Ensemble Machine Learning application with Boosting technique

MNIST case study :

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn. ensemble import AdaBoostClassifier

data = pd.read_csv('mnist.csv')

df_x = data.iloc[:,1:]  # Labels
df_y = data.iloc[:,0]   # Pixels

x_train, x_test, y_train, y_test = train_test_split(df_x, df_y, test_size=0.2,
random_state=4)

obj = DecisionTreeClassifier(___, ___, ___);
adb = AdaBoostClassifier(obj,n_estimators =___, learning_rate = __);
adb = AdaBoostClassifier(DecisionTreeClassifier(),n_estimators = 100 , learning_rate =
1)
adb.fit(x_train,y_train)

print("Testing accuracy using bagging classifier : ",adb.score(x_test,y_test)*100)

print("Training accuracy using bagging classifier : ",adb.score(x_train,y_train)*100)
```


Iris Case study :

```
from sklearn.ensemble import AdaBoostClassifier
from sklearn import datasets
# Import train_test_split function
from sklearn.model_selection import train_test_split
#Import scikit-learn metrics module for accuracy calculation
from sklearn import metrics

# Load data
iris = datasets.load_iris()
X = iris.data
```

Iris Case study :

```
from sklearn.ensemble import AdaBoostClassifier
from sklearn import datasets
# Import train_test_split function
from sklearn.model_selection import train_test_split
#Import scikit-learn metrics module for accuracy calculation
from sklearn import metrics

# Load data
iris = datasets.load_iris()
X = iris.data
y = iris.target

# Split dataset into training set and test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3) # 70% training
and 30% test

# Create adaboost classifer object
abc = AdaBoostClassifier(n_estimators=50,
                         learning_rate=1)
# Train Adaboost Classifer
model = abc.fit(X_train, y_train)

#Predict the response for test dataset
y_pred = model.predict(X_test)

print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```