# Lasso and Ridge Regression

**Instructions:**

Please share your answers filled in-line in the word document. Submit code separately wherever applicable.

Please ensure you update all the details:

**Name: Anandakrishnan k v ;;;; Batch ID:**19042021

**Topic: Lasso and Ridge Regression**

**Grading Guidelines:**

**1. An assignment submission is considered complete only when correct and executable code(s) are submitted along with the documentation explaining the method and results. Failing to submit either of those will be considered an invalid submission and will not be considered for evaluation.**

**2. Assignments submitted after the deadline will affect your grades.**

**Grading:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Ans** | **Date** |  |  | **Ans** | **Date** |
| Correct | On time | A | 100 |  |  |
| 80% & above | On time | B | 85 | Correct | Late |
| 50% & above | On time | C | 75 | 80% & above | Late |
| 50% & below | On time | D | 65 | 50% & above | Late |
|  |  | E | 55 | 50% & below |  |
| Copied/No Submission |  | F | 45 |  |  |

* **Grade A: (>= 90):** When all assignments are submitted on or before the given deadline.
* **Grade B: (>= 80 and < 90):** 
  + When assignments are submitted on time but less than 80% of problems are completed.

(OR)

* + All assignments are submitted after the deadline.
* **Grade C: (>= 70 and < 80):** 
  + When assignments are submitted on time but less than 50% of the problems are completed.

(OR)

* + Less than 80% of problems in the assignments are submitted after the deadline.
* **Grade D: (>= 60 and < 70):**
  + Assignments submitted after the deadline and with 50% or less problems.
* **Grade E: (>= 50 and < 60):** 
  + Less than 30% of problems in the assignments are submitted after the deadline.

(OR)

* + Less than 30% of problems in the assignments are submitted before the deadline.
* **Grade F: (< 50):** No submission (or) malpractice.

**Hints:**

1. **Business Problem**
   1. **What is the business objective?**
   2. **Are there any constraints?**
2. **Work on each feature of the dataset to create a data dictionary as displayed in the below image:**



**2.1 Make a table as shown above and provide information about the features such as its data type and its relevance to the model building. And if not relevant, provide reasons and a description of the feature.**

1. **Data Pre-processing**

**3.1 Data Cleaning, Feature Engineering, etc.**

**3.2 Outlier Treatment.**

1. **Exploratory Data Analysis (EDA):**
   1. **Summary.**
   2. **Univariate analysis.**
   3. **Bivariate analysis.**
2. **Model Building**
   1. **Build the model on the scaled data (try multiple options).**
   2. **Perform Lasso and Ridge Regression.**
   3. **Train and test the model and compare RMSE values. Tabulate R-Squared and RMSE values for different models in the documentation and provide an explanation.**
   4. **Briefly explain the model output in the documentation.**
3. **Write about the benefits/impact of the solution - in what way does the business (client) benefit from the solution provided?**

**Business Problem:-**

Officeworks is a leading retail store in Australia, with numerous outlets around the country. The manager would like to improve the customer experience by providing them online predictive prices for their laptops if they want to sell them. To improve this experience the manager would like us to build a model which is sustainable and accurate enough. Apply Lasso and Ridge Regression model on the dataset and predict the price, given other attributes. Tabulate R squared, RMSE, and correlation values.

**What is the business objective?**

Build a model which is sustainable and accurate enough to do online predictive prices for the laptops of the customer if they want to sell them

**Are there any constraints?**

**Maximize:** customer satisfaction

**Maximize:** accuracy of the model

**Maximize:** business

**Maximize:** profit

**Python Code:-**

# Multilinear Regression with Regularization using L1 and L2 norm

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import statsmodels.formula.api as smf

import seaborn as sns

from sklearn.preprocessing import LabelEncoder

from sklearn.preprocessing import StandardScaler, MinMaxScaler,RobustScaler

# loading the data

df = pd.read\_csv("C:/Users/user/Downloads/lasso ridge/Computer\_Data (1).csv")

#creating one copy

df1=df.copy(deep=True)

###### Null value Treatment ########

df1.isna().sum()

df1.dropna(axis = 0, inplace = True) ## drop na values

df1.info()

df1.columns

#summary

df1.describe()

# converting ouput variable to numeric binary dummy variable format

lb = LabelEncoder()

df1['cd'] =lb.fit\_transform(df1['cd'])

df1['multi'] =lb.fit\_transform(df1['multi'])

df1['premium'] =lb.fit\_transform(df1['premium'])

# normalisation

from sklearn.preprocessing import StandardScaler, MinMaxScaler,RobustScaler

# define standard scaler

scaler1 = MinMaxScaler() # MinMax Scaler or Normalization

# Transform data ; except index& out column

df1.iloc[:,2:] = scaler1.fit\_transform(df1.iloc[:,2:]) #Fit to data, then transform it.

