

[Open in app](#)

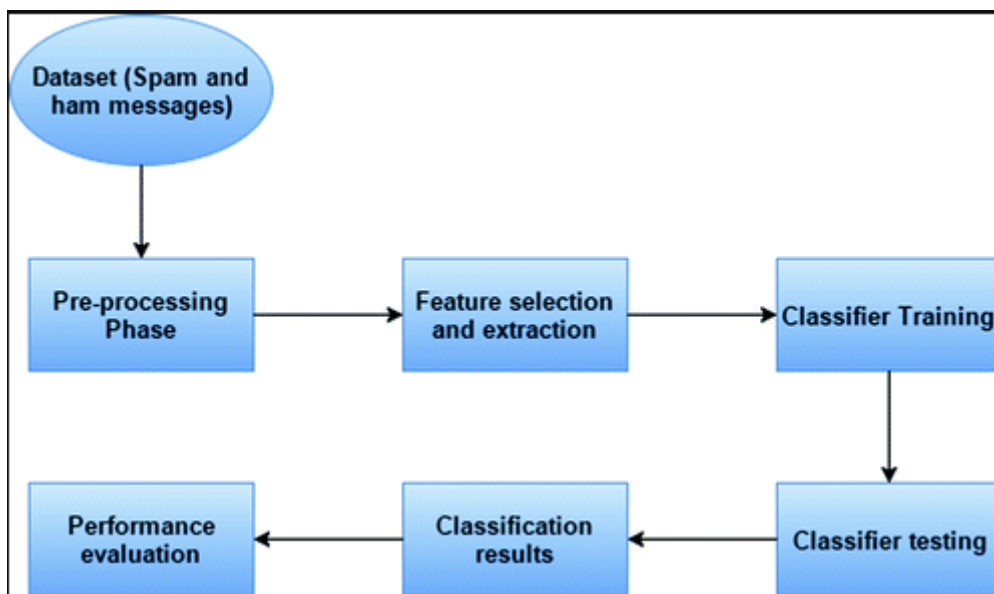
Nidhish Gandhi

1 Follower About

Pr-12. Mini Project- SMS Spam Detection Analysis



Nidhish Gandhi · Just now



Dataset Information

The “spam” concept is diverse: advertisements for products/web sites, make money fast schemes, chain letters, banking...

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 5,574 messages, tagged according to being ham (legitimate) or spam.

Attributes

- SMS Messages

- Label (spam/ham)

Import modules and loading dataset

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

```
df = pd.read_csv('spam.csv', encoding='latin-1')
df.head()
```

	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

```
[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()
```

	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

Preprocessing the dataset

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

```
messages    0
label       0
dtype: int64
```

```
[8] import nltk
nltk.download('stopwords')
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

```

```

[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data] Unzipping corpora/stopwords.zip.

```

```

[9] # clean the messages
df['clean_text'] = df['messages'].apply(clean_text)
df.head()

```

	messages	label	clean_text
0	Go until jurong point, crazy.. Available only ...	ham	go jurong point crazy available bugis n great ...
1	Ok lar... Joking wif u oni...	ham	ok lar joking wif u oni
2	Free entry in 2 a wkly comp to win FA Cup fina...	spam	free entry 2 wkly comp win fa cup final tkts 2...
3	U dun say so early hor... U c already then say...	ham	u dun say early hor u c already say
4	Nah I don't think he goes to usf, he lives aro...	ham	nah think goes usf lives around though

```

10] X = df['clean_text']
y = df['label']

```

Model Training

```

[11] 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))

from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
classify(model, X, y)

```

Checking accuracy for different models:

Logistic Regression and SVC

```
[11] Accuracy: 96.8413496051687
```

	precision	recall	f1-score	support
ham	0.97	1.00	0.98	1206
spam	0.99	0.77	0.87	187
accuracy			0.97	1393
macro avg	0.98	0.88	0.92	1393
weighted avg	0.97	0.97	0.97	1393

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

Accuracy: 98.27709978463747

	precision	recall	f1-score	support
ham	0.98	1.00	0.99	1206
spam	1.00	0.87	0.93	187
accuracy			0.98	1393
macro avg	0.99	0.94	0.96	1393
weighted avg	0.98	0.98	0.98	1393

RandomForestClassifier

```
from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier()
classify(model, X, y)
```

Accuracy: 97.20028715003589

	precision	recall	f1-score	support
ham	0.97	1.00	0.98	1206
spam	1.00	0.79	0.88	187
accuracy			0.97	1393
macro avg	0.98	0.90	0.93	1393
weighted avg	0.97	0.97	0.97	1393

So by using different models we can get better accuracy. So in this project I have loaded the dataset and done preprocessing. Afterwards I have trained using different models.

[About](#) [Write](#) [Help](#) [Legal](#)

Get the Medium app



