```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
from sklearn.model_selection import train_test_split
from sklearn import ensemble
from sklearn.ensemble import BaggingClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import AdaBoostClassifier
from sklearn import svm
from sklearn.svm import SVC
# SMSSpamCollection file уншиж авах хэсэг
df = pd.read_table('SMSSpamCollection',
                   sep='\t',
                   header=None,
                   names=['label', 'sms_message'])

print('Мөрийн тоо:', df.shape[0])
print('Баганын тоо:', df.shape)
df.head()
```

```
No.of rows: 5572
No.of columns: (5572, 2)
```

	label	sms_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...

Data Preprocessing

```
# ham болон spam -р эхэлсэн хэсгүүдийг 0, 1 болгон хөрвүүлнэ.
df['label'] = df.label.map({'ham':0, 'spam':1})
df.head()
```

```
(5572, 2)
```

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

```
# харгалзах утгуудыг оноож өгнө. Жишээ нь: X_train хэсэгт sms_message баганыг г.м
X_train, X_test, y_train, y_test = train_test_split(df['sms_message'],
                                                    df['label'],
                                                    random_state=1,
                                                   test_size=0.3)

print('Мөр бичлэгийн тоо: {}'.format(df.shape[0]))
print('Сургалтын дата доторх өгөгдлийн тоо: {}'.format(X_train.shape[0]))
print('Өгөгдлийг турших өгөгдлийн цэгийн тоо : {}'.format(X_test.shape[0]))
```

```
Total no. of data points: 5572
No. of data points in training dataset: 3900
No. of data points in testing dataset: 1672
```

Үгсийн багцыг мэдээллийн санд ашиглах

Одоо бид өгөгдлөө хуваасан тул бидний дараагийн зорилго бол үгсийг хүссэн матрицын формат руу хөрвүүлэх явдал юм. Үүнийг хийхийн тулд бид CountVectorizer () -ийг ашиглах болно. Энд хоёр алхамыг анхаарч үзэх хэрэгтэй.

- Нэгдүгээрт, бид сургалтын өгөгдлөө (X_train) CountVectorizer () -т багтааж, матрицыг буцааж өгөх ёстой.
- Хоёрдугаарт, бид матрицыг буцаахын тулд тестийн өгөгдлөө (X_test) өөрчлөх хэрэгтэй.

X_train бол манай мэдээллийн бааз дахь 'sms_message' баганад зориулсан сургалтын өгөгдөл бөгөөд үүнийг ашиглан загвараа сургах болно. X_test бол 'sms_message' баганын туршилтын өгөгдөл бөгөөд энэ нь урьдчилан таамаглахад ашиглагдах өгөгдөл юм (матрицад шилжсэний дараа). Дараа нь бид эдгээр таамаглалыг дараагийн шатанд y_test-тэй харьцуулах болно.

```
# CountVectorizer аргыг ашиглах
count_vector = CountVectorizer()

# Сургалтын өгөгдлийг тохируулаад дараа нь матрицыг буцаана (Текстийн өгөгдлийн багц (мессеж) -ийг үгийн давтамжийн матриц бо
training_data = (count_vector.fit_transform(X_train)).toarray()
print(training_data)
print(training_data.shape)
```

```
[[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 ...
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]]
(3900, 7155)
```

Туршилтын өгөгдлийг хувиргаж, матрицыг буцаана.

- Бид тестийн өгөгдлийг CountVectorizer () -д тохируулахгүй байгааг анхаарна уу.

```
testing_data = (count_vector.transform(X_test)).toarray()
print(testing_data)
```

```
print(testing_data.shape)
```

```
[[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 ...
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]]
(1672, 7155)
```

Сургалтын модел

```
# Бидний загварыг бэлэн болгох
naive_bayes = MultinomialNB()
# Сургалтын өгөгдөлд манай загварыг тохируулах
naive_bayes.fit(training_data, y_train)
```

```
MultinomialNB(alpha=1.0, class_prior=None, fit_prior=True)
```

Урьдчилан таамаглах(Predict)

```
# Урьдчилан таамаглах тест өгөгдөл
predictions = naive_bayes.predict(testing_data)
```

```
print('Нарийвчлалын(Accuracy) дүн: ', format(accuracy_score(y_test, predictions)))
print('Нарийвчлал(Precision) дүн: ', format(precision_score(y_test, predictions)))
print('Дахин санах: ', format(recall_score(y_test, predictions)))
print('F1 дүн: ', format(f1_score(y_test, predictions)))
```

```
Accuracy score:  0.9874401913875598
Precision score:  0.9728506787330317
Recall score:  0.9347826086956522
F1 score:  0.9534368070953437
```

```
#  200 сул суралцагч (n_estimators) болон бусад бүх зүйлийг анхдагч утга болгон ашиглах
```

```
Bag = BaggingClassifier(n_estimators=200)

# RandomForestClassifier програмыг ашиглан:
# 200 сул суралцагч (n_estimators) болон бусад бүх зүйлийг анхдагч утга болгон ашиглах

RF = RandomForestClassifier(n_estimators=200)

# AdaBoostClassifier-ийг дараах байдлаар ашиглана:
# 300 сул суралцагчтай (n_estimators) ба суралцах түвшин 0.2 байна

ADBoost = AdaBoostClassifier(n_estimators=300, learning_rate=0.2)

# Анхдагч параметрийн утгатай SVM-ийг тохируулах:
SVM = SVC()
```

```
# BaggingClassifier-ээ сургалтын өгөгдөлд тохируулна
Bag.fit(training_data, y_train)
```

```
BaggingClassifier(base_estimator=None, bootstrap=True, bootstrap_features=False,
                  max_features=1.0, max_samples=1.0, n_estimators=200,
                  n_jobs=None, oob_score=False, random_state=None, verbose=0,
                  warm_start=False)
```

```
# RandomForestClassifier програмаа сургалтын өгөгдөлд тохируулна
RF.fit(training_data, y_train)
```

```
RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                       criterion='gini', max_depth=None, max_features='auto',
                       max_leaf_nodes=None, max_samples=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, n_estimators=200,
                       n_jobs=None, oob_score=False, random_state=None,
                       verbose=0, warm_start=False)
```

```
# AdaBoostClassifier програмаа сургалтын өгөгдөлд тохируулна
```

```
ADBoost.fit(training_data, y_train)
```

```
AdaBoostClassifier(algorithm='SAMME.R', base_estimator=None, learning_rate=0.2,
                   n_estimators=300, random_state=None)
```

```
# SVM-ээ сургалтын өгөгдөлд нийцүүлэх
SVM.fit(training_data, y_train)
```

```
SVC(C=1.0, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