# converting back to dataframe

df1 = pd.DataFrame(df1)

df.columns

df1.columns = 'index', 'price', 'speed', 'hd', 'ram', 'screen', 'cd', 'multi','premium', 'ads', 'trend'

# droping index column for easy operation

df1.drop('index', axis=1, inplace = True)

df1.head()

df1.describe()

# Correlation matrix

a = df1.corr()

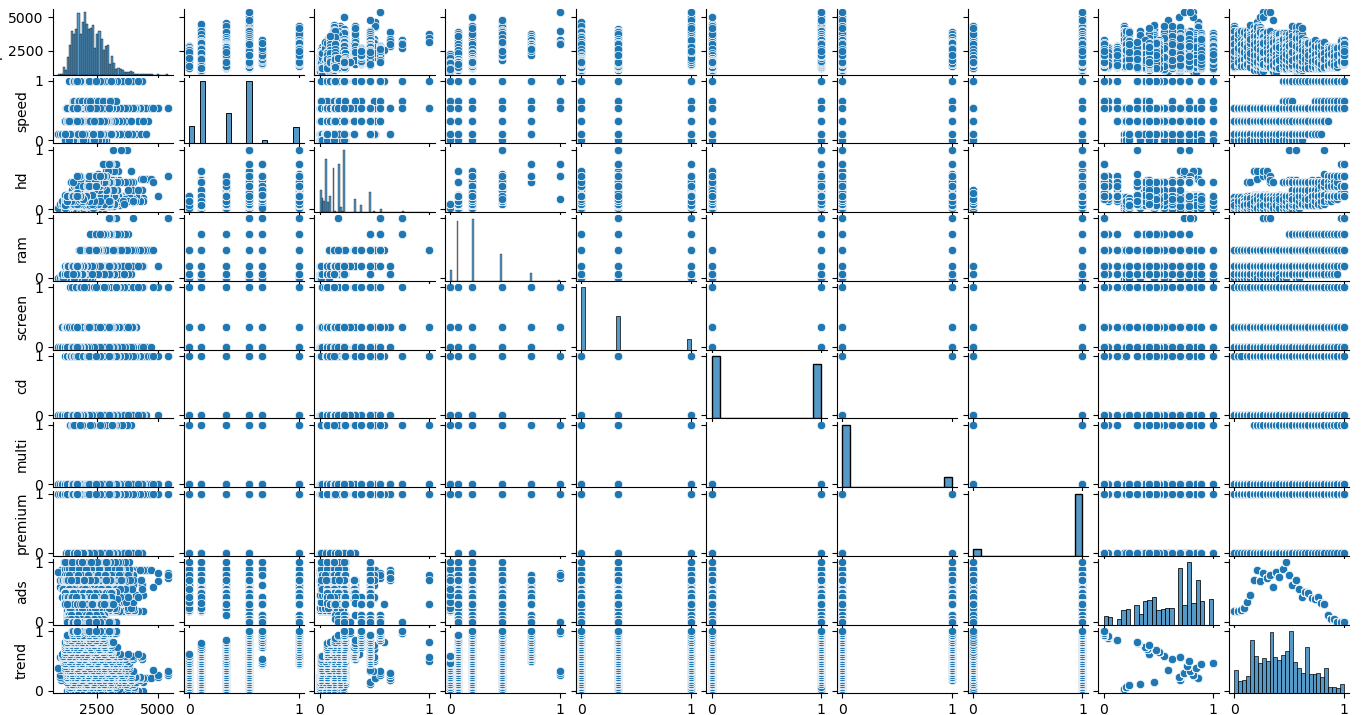
a

# EDA

a1 = df1.describe()

# Scatter plot and histogram between variables

sns.pairplot(df1) # not having any multicolinearity issue between independant input variables



