**BUILDING A AI – POWERED SPAM CLASSIFIER**

**By: Kevin B.**

**211121104032**

**Introduction to a Spam Classifier Model Document**

**Title : Building an Effective Spam Classifier: A Deep Dive into Text Classification**

**Abstract:**

**The exponential growth of digital communication has led to an ever-increasing influx of unsolicited and potentially harmful messages, commonly known as spam. In response to this pervasive issue, spam classifiers have become a crucial part of modern communication systems, ensuring that legitimate messages reach their intended recipients while filtering out unwanted content.**

**This document provides an in-depth introduction to the concept of spam classification, its significance, and the methods and techniques used to build effective spam classifiers. Whether you are a developer, data scientist, or a business looking to enhance your email security, this guide will serve as a comprehensive starting point for understanding and implementing spam classifiers.**

**Overview of the process:**

**The following is an overview of the process of building a house**

**price prediction model by feature selection, model training, and**

**evaluation:**

**1. Prepare the data: This includes cleaning the data, removing**

**outliers, and handling missing values.**

**2. Perform feature selection: This can be done using a variety of**

**methods, such as correlation analysis, information gain, and recursive**

**feature elimination.**

**3. Train the model: There are many different machine learning**

**algorithms that can be used for house price prediction. Some popular**

**choices include linear regression, random forests, and gradient boosting**

**machines.**

**4. Evaluate the model: This can be done by calculating the mean**

**squared error (MSE) or the root mean squared error (RMSE) of the**

**model's predictions on the held-out test set.**

**5. Deploy the model: Once the model has been evaluated and found**

**to be performing well, it can be deployed to production so that it can be**

**used to predict the house prices of new houses.**

**Importing the Dependencies :**

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

Data Collection & Pre-Processing

# loading the data from csv file to a pandas Dataframe  
raw\_mail\_data = pd.read\_csv('/content/mail\_data.csv')

print(raw\_mail\_data)

Category Message  
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...  
... ... ...  
5567 spam This is the 2nd time we have tried 2 contact u...  
5568 ham Will ü 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  
  
[5572 rows x 2 columns]

# replace the null values with a null string  
mail\_data = raw\_mail\_data.where((pd.notnull(raw\_mail\_data)),'')

# printing the first 5 rows of the dataframe  
mail\_data.head()

Category Message  
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...

<google.colab.\_quickchart\_helpers.SectionTitle at 0x7811ac43afe0>

from matplotlib import pyplot as plt  
import seaborn as sns  
\_df\_0.groupby('Category').size().plot(kind='barh', color=sns.palettes.mpl\_palette('Dark2'))  
plt.gca().spines[['top', 'right',]].set\_visible(False)

from matplotlib import pyplot as plt  
import seaborn as sns  
\_df\_1.groupby('Message').size().plot(kind='barh', color=sns.palettes.mpl\_palette('Dark2'))  
plt.gca().spines[['top', 'right',]].set\_visible(False)

<google.colab.\_quickchart\_helpers.SectionTitle at 0x7811ac43bfa0>

from matplotlib import pyplot as plt  
import seaborn as sns  
import pandas as pd  
plt.subplots(figsize=(8, 8))  
df\_2dhist = pd.DataFrame({  
 x\_label: grp['Message'].value\_counts()  
 for x\_label, grp in \_df\_2.groupby('Category')  
})  
sns.heatmap(df\_2dhist, cmap='viridis')  
plt.xlabel('Category')  
plt.ylabel('Message')

# checking the number of rows and columns in the dataframe  
mail\_data.shape

(5572, 2)

Label Encoding

# label spam mail as 0; ham mail as 1;  
  
mail\_data.loc[mail\_data['Category'] == 'spam', 'Category',] = 0  
mail\_data.loc[mail\_data['Category'] == 'ham', 'Category',] = 1

spam - 0

ham - 1

# separating the data as texts and label  
  
X = mail\_data['Message']  
  
Y = mail\_data['Category']

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 ü 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: Message, Length: 5572, dtype: object

print(Y)

0 1  
1 1  
2 0  
3 1  
4 1  
 ..  
5567 0  
5568 1  
5569 1  
5570 1  
5571 1  
Name: Category, Length: 5572, dtype: object

Splitting the data into training data & test 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)

(5572,)  
(4457,)  
(1115,)

Feature Extraction

from sklearn.feature\_extraction.text import TfidfVectorizer  
  
# Create a TF-IDF vectorizer  
feature\_extraction = TfidfVectorizer(min\_df=1, stop\_words='english', lowercase=True)  
  
# Transform the training and test data  
X\_train\_features = feature\_extraction.fit\_transform(X\_train)  
X\_test\_features = feature\_extraction.transform(X\_test)  
  
