# EMAIL SPAM DETECTION WITH MACHINE LEARNING

May 15, 2023

## 1 EMAIL SPAM DETECTION WITH MACHINE LEARNING

Dataset Information

The "spam" concept is diverse: advertisements for products/web sites, make money fast schemes, chain letters, pornography…

The SMS Spam Collection is a set of SMS tagged messages that have been collected for SMS Spam research. It contains one set of SMS messages in English of 5,572 messages, tagged according being ham (legitimate) or spam.

Attributes

Email Messages Label (spam/ham)

[1]:
```
#Import modules
import pandas as pd
import numpy as np
import nltk
import re
from nltk.corpus import stopwords
```

[2]:
```
df = pd.read_csv("C:/Users/MyPc/Downloads/task4/spam.csv",␣
↪encoding=('ISO-8859-1'), low_memory =False)
```

[3]:
```
df.head()
```

[3]:
```
     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

  Unnamed: 3 Unnamed: 4
0        NaN        NaN
1        NaN        NaN
2        NaN        NaN
3        NaN        NaN
4        NaN        NaN
```

[4]:
```
# get necessary columns for processing
df = df[['v2', 'v1']]
# df.rename(columns={'v2': 'messages', 'v1': 'label'}, inplace=True)
df = df.rename(columns={'v2': 'messages', 'v1': 'label'})
df.head()
```

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

## 2 Preprocessing the dataset

[5]:
```
# check for null values
df.isnull().sum()
```

[5]:
```
messages    0
label       0
dtype: int64
```

[6]:
```
df.shape
```

[6]:
```
(5572, 2)
```

[7]:
```
df.columns
```

[7]:
```
Index(['messages', 'label'], dtype='object')
```

[8]:
```
df.drop_duplicates(inplace=True)
print(df.shape)
```

```
(5169, 2)
```

[9]:
```
# download the stopwords package
nltk.download("stopwords")
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\MyPc\AppData\Roaming\nltk_data…
[nltk_data]   Package stopwords is already up-to-date!
```

[9]:
```
True
```

[10]:
```
STOPWORDS = set(stopwords.words('english'))

def clean_text(text):
    # convert to lowercase
    text = text.lower()
    # remove special characters
    text = re.sub(r'[^0-9a-zA-Z]', ' ', text)
    # remove extra spaces
    text = re.sub(r'\s+', ' ', text)
    # remove stopwords
    text = " ".join(word for word in text.split() if word not in STOPWORDS)
    return text
```

[11]:
```
# clean the messages
df['clean_text'] = df['messages'].apply(clean_text)
df.head()
```

[11]:
```
                                            messages label  \
0  Go until jurong point, crazy.. Available only …   ham
1                      Ok lar… Joking wif u oni…   ham
2  Free entry in 2 a wkly comp to win FA Cup fina…  spam
3  U dun say so early hor… U c already then say…   ham
4  Nah I don't think he goes to usf, he lives aro…   ham

                                         clean_text
0  go jurong point crazy available bugis n great …
1                            ok lar joking wif u oni
2  free entry 2 wkly comp win fa cup final tkts 2…
3                u dun say early hor u c already say
4             nah think goes usf lives around though
```

## 3 Input Split

[12]:
```
X = df['clean_text']
y = df['label']
```

## 4 Model Training

[13]:
```
from sklearn.pipeline import Pipeline
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.metrics import classification_report
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer,␣
 ↪TfidfTransformer

def classify(model, X, y):
    # train test split
    x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.25,␣
 ↪random_state=42, shuffle=True, stratify=y)
    # model training
    pipeline_model = Pipeline([('vect', CountVectorizer()),
                               ('tfidf', TfidfTransformer()),
                               ('clf', model)])
    pipeline_model.fit(x_train, y_train)

    print('Accuracy:', pipeline_model.score(x_test, y_test)*100)

#     cv_score = cross_val_score(model, X, y, cv=5)
#     print("CV Score:", np.mean(cv_score)*100)
    y_pred = pipeline_model.predict(x_test)
    print(classification_report(y_test, y_pred))
```

[14]:
```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
classify(model, X, y)
```

```
Accuracy: 95.7463263727765
              precision    recall  f1-score   support

         ham       0.95      1.00      0.98      1130
        spam       0.99      0.67      0.80       163

    accuracy                           0.96      1293
   macro avg       0.97      0.83      0.89      1293
weighted avg       0.96      0.96      0.95      1293
```

[15]:
```
from sklearn.naive_bayes import MultinomialNB
model = MultinomialNB()
classify(model, X, y)
```

```
Accuracy: 96.13302397525135
              precision    recall  f1-score   support

         ham       0.96      1.00      0.98      1130
        spam       1.00      0.69      0.82       163

    accuracy                           0.96      1293
   macro avg       0.98      0.85      0.90      1293
weighted avg       0.96      0.96      0.96      1293
```

[16]:
```
from sklearn.svm import SVC
model = SVC(C=3)
classify(model, X, y)
```

```
Accuracy: 97.6798143851508
              precision    recall  f1-score   support
```

```
         ham       0.97      1.00      0.99      1130
        spam       0.99      0.82      0.90       163

    accuracy                           0.98      1293
   macro avg       0.98      0.91      0.94      1293
weighted avg       0.98      0.98      0.98      1293
```

[17]:
```python
from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier()
classify(model, X, y)
```

```
Accuracy: 97.7571539056458
              precision    recall  f1-score   support

         ham       0.98      1.00      0.99      1130
        spam       0.99      0.83      0.90       163

    accuracy                           0.98      1293
   macro avg       0.98      0.91      0.95      1293
weighted avg       0.98      0.98      0.98      1293
```