#train test split

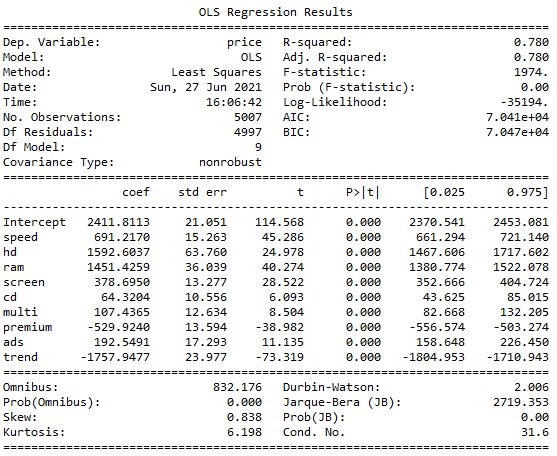
from sklearn.model\_selection import train\_test\_split

train, test = train\_test\_split(df1, test\_size = 0.2, random\_state=42)

# Preparing the model on train data

model = smf.ols("price ~ speed + hd + ram + screen + cd + multi + premium + ads + trend", data = train).fit()

model.summary()



#evaluation on test data

# Prediction

pred\_test = model.predict(test)

# Error

resid\_test = pred\_test - test.price

# RMSE value for data

rmse\_test = np.sqrt(np.mean(resid\_test \* resid\_test))

rmse\_test # 283.43

C:\Users\user\Documents\Figure_1.png

#evaluation on train data

# Prediction

pred\_train = model.predict(train)

# Error

resid\_train = pred\_train - train.price

# RMSE value for data

rmse\_train = np.sqrt(np.mean(resid\_train \* resid\_train))

rmse\_train # 273.15

C:\Users\user\Documents\Figure_1.png

# To overcome the issues(reduce error value OR over fit problem), LASSO and RIDGE regression are used

######## LASSO REGRESSION MODEL ##########

from sklearn.model\_selection import GridSearchCV

from sklearn.linear\_model import Lasso

help(Lasso)

lasso = Lasso()

parameters\_l = {'alpha': [1e-15, 1e-10, 1e-8, 1e-4, 1e-3, 1e-2, 1, 5 ,10, 20]}

lasso\_reg = GridSearchCV(lasso, parameters\_l, scoring = 'r2', cv = 5)

lasso\_reg.fit(train.iloc[:, 1:], train.price)

lasso\_reg.best\_params\_ # 0.01

lasso\_reg.best\_score\_ # 0.78

lasso\_pred\_train = lasso\_reg.predict(train.iloc[:, 1:])

# Adjusted r-square#

lasso\_reg.score(train.iloc[:, 1:], train.price) # 0.78

C:\Users\user\Documents\Figure_1.png

# RMSE

np.sqrt(np.mean((lasso\_pred\_train - train.price)\*\*2)) # 273.15

# lasso model for best alpha value

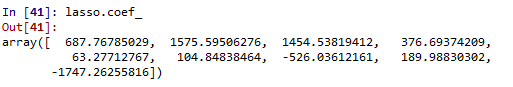
lasso = Lasso(alpha = 0.01, normalize = True)

# fit the train data with best alpha lasso model

lasso.fit(train.iloc[:, 1:], train.price)

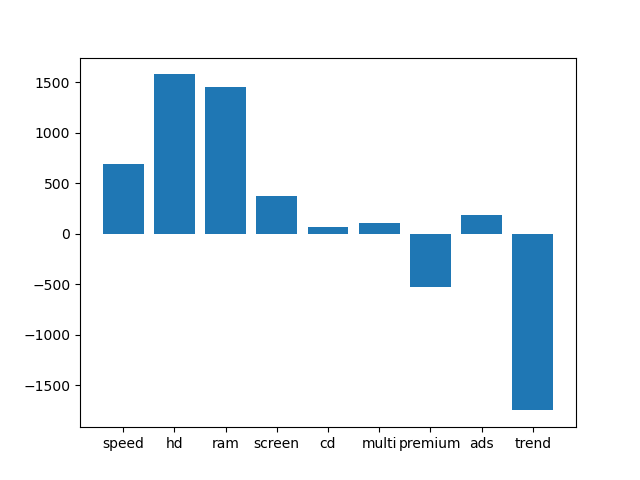
# Coefficient values for all independent variables#

lasso.coef\_



lasso.intercept\_

plt.bar(height = pd.Series(lasso.coef\_), x = pd.Series(train.columns[1:]))



###LASSO Evaluation on Test Data###

# Prediction

pred\_lasso\_test = lasso.predict(test.iloc[:, 1:])

# Error

resid\_test = pred\_lasso\_test - test.price

# RMSE value for data

rmse\_lasso\_test = np.sqrt(np.mean(resid\_test \* resid\_test))

rmse\_lasso\_test # 283.32

C:\Users\user\Documents\Figure_1.png

# Test data score in adusted r\_square term

lasso\_reg.score(test.iloc[:, 1:], test.price) # 0.75

C:\Users\user\Documents\Figure_1.png

####### RIDGE REGRESSION MODEL #######

from sklearn.linear\_model import Ridge

help(Ridge)

ridge = Ridge()

# since in ridge have no problem of coeficient elimination we are going for a wide search of lambda value(tuning parameter)

p = []

x = 1e-320

for l in range(1,9,1):

if(x > 1e-50):

break

else:

p.append(x)

x \*= 1e+40

l=l+1

for i in range(1, 23, 1):

p.append(x)

x = x \* 100

if(x > 1):

break

i = i+1

for c in range(1,50,4):

p.append(c)

p

parameters = {'alpha': p}

ridge\_reg = GridSearchCV(ridge, parameters, scoring = 'r2', cv = 5)

ridge\_reg.fit(train.iloc[:, 1:], train.price)

ridge\_reg.best\_params\_ # 0.009999888671826829

ridge\_reg.best\_score\_ # 0.78

ridge\_pred\_train = ridge\_reg.predict(train.iloc[:, 1:])

# Adjusted r-square#

ridge\_reg.score(train.iloc[:, 1:], train.price)

C:\Users\user\Documents\Figure_1.png

# RMSE

np.sqrt(np.mean((ridge\_pred\_train - train.price)\*\*2)) # 273.15

C:\Users\user\Documents\Figure_1.png

# ridge model for best alpha value

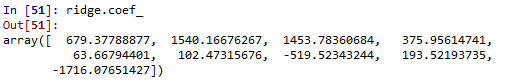
ridge = Ridge(alpha = 0.009999888671826829, normalize = True)

# fit the train data with best alpha ridge model

ridge.fit(train.iloc[:, 1:], train.price)

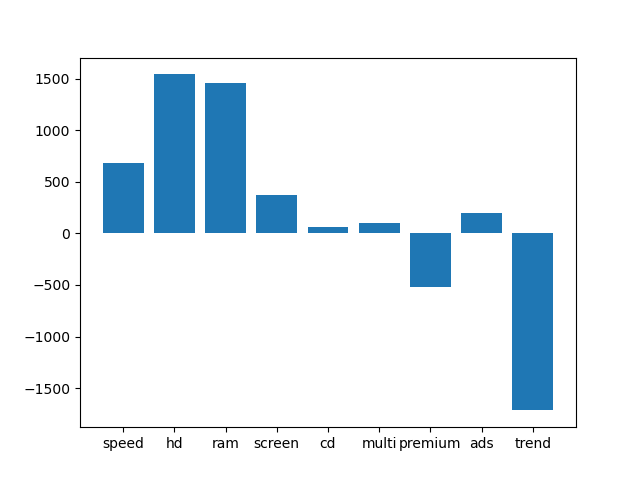
# Coefficient values for all independent variables#

ridge.coef\_



ridge.intercept\_

plt.bar(height = pd.Series(ridge.coef\_), x = pd.Series(train.columns[1:]))



###RIDGE Evaluation on Test Data###

# Prediction

pred\_ridge\_test = ridge.predict(test.iloc[:, 1:])

