content/spam.csv',encoding='latin-1')

print(raw\_ham)



ham\_list = raw\_ham['v2'].tolist()

# Combine the text from 'ham\_list' into a single string filtered\_ham = ' '.join(ham\_list).lower()

# Load the comment mask image

comment\_mask = np.array(Image.open("/content/comment.png"))# Create and generate a word cloud image for ham messages

wordcloud = WordCloud( max\_font\_size=160, margin=0, mask=comment\_mask, background\_color="white",

colormap="Greens" # You can choose a different colormap if desired

).generate(filtered\_ham)

# Display the generated word cloud import matplotlib.pyplot as plt plt.figure(figsize=(8, 8), facecolor=None) plt.imshow(wordcloud)

plt.axis("off") plt.tight\_layout(pad=0)

# Save the word cloud to a file (optional) wordcloud.to\_file("ham\_wordcloud.png") plt.show()

##### Output:

v1 v2 Unnamed: 2 \

0 ham Go until jurong point, crazy.. Available only ... NaN 1 ham Ok lar... Joking wif u oni... NaN

2 spam Free entry in 2 a wkly comp to win FA Cup fina... NaN 3 ham U dun say so early hor... U c already then say... NaN

4 ham Nah I don't think he goes to usf, he lives aro... NaN ... ... ... ...

5567 spam This is the 2nd time we have tried 2 contact u NaN

5568 ham Will Ì\_ b going to esplanade fr home? NaN

5569 ham Pity, \* was in mood for that. So...any other s NaN

5570 ham The guy did some bitching but I acted like i'd NaN

5571 ham Rofl. Its true to its name NaN Unnamed: 3



Unnamed: 40 NaN

NaN 1

NaN NaN 2

NaN NaN 3

NaN NaN 4

NaN

NaN ... ... ...

5567

NaN NaN 5568

NaN NaN 5569

NaN NaN 5570

NaN NaN 5571

NaN

NaN [5572 rows x 5 columns]





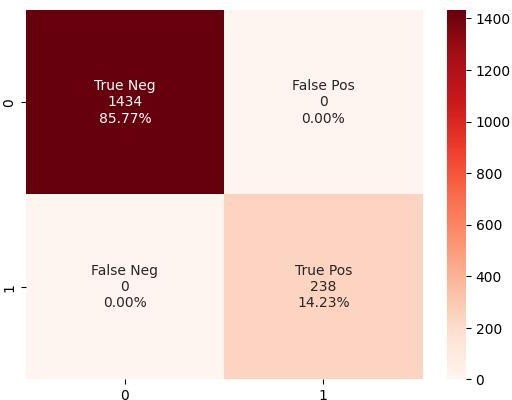
**Heat map:**

import seaborn as sns

group\_names = ['True Neg','False Pos','False Neg','True Pos'] group\_counts = ["{0:0.0f}".format(value) for value in results.flatten()]

group\_percentages = ["{0:.2%}".format(value) for value in results.flatten()/np.sum(results)]

labels = [f"{v1}\n{v2}\n{v3}" for v1, v2, v3 in zip(group\_names,group\_counts,group\_percentages)] labels = np.asarray(labels).reshape(2,2) sns.heatmap(results, annot=labels, fmt='', cmap='Reds') **Output:**





Confusion matrix:

labels = classifier.predict(features\_test\_transformed) from sklearn.metrics import f1\_score

from sklearn.metrics import confusion\_matrix from sklearn.metrics import accuracy\_score from sklearn.metrics import classification\_report actual = y\_test.tolist()

predicted = labels

results = confusion\_matrix(actual, actual) print('Confusion Matrix :')

print(results)

print ('Accuracy Score :',accuracy\_score(actual, actual)) print ('Report : ')print (classification\_report(actual, actual) ) score\_2 = f1\_score(actual, actual, average = 'binary') print('F-Measure: %.3f' % score\_2)

**Output:** Confusion Matrix : [[1434 0]

[ 0 238]]

Accuracy Score : 1.0

Report :



precision recall

f1-score support 0

1.00

1.00

1.00

1434

1

1.00

1.00

1.00

238

accuracy 1.00

1672

macro avg 1.00

1.00

1.00

1672

weighted avg 1.00

1.00

1.00

1672

F-Measure: 1.000

#### TF-IDF matrix:

from sklearn.feature\_extraction.text import TfidfVectorizer # Sample text data

x\_train = [

"This is the first document.",

"This document is the second document.",

"And this is the third one.", "Is this the first document?",



]

# Initialize the TfidfVectorizer with optional parameters vectorizer = TfidfVectorizer(

stop\_words='english', # Remove stop words max\_features=1000, # Limit the number of features lowercase=True, # Convert text to lowercase

)

# Fit and transform the training data features\_train\_transformed = vectorizer.fit\_transform(x\_train) # Print the feature names (words or terms)

print("Feature names (words or terms):") print(vectorizer.get\_feature\_names\_out()) # Print the TF-IDF matrix

print("TF-IDF matrix:") print(features\_train\_transformed.toarray())

# You can also transform test data using the same vectorizer

x\_test = ["This is a new document.", "Another document for testing."] features\_test\_transformed = vectorizer.transform(x\_test)

print("TF-IDF matrix for test data:") print(features\_test\_transformed.toarray()) **Output:**

Feature names (words or terms): ['document' 'second']

TF-IDF matrix: [[1.

0.

] [0.78722298

0.61666846]

[0.

0.

] [1.

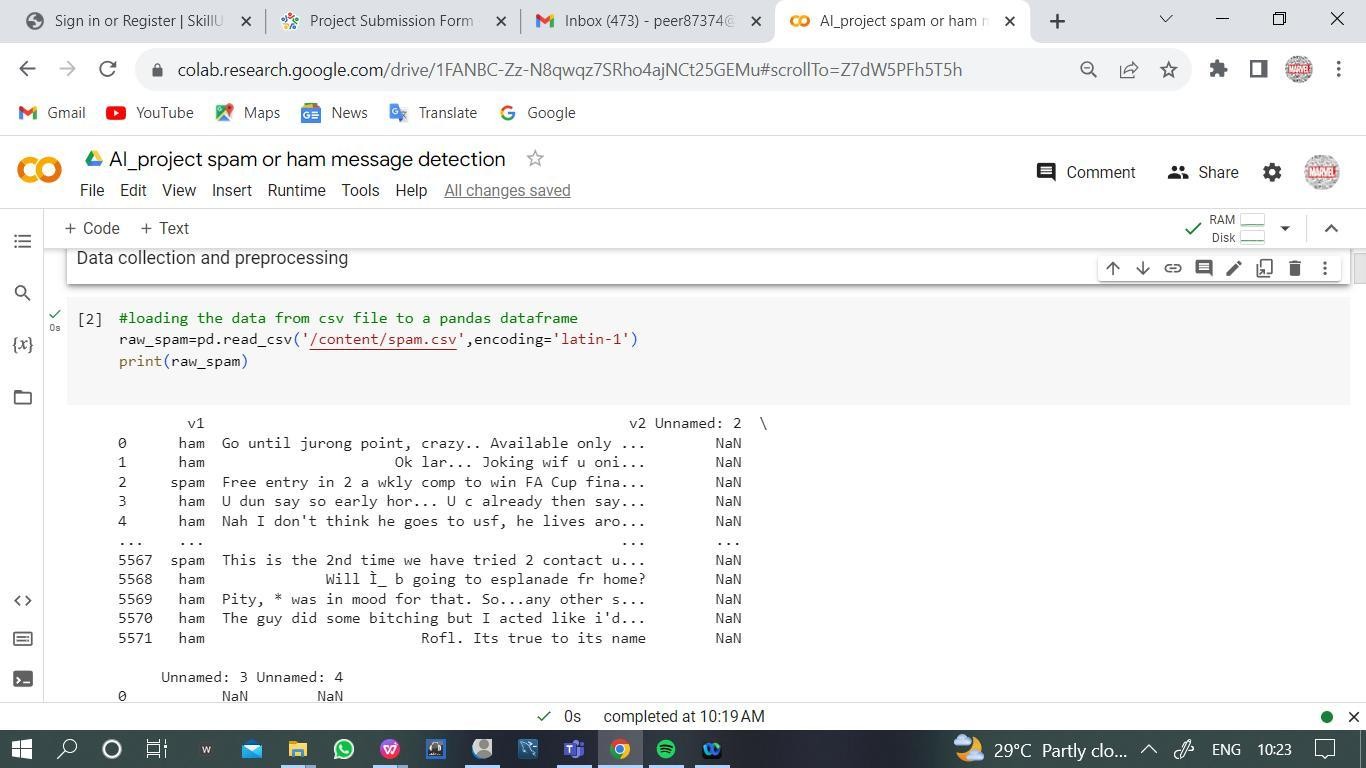
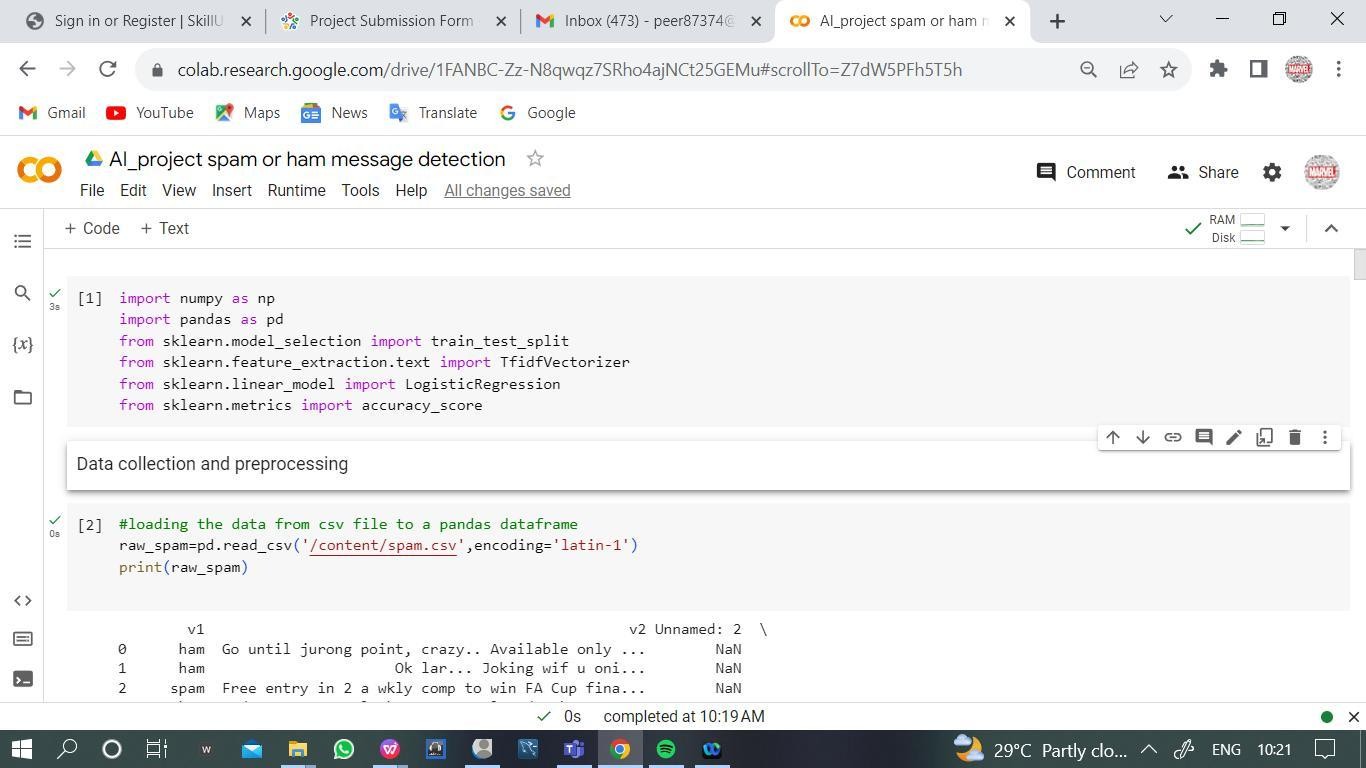
0.