# Convert Y\_train and Y\_test values to integers  
Y\_train = Y\_train.astype('int')  
Y\_test = Y\_test.astype('int')

print(X\_train)

3075 Don know. I did't msg him recently.  
1787 Do you know why god created gap between your f...  
1614 Thnx dude. u guys out 2nite?  
4304 Yup i'm free...  
3266 44 7732584351, Do you want a New Nokia 3510i c...  
 ...   
789 5 Free Top Polyphonic Tones call 087018728737,...  
968 What do u want when i come back?.a beautiful n...  
1667 Guess who spent all last night phasing in and ...  
3321 Eh sorry leh... I din c ur msg. Not sad alread...  
1688 Free Top ringtone -sub to weekly ringtone-get ...  
Name: Message, Length: 4457, dtype: object

print(X\_train\_features)

(0, 5413) 0.6198254967574347  
 (0, 4456) 0.4168658090846482  
 (0, 2224) 0.413103377943378  
 (0, 3811) 0.34780165336891333  
 (0, 2329) 0.38783870336935383  
 (1, 4080) 0.18880584110891163  
 (1, 3185) 0.29694482957694585  
 (1, 3325) 0.31610586766078863  
 (1, 2957) 0.3398297002864083  
 (1, 2746) 0.3398297002864083  
 (1, 918) 0.22871581159877646  
 (1, 1839) 0.2784903590561455  
 (1, 2758) 0.3226407885943799  
 (1, 2956) 0.33036995955537024  
 (1, 1991) 0.33036995955537024  
 (1, 3046) 0.2503712792613518  
 (1, 3811) 0.17419952275504033  
 (2, 407) 0.509272536051008  
 (2, 3156) 0.4107239318312698  
 (2, 2404) 0.45287711070606745  
 (2, 6601) 0.6056811524587518  
 (3, 2870) 0.5864269879324768  
 (3, 7414) 0.8100020912469564  
 (4, 50) 0.23633754072626942  
 (4, 5497) 0.15743785051118356  
 : :  
 (4454, 4602) 0.2669765732445391  
 (4454, 3142) 0.32014451677763156  
 (4455, 2247) 0.37052851863170466  
 (4455, 2469) 0.35441545511837946  
 (4455, 5646) 0.33545678464631296  
 (4455, 6810) 0.29731757715898277  
 (4455, 6091) 0.23103841516927642  
 (4455, 7113) 0.30536590342067704  
 (4455, 3872) 0.3108911491788658  
 (4455, 4715) 0.30714144758811196  
 (4455, 6916) 0.19636985317119715  
 (4455, 3922) 0.31287563163368587  
 (4455, 4456) 0.24920025316220423  
 (4456, 141) 0.292943737785358  
 (4456, 647) 0.30133182431707617  
 (4456, 6311) 0.30133182431707617  
 (4456, 5569) 0.4619395404299172  
 (4456, 6028) 0.21034888000987115  
 (4456, 7154) 0.24083218452280053  
 (4456, 7150) 0.3677554681447669  
 (4456, 6249) 0.17573831794959716  
 (4456, 6307) 0.2752760476857975  
 (4456, 334) 0.2220077711654938  
 (4456, 5778) 0.16243064490100795  
 (4456, 2870) 0.31523196273113385

Training the Model

Logistic Regression

model = LogisticRegression()

# training the Logistic Regression model with the training data  
model.fit(X\_train\_features, Y\_train)

LogisticRegression()

Evaluating the trained 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\_data)

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

Accuracy on training data : 0.9670181736594121

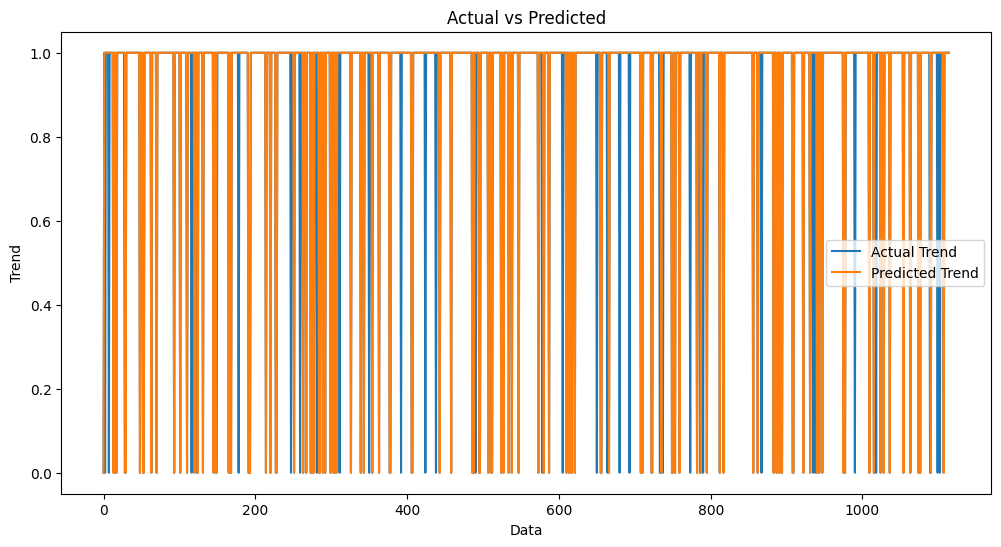
# prediction on test data  
  
prediction\_on\_test\_data = model.predict(X\_test\_features)  
accuracy\_on\_test\_data = accuracy\_score(Y\_test, prediction\_on\_test\_data)

print('Accuracy on test data : ',accuracy\_on\_test\_data)

Accuracy on test data : 0.9659192825112107

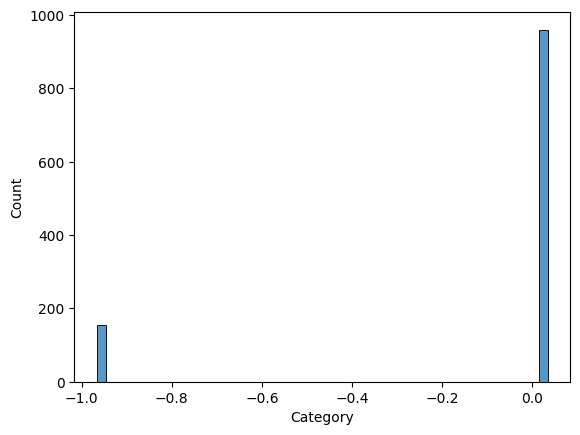
import matplotlib.pyplot as plt  
plt.figure(figsize=(12,6))  
plt.plot(np.arange(len(Y\_test)), Y\_test, label='Actual Trend')  
plt.plot(np.arange(len(Y\_test)), prediction\_on\_test\_data, label='Predicted Trend')  
plt.xlabel('Data')  
plt.ylabel('Trend')  
plt.legend()  
plt.title('Actual vs Predicted')

Text(0.5, 1.0, 'Actual vs Predicted')



import seaborn as sns  
sns.histplot((Y\_test-accuracy\_on\_test\_data), bins=50)

<Axes: xlabel='Category', ylabel='Count'>



Building a Predictive System

input\_mail = ["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 wonderful and a blessing at all times"]  
  
# convert text to feature vectors  
input\_data\_features = feature\_extraction.transform(input\_mail)  
  
# making prediction  
  
prediction = model.predict(input\_data\_features)  
print(prediction)  
  
  
if (prediction[0]==1):  
 print('Ham mail')  
  
else:  
 print('Spam mail')

[1]  
Ham mail

input\_mail = ["Urgent UR awarded a complimentary trip to EuroDisinc Trav, Aco&Entry41 Or Â£1000. To claim txt DIS to 87121 18+6\*Â£1.50(moreFrmMob. ShrAcomOrSglSuplt)10, LS1 3AJ"]  
  
# convert text to feature vectors  
input\_data\_features = feature\_extraction.transform(input\_mail)  
  
# making prediction  
  
prediction = model.predict(input\_data\_features)  
print(prediction)  
  
  
if (prediction[0]==1):  
 print('Ham mail')  
  
else:  
 print('Spam mail')

[0]  
Spam mail

**Conclusion :**

* + In this journey through the development of a spam classifier model, we've explored the intricacies of addressing the pervasive problem of unsolicited and potentially harmful messages in digital communication. This model represents a critical component in maintaining the integrity of communication channels, protecting users from malicious content, and enhancing overall online experiences.
  + In conclusion, the spam classifier model we've explored is not just a technical tool; it's a sentinel that guards the gates of digital communication. Its development, deployment, and continuous improvement are critical in the ongoing battle against spam. By building and maintaining effective spam classifiers, individuals and organizations can contribute to a safer, more secure, and more enjoyable online experience for all.