# Error

resid\_test = pred\_ridge\_test - test.price

# RMSE value for data

rmse\_ridge\_test = np.sqrt(np.mean(resid\_test \* resid\_test))

rmse\_ridge\_test # 283.38

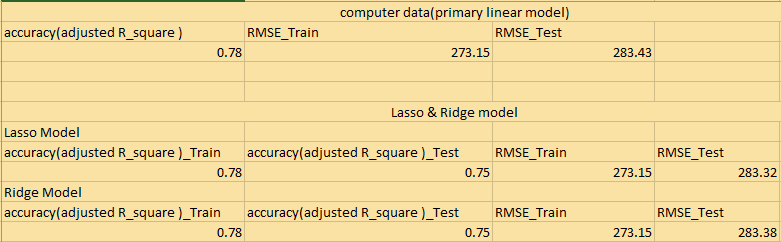
C:\Users\user\Documents\Figure_1.png

# Test data score in adusted r\_square term

ridge\_reg.score(test.iloc[:, 1:], test.price) # 0.75

C:\Users\user\Documents\Figure_1.png

**Summary:-**

****

* All model showing almost similar response
* lasso & Ridge showing similar performance because like ridge model there haven’t happened any coeficient elimination in lasso model.
* So both model are showing similar response in every aspects(train & test data model accuracy in adjusted R2 & in RMSE error term)

Business benefit:-

It helps the client to predict the prices of the customers laptop from the given datas. There by provide an accurate and enhance or fast system to get a result within times for both clients and customers. Accurate and fast results will catch up or attract more customers and this will helps to improves the business.

**Business Problem:-**

An online car sales platform would like to improve its customer base and their experience by providing them an easy way to buy and sell cars. For this, they would like to have an automated model which can predict the price of the car once the user inputs the required factors. Help the business achieve the objective by applying Lasso and Ridge Regression on it. Please use the below columns for the analysis: Price, Age\_08\_04, KM, HP, cc, Doors, Gears, Quarterly\_Tax, Weight.

**What is the business objective?**

Build an automated model which can predict the price of the car once the user inputs the required factors

**Are there any constraints?**

**Maximize:** customer satisfaction

**Maximize:** accuracy of the model

**Maximize:** business improvemnt

**Maximize:** profit

**Python Code:-**

# Multilinear Regression with Regularization using L1 and L2 norm

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import statsmodels.formula.api as smf

import seaborn as sns

from sklearn.preprocessing import LabelEncoder

from sklearn.preprocessing import StandardScaler, MinMaxScaler,RobustScaler

# loading the data

df = pd.read\_csv("C:/Users/user/Downloads/lasso ridge/ToyotaCorolla (1).csv",encoding = ('ISO-8859-1'),low\_memory = False)

#creating one copy

df1=df.copy(deep=True)

df1.columns

# droping unwanted columns

df1 = df1[['Price', 'Age\_08\_04', 'KM', 'HP', 'cc','Doors', 'Gears', 'Quarterly\_Tax', 'Weight']]

###### Null value Treatment ########

df1.isna().sum()

df1.dropna(axis = 0, inplace = True) ## drop na values

df1.info()

df1.columns

#summary

df1.describe()

### standardization scaling ###

#Importing the Libraries

from sklearn.preprocessing import StandardScaler, MinMaxScaler,RobustScaler

# define standard scaler

scaler = StandardScaler() # Standard Scaler or Standardization

# Transform data ## "TYPE" column is considered as ouput. so not gonna do any scaling there

df1.iloc[:,1:] = scaler.fit\_transform(df1.iloc[:,1:]) #Fit to data, then transform it.

df1.describe()

df1.columns

df1.head()