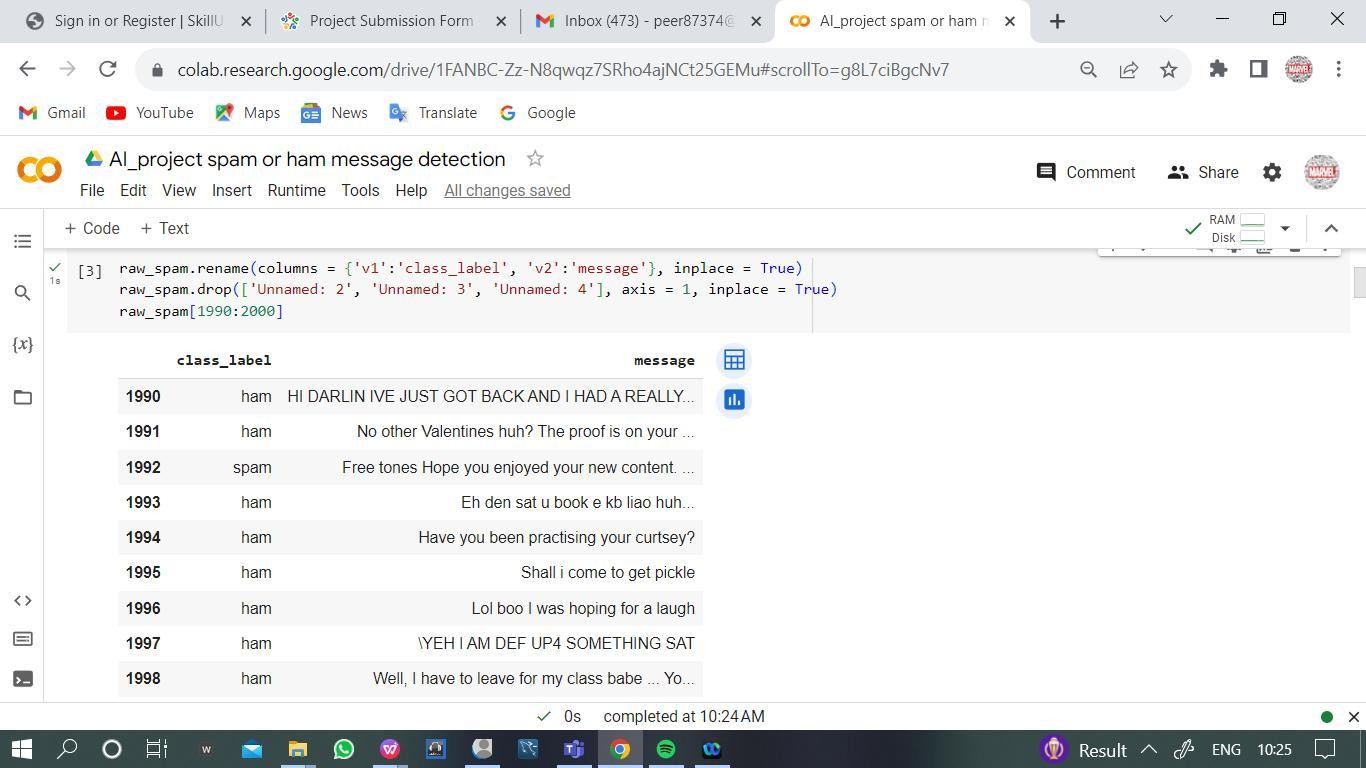


]]

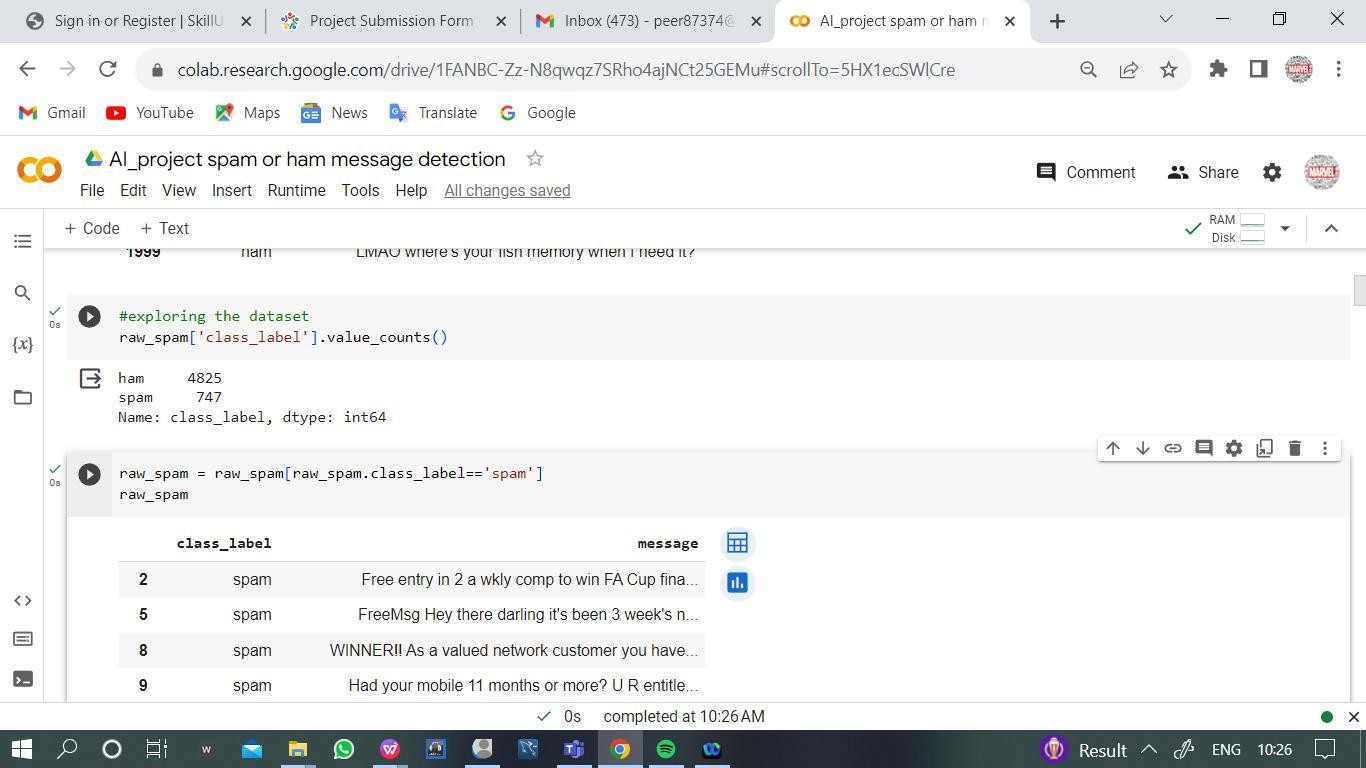
TF-IDF matrix for test data: [[1. 0.]

