**Assignment 3 (Implementation of various techniques)**

from sklearn.linear\_model import LinearRegression

import matplotlib.pyplot as plt

import pandas as pd

from sklearn.datasets import load\_diabetes

from sklearn.linear\_model import LinearRegression, Ridge

from sklearn.model\_selection import cross\_val\_score

diabetes = load\_diabetes()

lr = LinearRegression(normalize = True)

lr\_scores = cross\_val\_score(lr, diabetes.data, diabetes.target,cv=10)

print(lr\_scores)

lr\_scores.mean()

**Output:**

[0.55614411 0.23056092 0.35357777 0.62190498 0.26587602 0.61819338

0.41815916 0.43515232 0.43436983 0.68568514]

0.4619623619583372

## Ridge

rg = Ridge(0.005, normalize=True)

rg\_scores = cross\_val\_score(rg, diabetes.data, diabetes.target, cv=10)

print(rg\_scores)

rg\_scores.mean()

## RidgeCV for set of alpha values, alpha\_ to extract alpha value

from sklearn.linear\_model import RidgeCV

rg = RidgeCV(alphas=(1.0, 0.1, 0.01, 0.005, 0.0025, 0.001, 0.00025),normalize=True)

rg.fit(diabetes.data, diabetes.target)

rg.alpha\_

rg\_cv\_scores = cross\_val\_score(rg, diabetes.data, diabetes.target, cv=10)

print(rg\_cv\_scores)

rg\_scores.mean()

**Output:**

[0.55014745 0.24000807 0.36373382 0.61657429 0.2695777 0.62172635

0.42346862 0.42813504 0.43117332 0.68303604]

[0.52803256 0.23657595 0.3565488 0.607513 0.2695777 0.62047382

0.42149214 0.43894932 0.43138195 0.6642474 ]

Out[4]:

0.4627580697072979

## Lasso, LassoCV

from sklearn.linear\_model import Lasso, LassoCV

ls = Lasso(alpha=0.005, normalize=True)

ls\_scores = cross\_val\_score(ls, diabetes.data, diabetes.target, cv=10)

print(ls\_scores)

ls\_scores.mean()

from sklearn.linear\_model import LassoCV

ls\_cv = LassoCV(alphas=(1.0,0.1,0.01,0.005,0.0025,0.001,0.00025),normalize=True)

ls\_cv.fit(diabetes.data, diabetes.target)

ls\_cv.alpha\_

ls\_cv\_scores = cross\_val\_score(ls\_cv, diabetes.data, diabetes.target,cv=10)

print(ls\_cv\_scores)

ls\_cv\_scores.mean()

**Output:**

[0.55078146 0.23993097 0.36453647 0.61454396 0.26894036 0.62269952

0.42465645 0.42726522 0.43075777 0.68414508]

[0.51796189 0.23824666 0.35415718 0.59752149 0.27503201 0.62269952

0.41851368 0.42047653 0.42562829 0.68231416]

Out[5]:

0.45525514142746404

## ElasticNet, ElasticNetCV

from sklearn.linear\_model import ElasticNet, ElasticNetCV

en = ElasticNet(alpha=0.001,l1\_ratio=0.8, normalize=True)

en\_scores = cross\_val\_score(en, diabetes.data, diabetes.target, cv=10)

print(en\_scores)

en\_scores.mean()

encv = ElasticNetCV(alphas=(0.1, 0.01,0.005, 0.0025, 0.001),l1\_ratio=(0.1,0.25,0.5,0.75,

0.8), normalize=True)

encv.fit(diabetes.data, diabetes.target)

print(encv.alpha\_)

print(encv.l1\_ratio\_)

encv\_scores = cross\_val\_score(encv, diabetes.data, diabetes.target,cv=10)

print(encv\_scores)

encv\_scores.mean()

**Output:**

[0.53103739 0.24682675 0.38160097 0.60832995 0.2830996 0.62083992

0.43113636 0.43484866 0.43055758 0.6676087 ]

0.001

0.8

[0.52796681 0.24682675 0.38160097 0.60743123 0.28526958 0.61931497

0.43113636 0.43907824 0.43137097 0.66429212]

Out[6]:

0.46342880159848276

## Comparative analysis

#Linear Regression Score

print(lr\_scores.mean())

#Ridge Score

print(rg\_scores.mean())

#Lasso Score

print(ls\_scores.mean())

#ElasticNet Score

print(en\_scores.mean())

**Output:**

0.4619623619583372

0.4627580697072979

0.4628257255366856

0.46358858847836454

import matplotlib.pyplot as plt

import numpy as np

objects = ('Linear','Ridge','Lasso','ElasticNet')

y\_pos = np.arange(len(objects))

p = [0.4619623619583371,0.4627580697072979,0.4628257255366856,0.46358858847836454]

plt.bar(y\_pos, p, align='center',alpha=0.5, color=['hotpink', 'yellow', 'lime','cyan'])

plt.xticks(y\_pos, objects)

plt.ylabel('Accuracy')

plt.title('Linear Models')

ElasticNetCV(alphas=(0.1, 0.01, 0.005, 0.0025, 0.001),

l1\_ratio=(0.1, 0.25, 0.5, 0.75, 0.8), normalize=True)

plt.show()

**Output:**

**A picture containing bar chart

Description automatically generated**