```
# Туршилтын өгөгдөл дээр BaggingClassifier ашиглан урьдчилан таамаглах
y_Bag = Bag.predict(testing_data)

# Туршилтын өгөгдөлд RandomForestClassifier ашиглан урьдчилан таамаглах
y_RF = RF.predict(testing_data)

# Туршилтын өгөгдөлд AdaBoostClassifier ашиглан урьдчилан таамаглах
y_ADBoost = ADBoost.predict(testing_data)

# Туршилтын өгөгдөлд SVM ашиглан урьдчилан таамаглах
y_SVM = SVM.predict(testing_data)
```

```
def print_metrics(y_true, preds, model_name=None):
    '''
    Ороолт:
    y_true = y_test = Өгөгдлийн баазад үнэн байх y утга (numpy array or pandas series)
    preds = predicted value (y_Bag, y_RF, y_ADBoost) = the predictions for those values from some model (numpy array or panda
    model_name = BaggingClassifier/RandomForestClassifier/AdaBoostClassifier = a name associated with the model if you would

    Гаралт:
    accuracy(нарийвчлал), precision(нарийвчлал), recall(дахин холбогдох), болон F1 оноог хэвлэнэ
    '''
    if model_name == None:
        print('Accuracy score: ', format(accuracy_score(y_true, preds)))
```

```
        print('Precision score: ', format(precision_score(y_true, preds)))
        print('Recall score: ', format(recall_score(y_true, preds)))
        print('F1 score: ', format(f1_score(y_true, preds)))
        print('\n\n')

    else:
        print('Accuracy score for ' + model_name + ' :' , format(accuracy_score(y_true, preds)))
        print('Precision score ' + model_name + ' :', format(precision_score(y_true, preds)))
        print('Recall score ' + model_name + ' :', format(recall_score(y_true, preds)))
        print('F1 score ' + model_name + ' :', format(f1_score(y_true, preds)))
        print('\n\n')
```

```
# Print Bagging scores
print_metrics(y_true=y_test, preds=y_Bag, model_name="Bagging Classifier")

# Print Random Forest scores
print_metrics(y_true=y_test, preds=y_RF, model_name="Randomforest Classifier")

# Print AdaBoost scores
print_metrics(y_true=y_test, preds=y_ADBoost, model_name="AdaBoost Classifier")

# Naive Bayes Classifier scores
print_metrics(y_true=y_test, preds=predictions, model_name="Naive Bayes Classifier")

# SVM Classifier scores

print('Accuracy score for SVM :' , format(accuracy_score(y_test, y_SVM)))
print('Precision score for SVM :', format(precision_score(y_test, y_SVM, average= 'weighted', labels=np.unique(y_SVM))))
print('Recall score for SVM :', format(recall_score(y_test, y_SVM, average= 'weighted', labels=np.unique(y_SVM))))
print('F1 score for SVM :', format(f1_score(y_test, y_SVM, average= 'weighted', labels=np.unique(y_SVM))))
```

```
Accuracy score for Bagging Classifier : 0.9736842105263158
Precision score Bagging Classifier : 0.9227272727272727
Recall score Bagging Classifier : 0.8826086956521739
F1 score Bagging Classifier : 0.9022222222222223



Accuracy score for Randomforest Classifier : 0.9742822966507177
```

```
Precision score Randomforest Classifier : 0.9895287958115183
Recall score Randomforest Classifier : 0.8217391304347826
F1 score Randomforest Classifier : 0.8978622327790974



Accuracy score for AdaBoost Classifier : 0.9760765550239234
Precision score AdaBoost Classifier : 0.9611650485436893
Recall score AdaBoost Classifier : 0.8608695652173913
F1 score AdaBoost Classifier : 0.9082568807339451



Accuracy score for Naive Bayes Classifier : 0.9874401913875598
Precision score Naive Bayes Classifier : 0.9728506787330317
Recall score Naive Bayes Classifier : 0.9347826086956522
F1 score Naive Bayes Classifier : 0.9534368070953437



Accuracy score for SVM : 0.9766746411483254
Precision score for SVM : 0.97674250981534
Recall score for SVM : 0.9766746411483254
F1 score for SVM : 0.9759559119223289
```