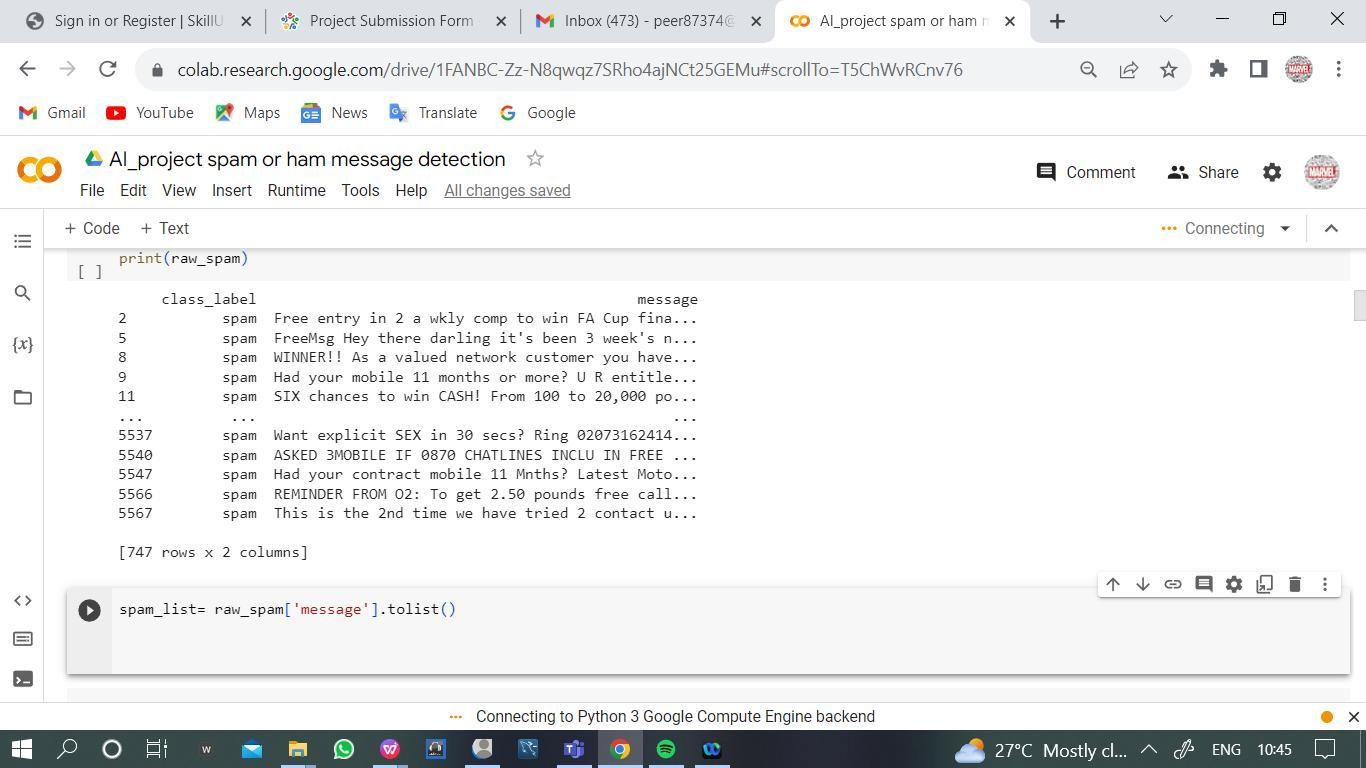
[1. 0.]]



