In [336…
```
import pandas as pd
import numpy as np
import nltk
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
import string



df = pd.read_csv("train.csv")
df['email'] = df['email'].str.lower()
df.rename(columns={'label': 'spam'}, inplace=True)
df['spam'] = df['spam'].apply(lambda x: 1 if x == 'spam' else 0)
df.head(5)
```

Out[336]:

	email	spam
0	minintk 2002-08-16 __ __ _2002-08-16 _ ____...	0
1	tags reveal if frozen food is rottenurl: http:...	0
2	personal 75% off to hibody@csmining.org. pfi...	1
3	from fork-admin@xent.com mon sep 23 22:47:38 ...	0
4	re: anolther sequence related tracebacki have ...	0

In [337…
```
df.duplicated().sum()
```

Out[337]:
```
136
```

In [338…
```
df.drop_duplicates(inplace=True)
```

In [339…
```
df.isnull().sum()
```

Out[339]:
```
email    0
spam     0
dtype: int64
```

In [340…
```
df.shape
```

Out[340]:
```
(7364, 2)
```

In [341…
```
nltk.download('stopwords')
```

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

Out[341]:
```
True
```

In [342…
```
def process_text(text):
    nopunc = [char for char in text if char not in string.punctuation]
    nopunc = ''.join(nopunc)

    clean_words = [word for word in nopunc.split() if word.lower() not in stopwords.word

    #PorterStemmer seemed to worsen results

    return clean_words
```

In [343…
```
df['email'].head().apply(process_text)
```

Out[343]:
```
0    [minintk, 20020816, 20020816, join, mail, empt...
1    [tags, reveal, frozen, food, rottenurl, httpww...
2    [personal, 75, hibodycsminingorg, pfizer, webl...
3    [forkadminxentcom, sep, 23, 224738, 2002, retu...
4    [anolther, sequence, related, tracebacki, patc...
Name: email, dtype: object
```

In [344…
```
from sklearn.feature_extraction.text import CountVectorizer

email_bow = CountVectorizer(analyzer = process_text).fit_transform(df['email'])
```

In [345…
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(email_bow, df['spam'], test_size = 0
```

In [346…
```
email_bow.shape
```

Out[346]:
```
(7364, 119078)
```

In [352…
```
from sklearn.svm import SVC
from sklearn.naive_bayes import MultinomialNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import BaggingClassifier
from sklearn.ensemble import GradientBoostingClassifier
```

In [353…
```
svc = SVC(kernel='rbf', gamma='scale', class_weight='balanced', C = 3)
knc = KNeighborsClassifier(n_neighbors=8, weights='uniform', p=2)
mnb = MultinomialNB(alpha = 1.1)
dtc = DecisionTreeClassifier()
bc = BaggingClassifier(n_estimators=70, n_jobs=-1)
gbdt = GradientBoostingClassifier(n_estimators=250,random_state=2)
```

In [354…
```
clfs = {
    'SVC' : svc,
    'KN' : knc,
    'NB': mnb,
    'DT': dtc,
    'BgC': bc,
    'GBDT':gbdt,
}
```

In [355…
```
def train_classifier(clf,X_train,y_train,X_test,y_test):
    clf.fit(X_train,y_train)
    y_pred = clf.predict(X_test)
    accuracy = accuracy_score(y_test,y_pred)
    precision = precision_score(y_test,y_pred)

    return accuracy,precision
```

In [356…
```
accuracy_scores = []
precision_scores = []

for name,clf in clfs.items():

    current_accuracy,current_precision = train_classifier(clf, X_train,y_train,X_test,y_

    print("For ",name)
    print("Accuracy - ",current_accuracy)
    print("Precision - ",current_precision)
```

```
    accuracy_scores.append(current_accuracy)
    precision_scores.append(current_precision)
```

```
For  SVC
Accuracy -  0.96673455532926
Precision -  0.9121212121212121
```

```
C:\Users\savva\anaconda3\lib\site-packages\sklearn\neighbors\_classification.py:228: Fut
ureWarning: Unlike other reduction functions (e.g. `skew`, `kurtosis`), the default beha
vior of `mode` typically preserves the axis it acts along. In SciPy 1.11.0, this behavio
r will change: the default value of `keepdims` will become False, the `axis` over which
the statistic is taken will be eliminated, and the value None will no longer be accepte
d. Set `keepdims` to True or False to avoid this warning.
  mode, _ = stats.mode(_y[neigh_ind, k], axis=1)
```

```
For  KN
Accuracy -  0.8316361167684997
Precision -  0.5743380855397149
For  NB
Accuracy -  0.9742023082145281
Precision -  0.9380804953560371
For  DT
Accuracy -  0.9592668024439919
Precision -  0.8942598187311178
For  BgC
Accuracy -  0.9769178547182621
Precision -  0.9283582089552239
For  GBDT
Accuracy -  0.9592668024439919
Precision -  0.9745454545454545
```

In [357…
```
performance_df = pd.DataFrame({'Algorithm':clfs.keys(),'Accuracy':accuracy_scores,'Preci
performance_df
```

Out[357]:

	Algorithm	Accuracy	Precision
5	GBDT	0.959267	0.974545
2	NB	0.974202	0.938080
4	BgC	0.976918	0.928358
0	SVC	0.966735	0.912121
3	DT	0.959267	0.894260
1	KN	0.831636	0.574338

In [358…
```
gbdt.fit(X_train,y_train)
y_pred = gbdt.predict(X_test)
confusion_matrix(y_test,y_pred)
```

Out[358]:
```
array([[1145,    7],
       [  53,  268]], dtype=int64)
```

In [363…
```
from sklearn.model_selection import cross_val_score
cv = cross_val_score(gbdt, X_train, y_train, cv=5)
```

In [365…
```
cv.mean()
```

Out[365]:
```
0.9568821092376348
```

In [ ]: