```
import numpy as np
import pandas as pd
import nltk
```

```
df_sms=pd.read_csv('spam.csv', encoding="ISO-8859-1")
```

```
df_sms.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

Next steps: Generate code with df_sms View recommended plots

```
df_sms = df_sms.drop([ "Unnamed: 2","Unnamed: 3","Unnamed: 4"], axis=1)
df_sms = df_sms.rename(columns={"v1":"label", "v2":"sms-text"})
df_sms.head()
```

	label	sms-text
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...

Next steps: Generate code with df_sms View recommended plots

```
df_sms.label.value_counts()
```

```
label
ham     4825
spam     747
Name: count, dtype: int64
```

```
df_sms.head()
```

	label	sms-text
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...

Next steps: Generate code with df_sms View recommended plots

```
df_sms.describe()
```

	length
count	5572.000000
mean	80.118808
std	59.690841
min	2.000000
25%	36.000000
50%	61.000000
75%	121.000000
max	910.000000

```
df_sms['length']= df_sms['sms-text'].apply(len)
```

```
df_sms['length']
```

```
0       111
1        29
2       155
3        49
4        61
        ...
5567    161
5568     37
5569     57
5570    125
5571     26
Name: length, Length: 5572, dtype: int64
```

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test= train_test_split(df_sms['sms-text'],df_sms['label'],test_size=0.20,random_state=1)
```

```
from sklearn.feature_extraction.text import CountVectorizer
```

```
from sklearn.naive_bayes import MultinomialNB
m = MultinomialNB()
```

```
vectorizer = CountVectorizer()
X_train_vec = vectorizer.fit_transform(X_train)
X_test_vec = vectorizer.transform(X_test)  # Note the use of transform, not fit_transform
```

```
# Initialize and train the MultinomialNB model
m.fit(X_train_vec, y_train)

# Now you can make predictions on the test data
predictions = m.predict(X_test_vec)
predictions
```

```
array(['ham', 'ham', 'ham', ..., 'ham', 'ham', 'ham'], dtype='<U4')
```

```
from sklearn.metrics import accuracy_score,precision_score,recall_score,f1_score
```

```
print('Accuracy score : {}'.format(accuracy_score(y_test, predictions)))
```

```
Accuracy score : 0.9847533632286996
```

```
print('Precision_score : {}'.format(precision_score(y_test, predictions,pos_label='spam')))
```

```
Precision_score : 0.9420289855072463
```

```
print('Recall score : {}'.format(recall_score(y_test, predictions,pos_label='spam')))
```

```
Recall score : 0.935251798561151
```