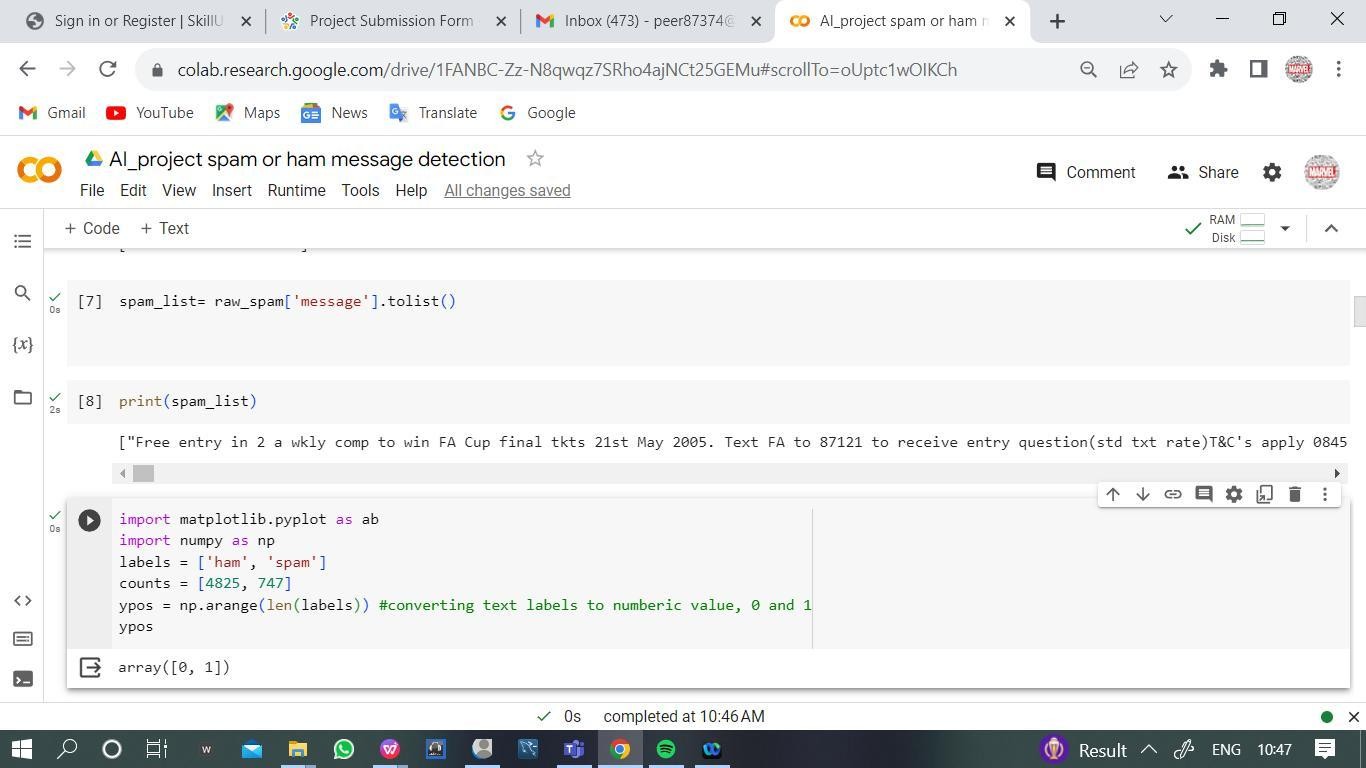


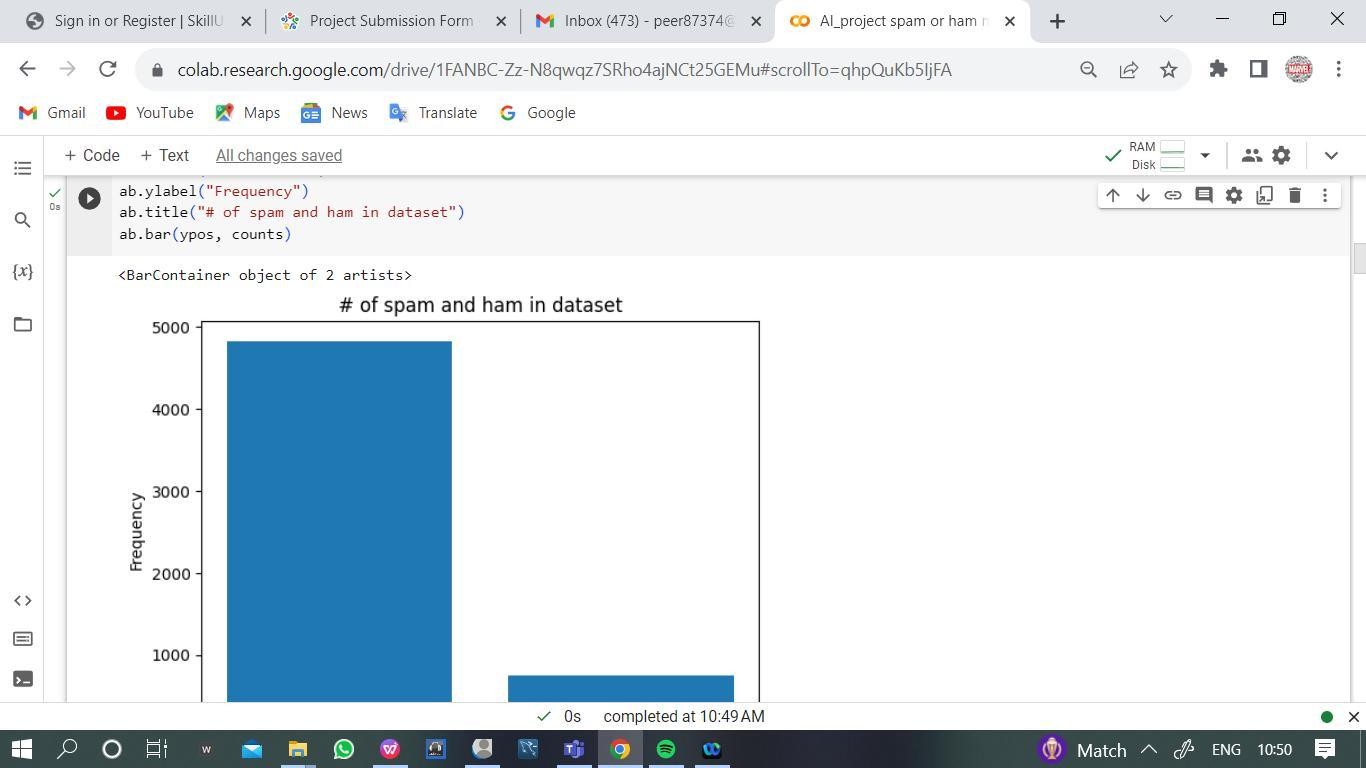




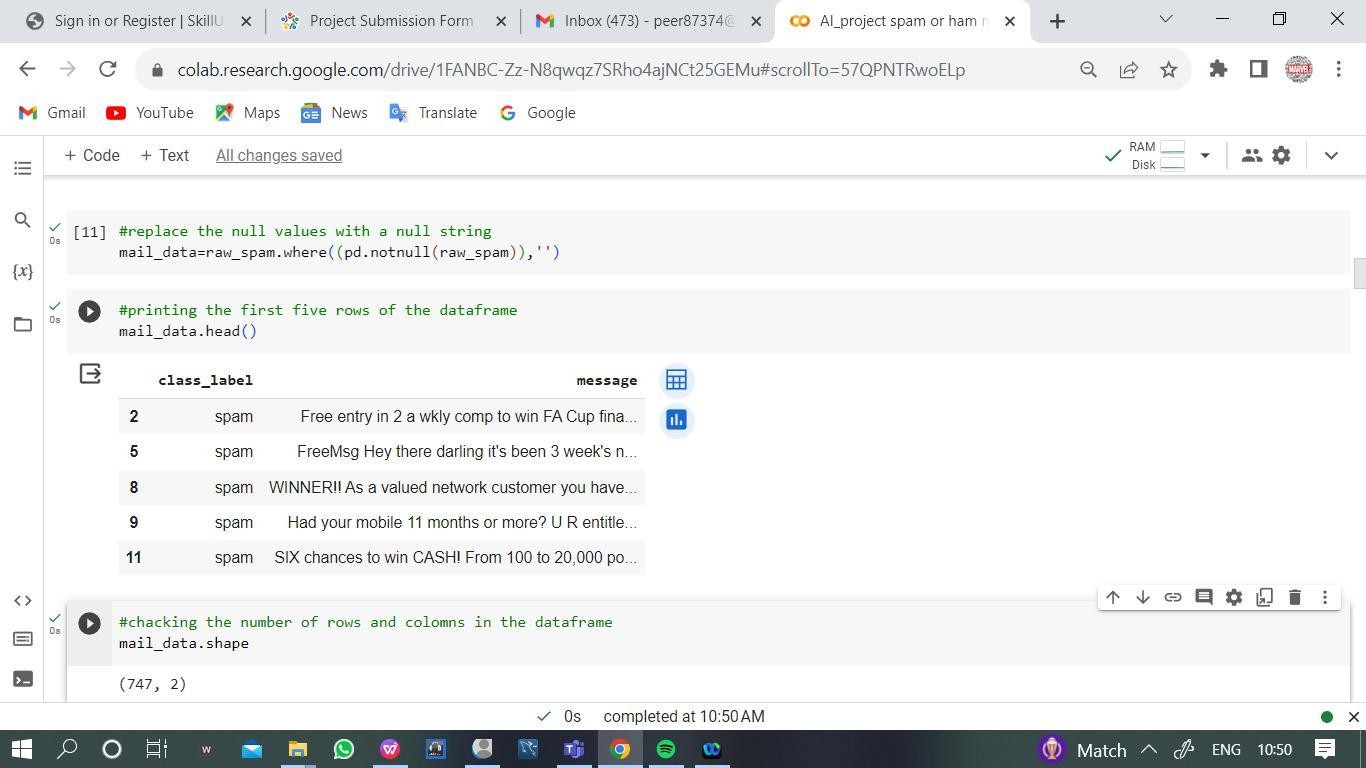


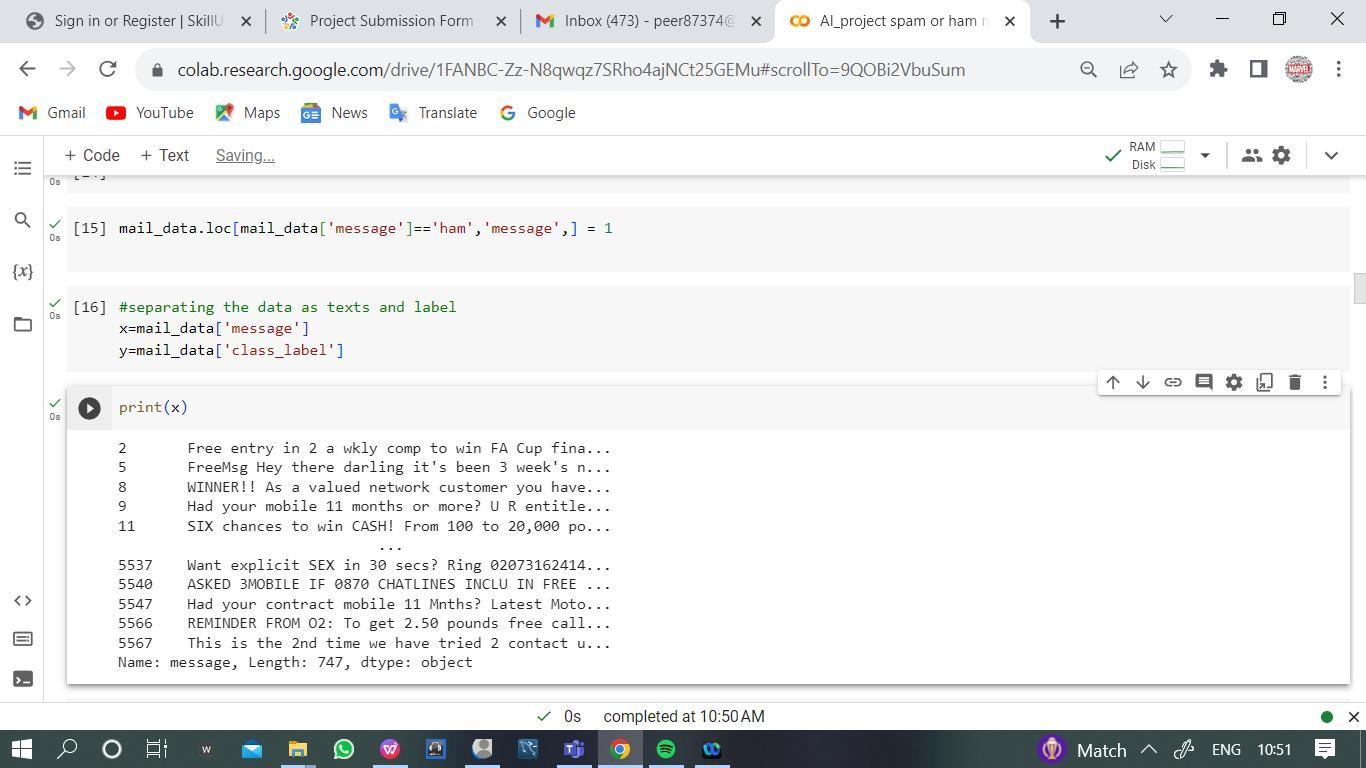




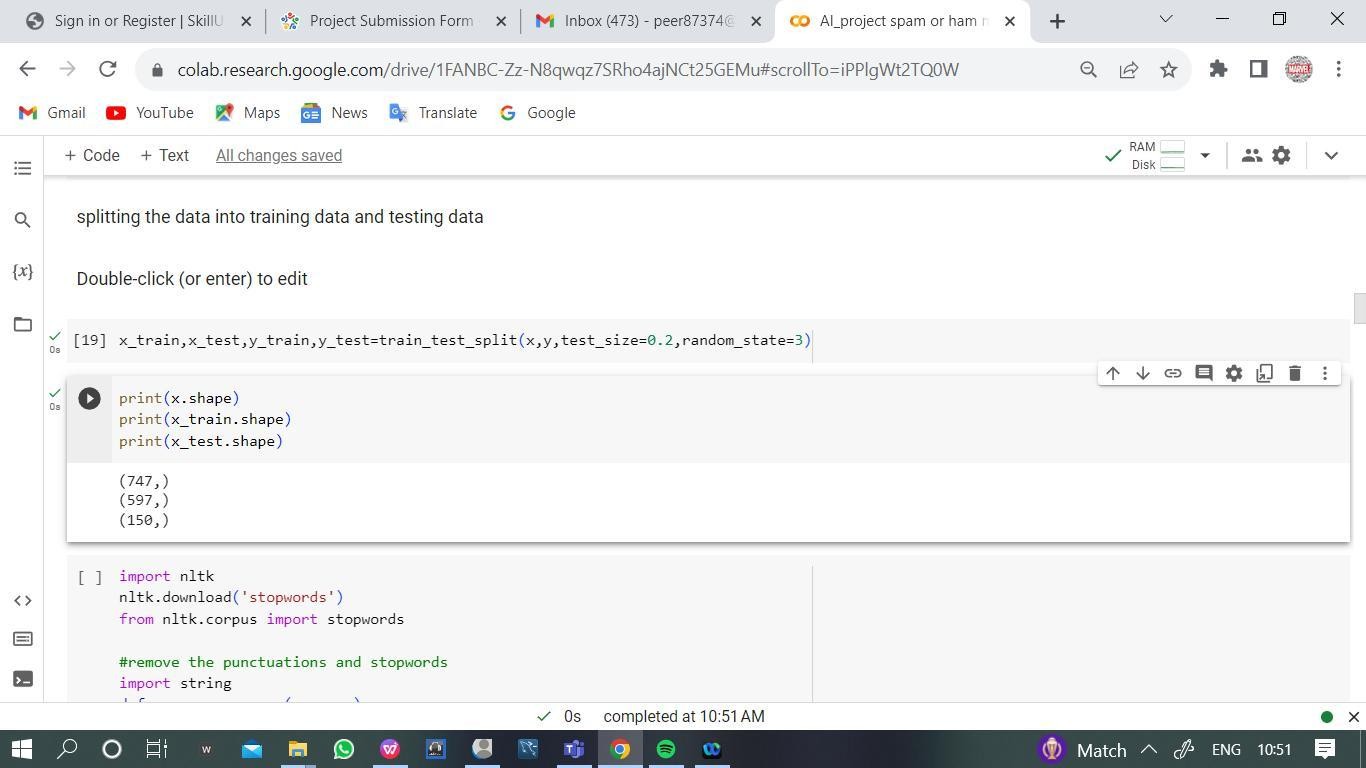


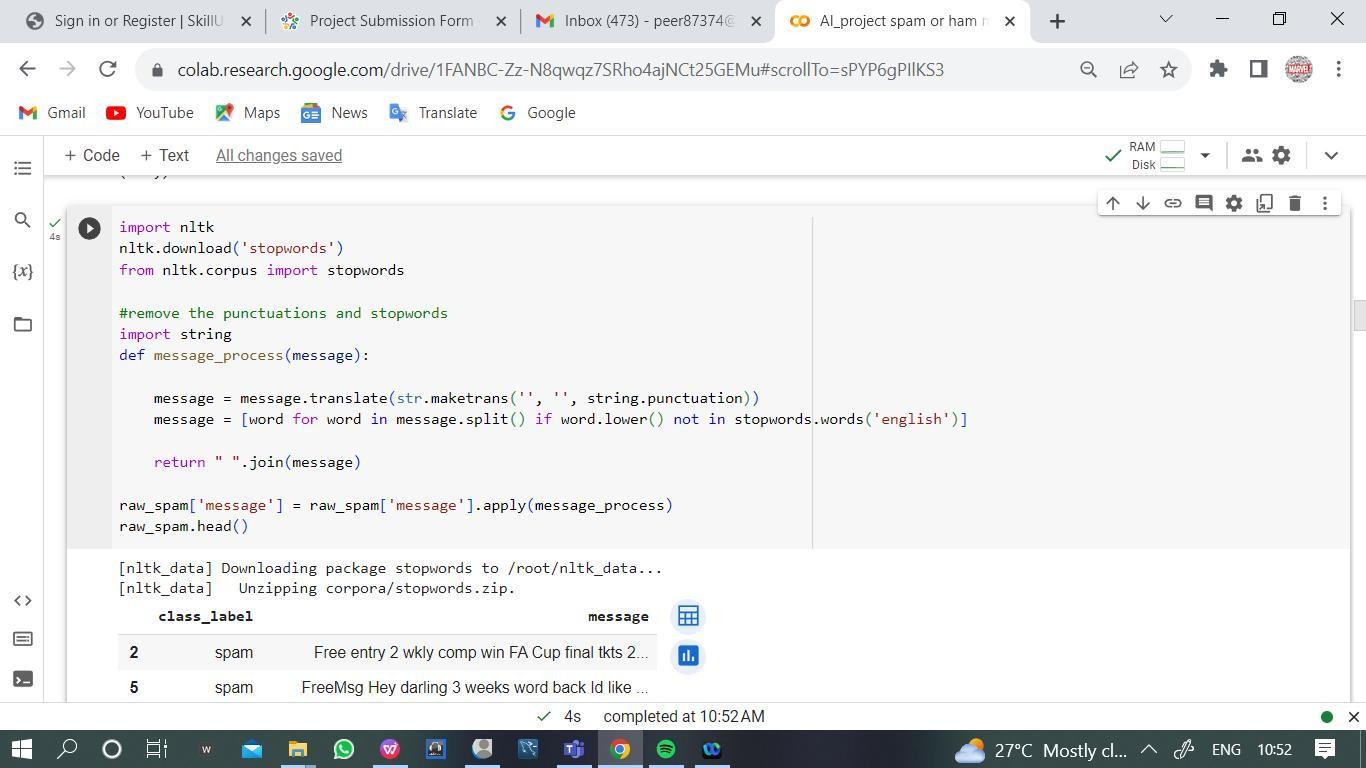




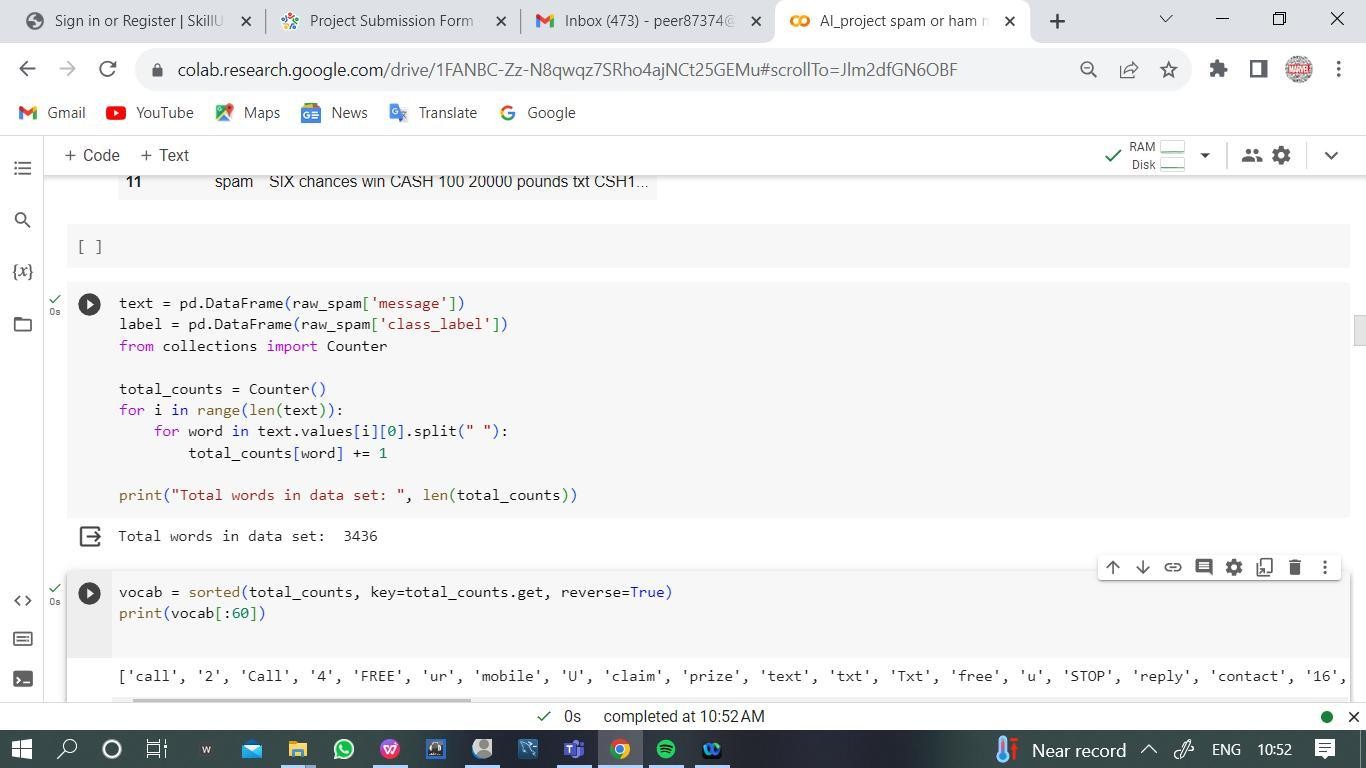


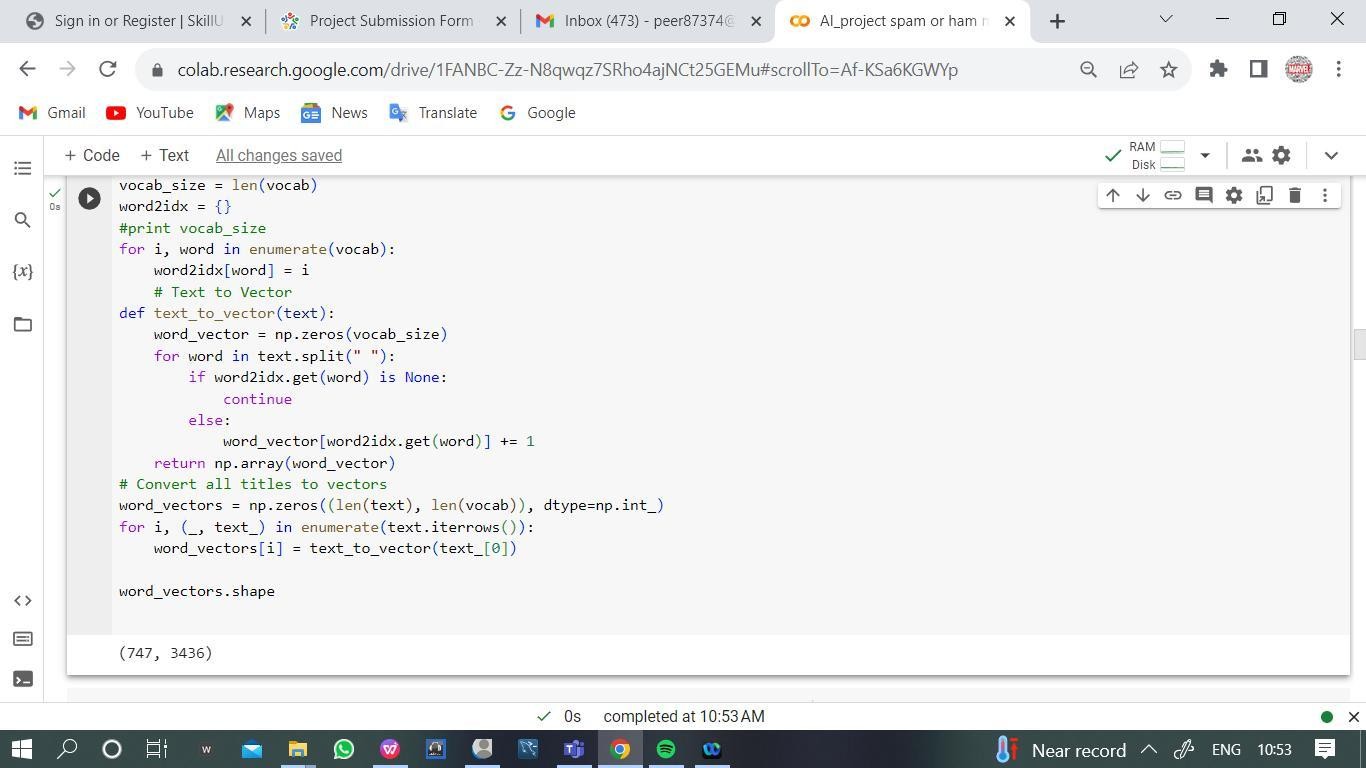




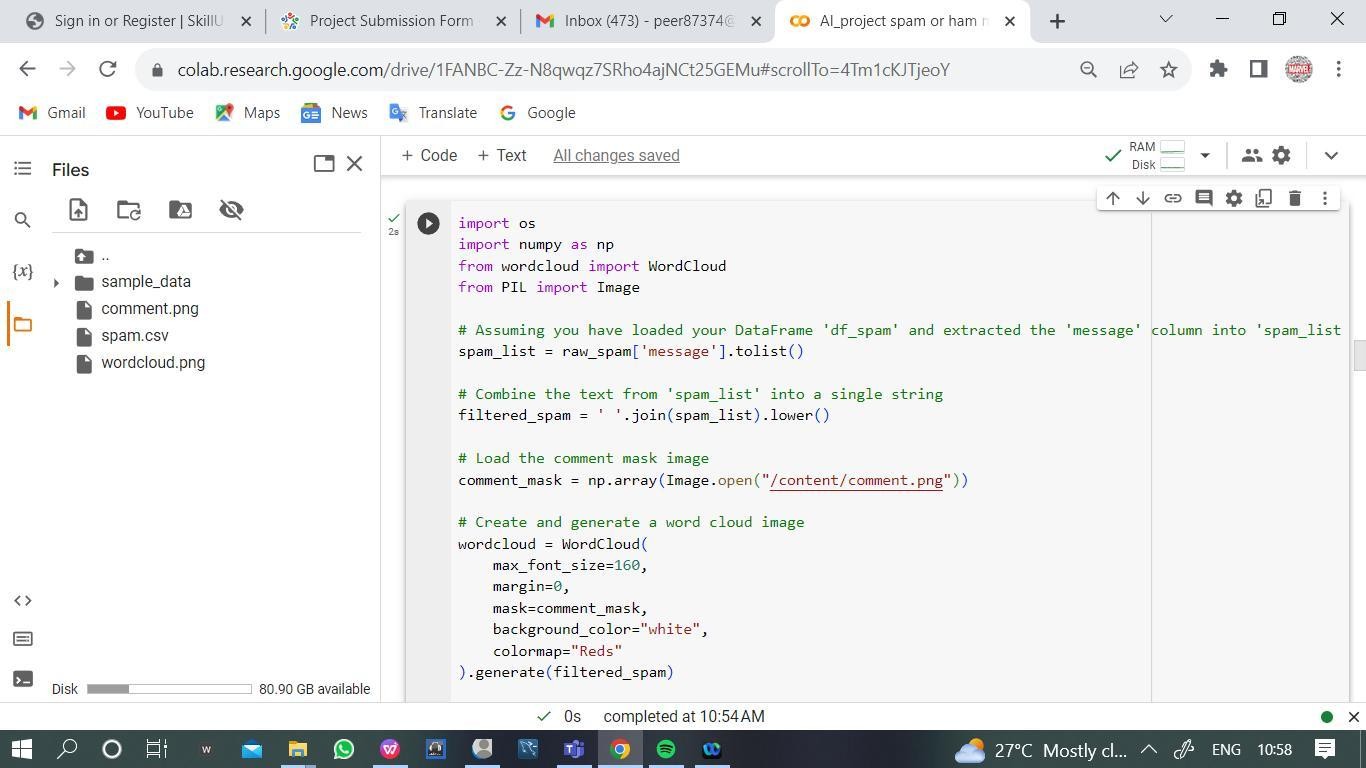






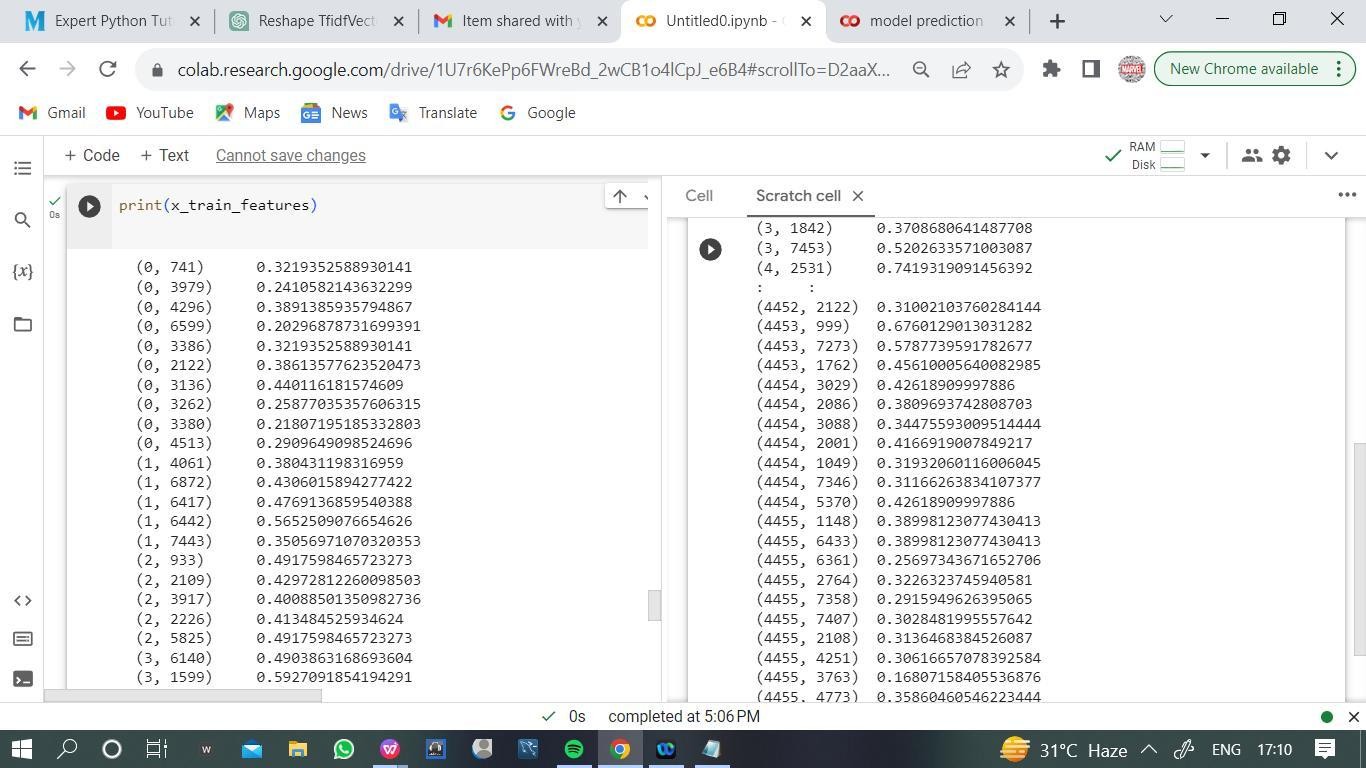
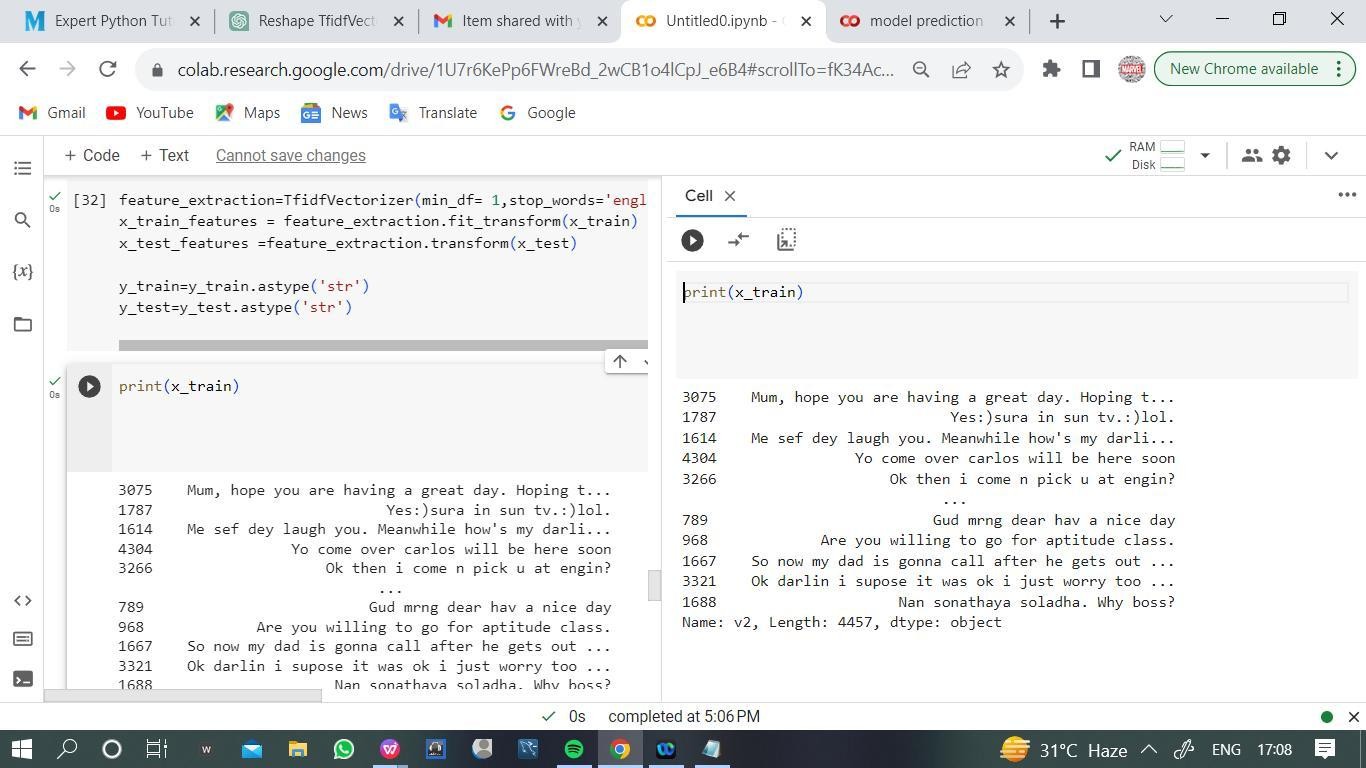






