Housing dataset

Using train_test_split where random state=42

Import libraries

```
import numpy as np
import pandas as pd
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
```

Loading Dataset

```
# Load the Boston Housing dataset
df = pd.read_csv('/content/BostonHousing.csv')
```

```
# Select all columns from the dataframe except the last one and assign them to variable 'X'
X = df.iloc[:, 0:-1]
# Select the last column from the dataframe and assign it to variable 'y'
y = df.iloc[:, -1]
```

Creating a linear regression model

```
# Create a linear regression model
model = LinearRegression()
x_train,x_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=42)
model.fit(x_train,y_train)
y_preds=model.predict(x_test)
print("Average MSE:", mean_squared_error(y_test,y_preds))
```

```
Average MSE: 24.291119474973478
```

Using train_test_split where random state=2

```
X = df.iloc[:,0:-1]
y = df.iloc[:,-1]
```

Creating a linear regression model

```
# Create a linear regression model
model = LinearRegression()
x_train,x_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=2)
model.fit(x_train,y_train)
y_preds=model.predict(x_test)
print("Average MSE:", mean_squared_error(y_test,y_preds))
```

```
Average MSE: 18.49542012244846
```

Leave one-out cross validation

import libraries

```
import pandas as pd
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import LeaveOneOut, cross_val_score
```

```
X = df.iloc[:,0:-1]
y = df.iloc[:,-1]
```

Creating a linear regression model and LeaveOneOut cross validator

```
# Create a linear regression model
model = LinearRegression()
# Create a LeaveOneOut cross-validator
loo = LeaveOneOut()
# Use cross_val_score for the dataset with the model and LOOCV
# This will return the scores for each iteration of LOOCV
scores = cross_val_score(model, X, y, cv=loo, scoring='neg_mean_squared_error')
mse_scores = -scores # Invert the sign of the scores
# Print the mean MSE over all LOOCV iterations
print("Mean MSE:", mse_scores.mean())
```

```
Mean MSE: 23.72574551947613
```

Implementing K-fold cross validation

Import Libraries

```
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import KFold
import pandas as pd


X = df.iloc[:,0:-1]
y = df.iloc[:,-1]
```

Creating a linear regression model and Initialize the KFold parameters

```
# Initialize a Linear Regression model
model = LinearRegression()
# Initialize the KFold parameters
kfold = KFold(n_splits=10, shuffle=True, random_state=2)
# Use cross_val_score on the model and dataset
scores = cross_val_score(model, X, y, cv=kfold, scoring='r2')
print("R2 scores for each fold:", scores)
print("Mean R2 score across all folds:", scores.mean())
```

```
R2 scores for each fold: [0.6958265  0.84231184 0.6873712  0.78049402 0.70296972 0.60709628
 0.5479473  0.71991392 0.71923209 0.81572717]
Mean R2 score across all folds: 0.7118890033406191
```

Implementing Stratified K-fold cross validation

```
from sklearn.datasets import load_iris
from sklearn.model_selection import StratifiedKFold, cross_val_score
from sklearn.linear_model import LogisticRegression
```

Creating a Logistic Regression model and StratifiedKFold object

```
# Load iris dataset
data = load_iris()
X, y = data.data, data.target
# Create a Logistic Regression model
model = LogisticRegression(max_iter=10000, random_state=42)
# Create StratifiedKFold object
skf = StratifiedKFold(n_splits=5, random_state=42, shuffle=True)
# Perform stratified cross validation
scores = cross_val_score(model, X, y, cv=skf, scoring='accuracy')
# Print the accuracy for each fold

print("Accuracies for each fold: ", scores)
print("Mean accuracy across all folds: ", scores.mean())
```

```
Accuracies for each fold:  [1.         0.96666667 0.93333333 1.         0.93333333]
Mean accuracy across all folds:  0.9666666666666668
```