```
#1. WAP to Implement Splitting data using
  # a.   Holdout,
  # b.   K Fold,
  #c.    Stratified K Fold,
  # d.   Leave-One-Out (LOO),
  # e.   Leave-P-Out (LPO),
  # f.    Shuffle Split. Compare the Algorithms .
#============================================================


#Holdout
#-----------------------------------------------------

import numpy as np
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt

X = np.random.rand(100, 2)
y = range(100)

X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=0.6, test_size=0.4)
print(len(X_train), len(X_test))
print(len(y_train), len(y_test))

print("Training set:")
print(X_train,)
plt.scatter(X_train, X_train)
plt.show()

print("test set:")
print( y_test)
plt.scatter(y_test, y_test)
plt.show()

#----------------------------
#K Fold
#-----------------------------------------------------
from sklearn import datasets
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import KFold, cross_val_score

X, y = datasets.load_iris(return_X_y=True)

clf = DecisionTreeClassifier(random_state=42)

k_folds = KFold(n_splits = 5)

scores_kcv = cross_val_score(clf, X, y, cv = k_folds)

print("Cross Validation Scores: ", scores_kcv)
print("Average CV Score: ", scores_kcv.mean())
print("Number of CV Scores used in Average: ", len(scores_kcv))

#-------------------------------------------
#Stratified K Fold
```

```
#-------------------------------------------------
from sklearn import datasets

from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import StratifiedKFold, cross_val_score
X, y = datasets.load_iris(return_X_y=True)
clf = DecisionTreeClassifier(random_state=42)
sk_folds = StratifiedKFold(n_splits = 5)
scores_sk = cross_val_score(clf, X, y, cv = sk_folds)
print("Cross Validation Scores: ", scores_sk)

print("Average CV Score: ", scores_sk.mean())

print("Number of CV Scores used in Average: ", len(scores_sk))

#-------------------------------------------------
#Leave-One-Out (LOO)
#-----------------------------------------------------------
from sklearn import datasets from sklearn.tree import DecisionTreeClassifier from
sklearn.model_selection import LeaveOneOut, cross_val_score
X, y = datasets.load_iris(return_X_y=True)
clf = DecisionTreeClassifier(random_state=42)
loo = LeaveOneOut()
scores_loo = cross_val_score(clf, X, y, cv = loo)
print("Cross Validation Scores: ", scores_loo)
print("Average CV Score: ", scores_loo.mean())
print("Number of CV Scores used in Average: ", len(scores_loo))

#-------------------------------------------------------------------
#Leave-P-Out (LPO)
#-------------------------------------------------------------------
from sklearn import datasets from sklearn.tree import DecisionTreeClassifier from
sklearn.model_selection import LeavePOut, cross_val_score
X, y = datasets.load_iris(return_X_y=True)
clf = DecisionTreeClassifier(random_state=42)
lpo = LeavePOut(p=2)
scores_lpo = cross_val_score(clf, X, y, cv = lpo)
print("Cross Validation Scores: ", scores_lpo)
print("Average CV Score: ", scores_lpo.mean())
print("Number of CV Scores used in Average: ", len(scores_lpo))

#-----------------------------------------
#Shuffle Split
#-----------------------------------------
from sklearn import datasets
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import ShuffleSplit, cross_val_score

X, y = datasets.load_iris(return_X_y=True)

clf = DecisionTreeClassifier(random_state=42)

ss = ShuffleSplit(train_size=0.6, test_size=0.3, n_splits = 5)

scores_shuf = cross_val_score(clf, X, y, cv = ss)
```

```
print("Cross Validation Scores: ", scores_shuf)
print("Average CV Score: ", scores.mean())
print("Number of CV Scores used in Average: ", len(scores_shuf))

#-------------------------------------------------------
pip install texttable
#-----------------------------------------------

# Comparison
#---------------------------------------------------------------------
# Creating an instance of Texttable class
from texttable import Texttable
my_table = Texttable()

# Adding rows to our tabular table

my_table.add_rows(    [["Algorithm", "Score", "Mean", "Number of CV Scores used in
Average"],
                       ["K Fold",scores_kcv, scores_kcv.mean(), len(scores_kcv)],
                       ["Stratified K Fold", scores_sk, scores_sk.mean(), len(scores_sk)],
                         ["Leave-One-Out (LOO)", scores_loo, scores_loo.mean(),
len(scores_loo)],
                         ["Leave-P-Out (LPO)", scores_lpo, scores_lpo.mean(),
len(scores_lpo)],
                       ["Shuffle ", scores_shuf, scores_shuf.mean(), len(scores_shuf)],
 ] )
# Printing Tabulated Data Using draw() function of Texttable Class print(my_table.draw())
```