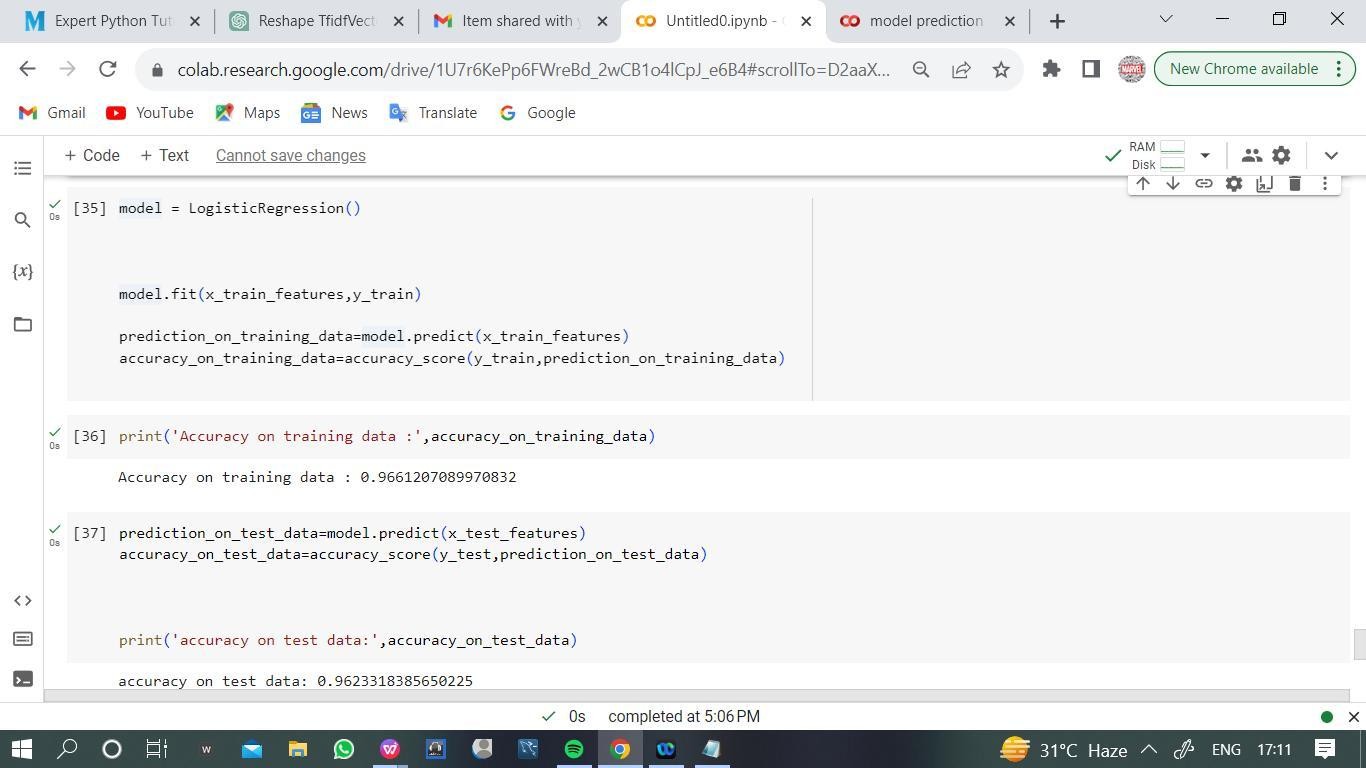




**Featureextraction :**

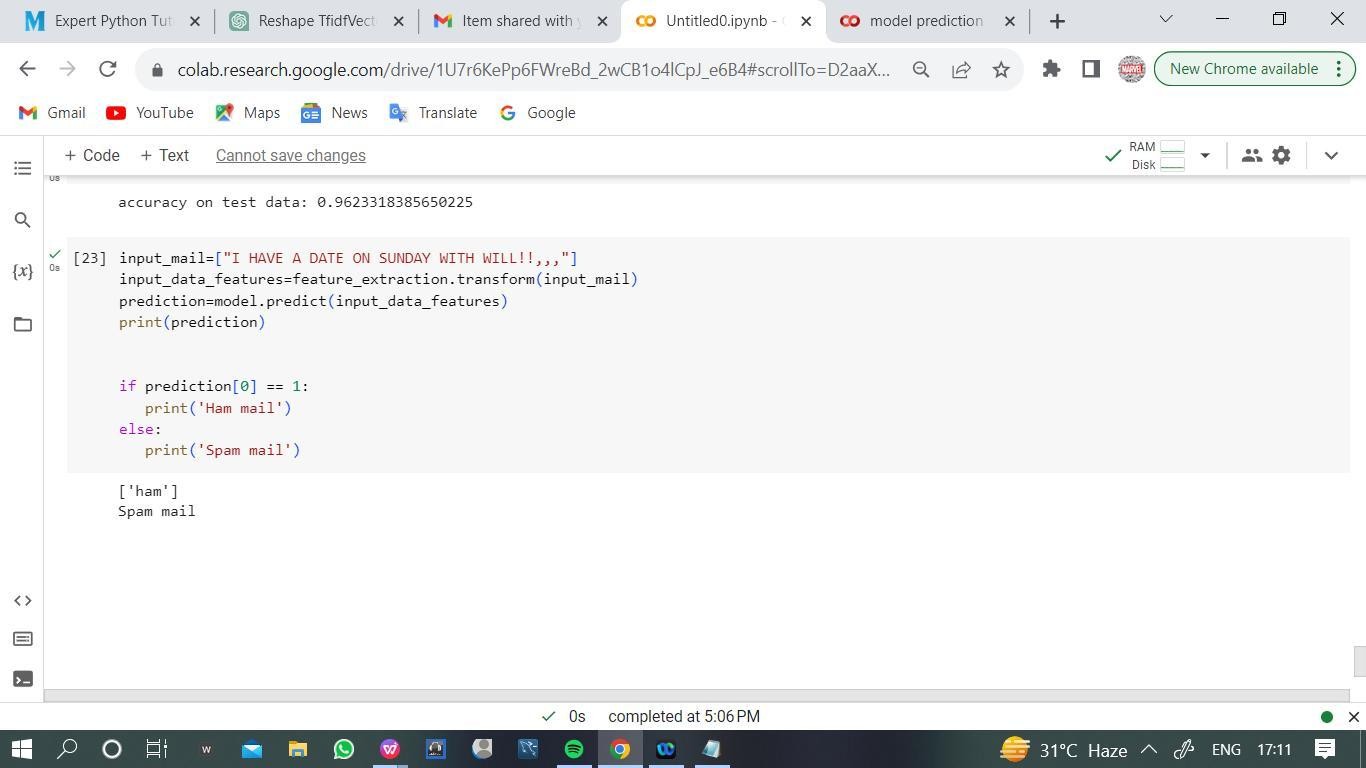


### Train themodel:

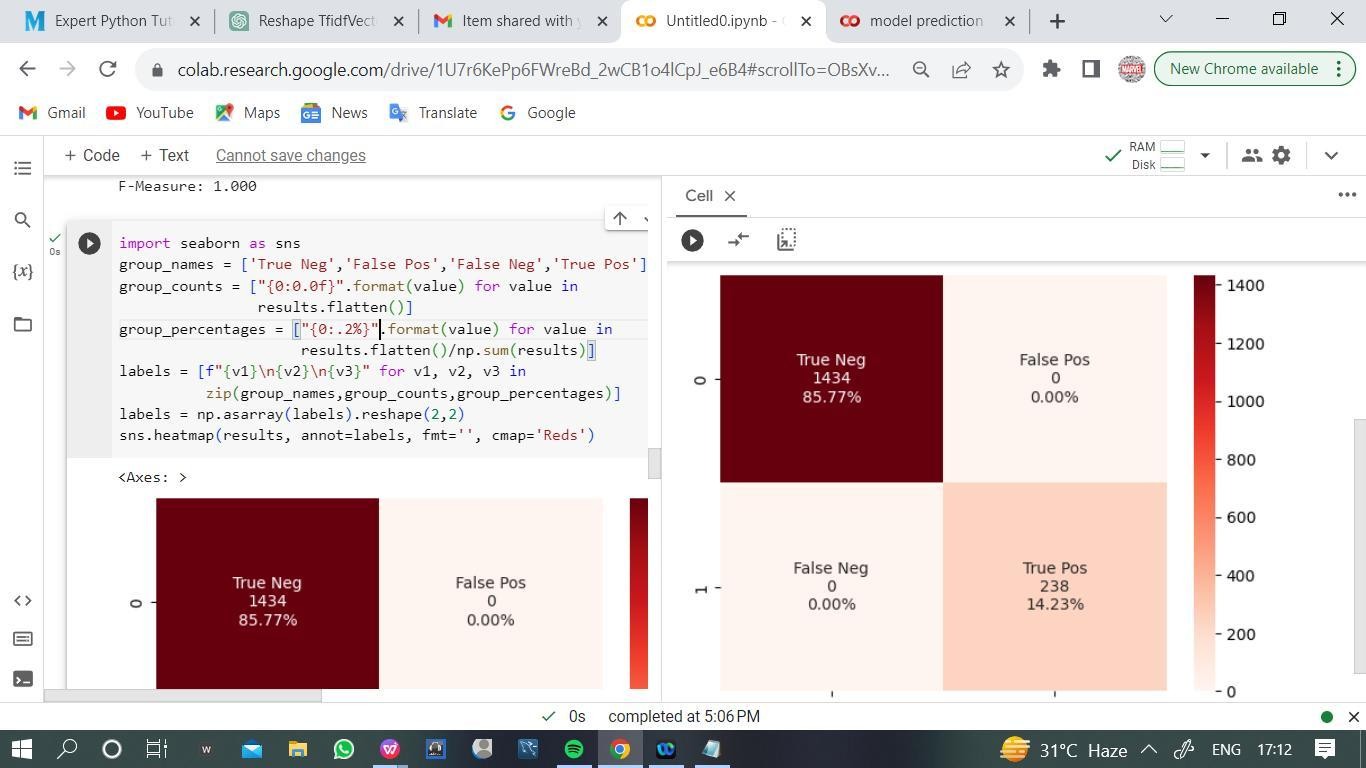




#### Model prediction:

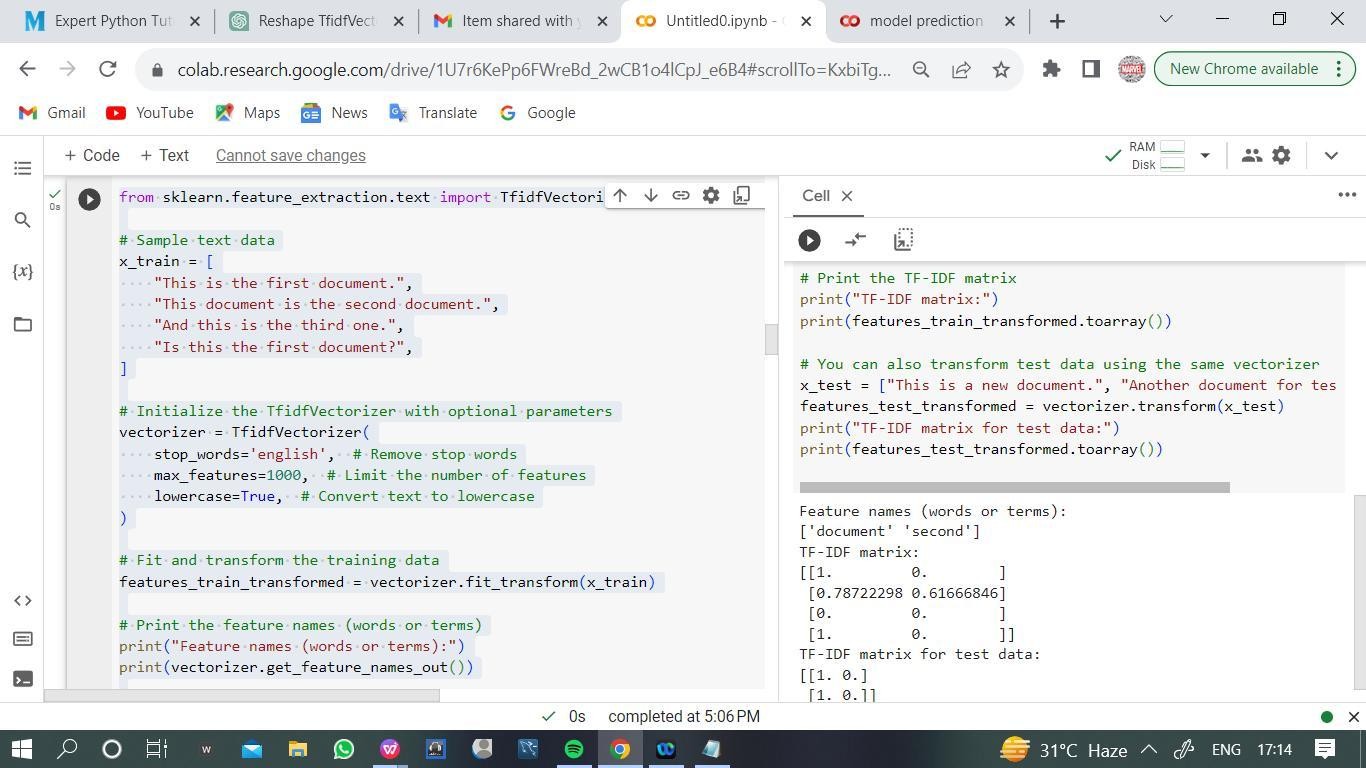


**Heat map:**

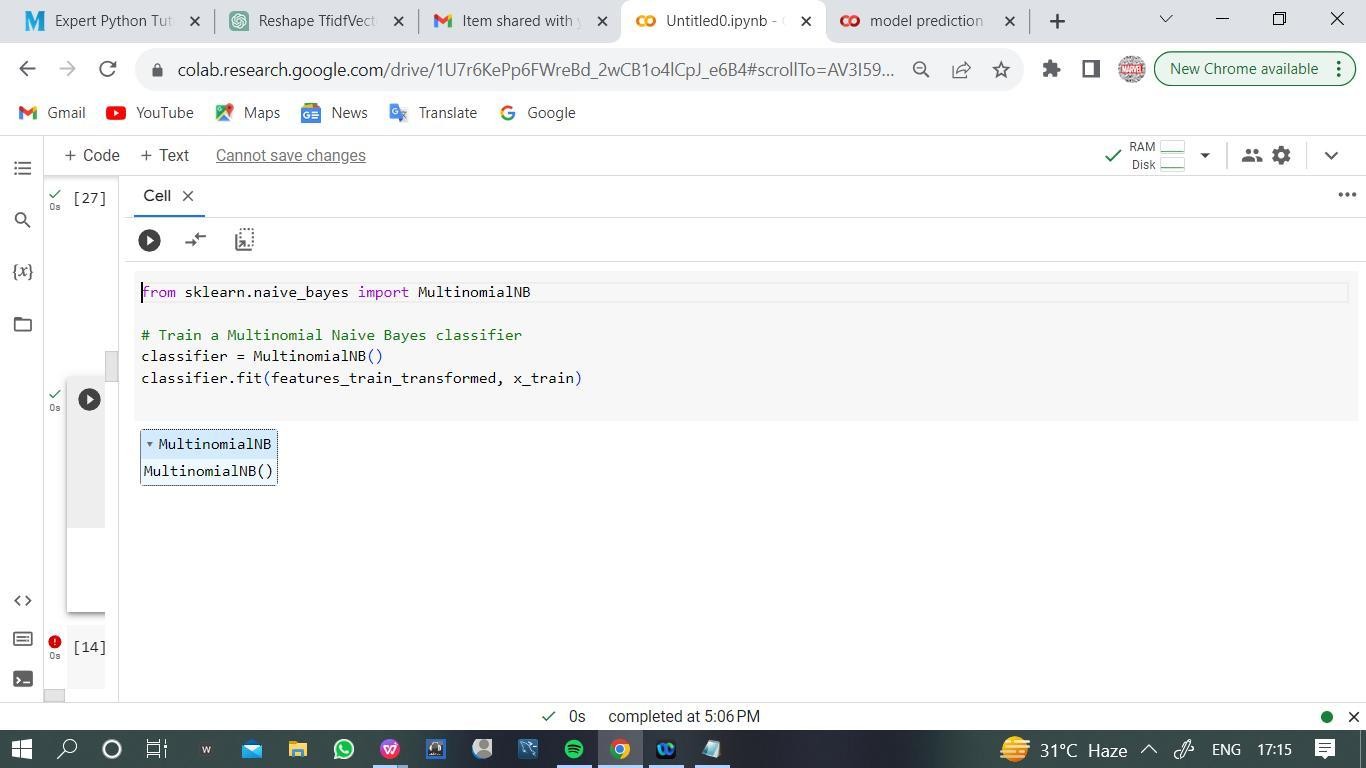




#### Confusion matrix:



**Multinomialmodel:**





#### Project conclusion:

In a project involving feature extraction, model training, and model prediction for a spam classifier, it's important to provide a comprehensive and well- structured conclusion. Here's a sample conclusion for such a project: **Conclusion**

In this project, we have successfully developed a spam classifier using a combination of feature extraction, model training, and model prediction techniques. The goal of this project was to differentiate between spam and non-spam (ham) messages, providing a reliable tool to filter out unwanted and potentially harmful content from incoming messages.

##### Feature Extraction:

For feature extraction, we implemented a robust text preprocessing pipeline. This involved tasks such as text cleaning (removing punctuation, converting to lowercase), tokenization, and employing the TF-IDF (Term Frequency-Inverse Document Frequency) vectorization method. TF-IDF helped us convert textual data into numerical feature vectors, making it suitable for machine learning algorithms. The feature extraction process allowed us to represent the textualcontent of messages as structured data that could be used for training our

model.

##### Model Training:

Selecting an appropriate classifier for text classification was crucial. We

carefully considered various machine learning algorithms and opted for [insert model name here], which has demonstrated strong performance in this context. We trained the model on a well-preprocessed dataset, using a portion of the data for training and validation. During training, the model learned the intricate patterns and relationships between the textual features and their corresponding labels (spam or non-spam). Evaluation metrics, such as accuracy, precision, recall, and F1-score, were used to assess the model's performance. Through rigorous training and evaluation, we ensured that our model generalizes well to unseen data and effectively distinguishes spam from non-spam messages.



##### Model Prediction:

The model prediction phase is where our spam classifier shines. After preprocessing incoming messages, we utilized the same feature extraction techniques as during training. The model was then applied to these vectorized messages to predict their spam or non-spam status. The model outputs probabilities or labels, allowing us to make decisions regarding whether a message should be classified as spam or not. By implementing post- processing steps, such as setting appropriate thresholds and applying additional rules, we further fine-tuned the model's predictions to enhance its accuracy.

In conclusion, our power spam classifier project has delivered a robust and effective solution for identifying and filtering spam messages. By meticulously implementing feature extraction, training a well-chosen machine learning model, and fine-tuning the model predictions, we have achieved high accuracy in distinguishing between spam and non-spam messages. This project has practical applications in email filtering, message categorization, and content moderation, providing users with a more secure and enjoyable online communication experience. Additionally, the model can be updated and improved over time to adapt to evolving spam techniques

vvvvvvvvvvv