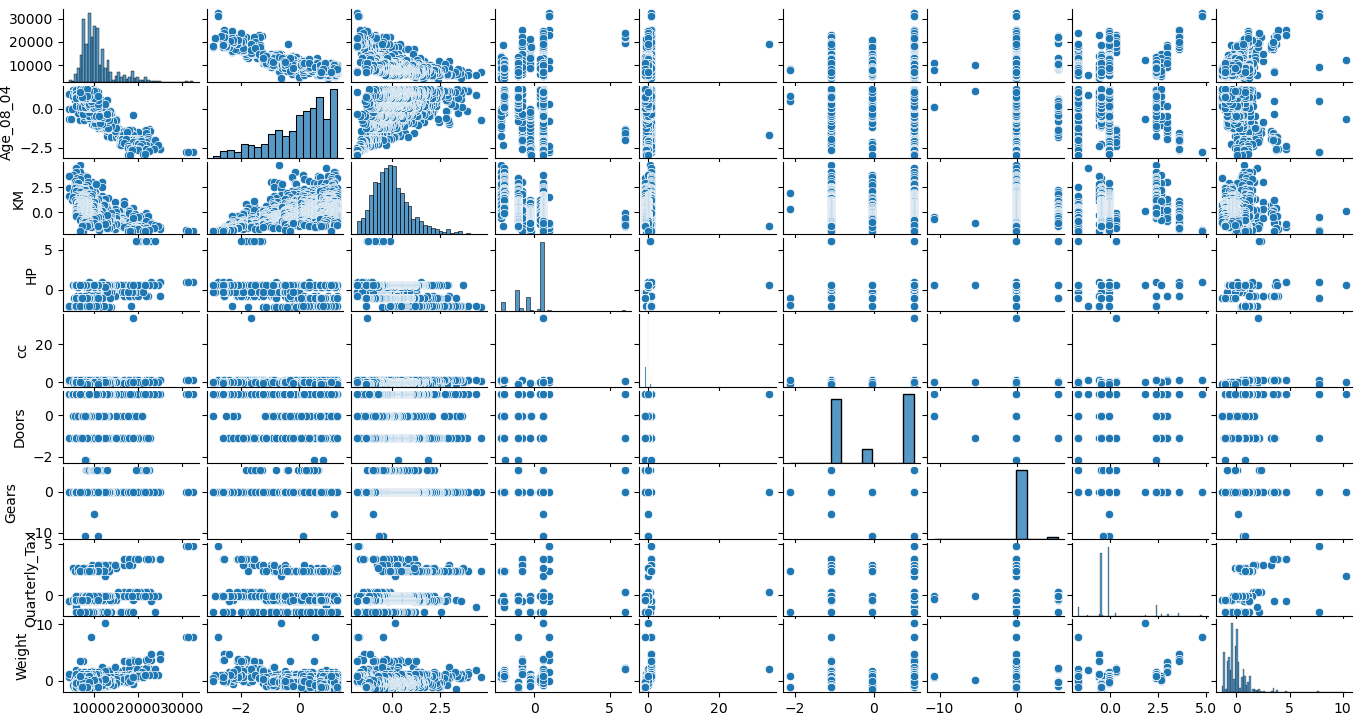
# Correlation matrix

a = df1.corr()

a

# Scatter plot and histogram between variables

sns.pairplot(df1) # not having any multicolinearity issue between independant input variables



#train test split

from sklearn.model\_selection import train\_test\_split

train, test = train\_test\_split(df1, test\_size = 0.2, random\_state=42)

# Preparing the model on train data

model = smf.ols("Price ~ Age\_08\_04 + KM + HP + cc + Doors + Gears + Quarterly\_Tax + Weight", data = train).fit()

model.summary()

#evaluation on test data

# Prediction

pred\_test = model.predict(test)

# Error

resid\_test = pred\_test - test.Price

# RMSE value for data

rmse\_test = np.sqrt(np.mean(resid\_test \* resid\_test))

rmse\_test # 1396.51

#evaluation on train data

# Prediction

pred\_train = model.predict(train)

# Error

resid\_train = pred\_train - train.Price

# RMSE value for data

rmse\_train = np.sqrt(np.mean(resid\_train \* resid\_train))

rmse\_train # 1327.13

# To overcome the issues(reduce error value OR over fit problem), LASSO and RIDGE regression are used

######## LASSO REGRESSION MODEL ##########

from sklearn.model\_selection import GridSearchCV

from sklearn.linear\_model import Lasso

help(Lasso)

lasso = Lasso()

parameters\_l = {'alpha': [1e-15, 1e-10, 1e-8, 1e-4, 1e-3, 1e-2, 1, 5 ,10, 20]}

lasso\_reg = GridSearchCV(lasso, parameters\_l, scoring = 'r2', cv = 5)

lasso\_reg.fit(train.iloc[:, 1:], train.Price)

lasso\_reg.best\_params\_ # 5

lasso\_reg.best\_score\_ # 0.85

lasso\_pred\_train = lasso\_reg.predict(train.iloc[:, 1:])

# Adjusted r-square#

lasso\_reg.score(train.iloc[:, 1:], train.Price) #0.86

# RMSE

np.sqrt(np.mean((lasso\_pred\_train - train.Price)\*\*2)) # 1358.57

# lasso model for best alpha value

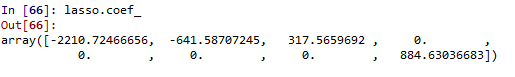
lasso = Lasso(alpha = 5, normalize = True)

# fit the train data with best alpha lasso model

lasso.fit(train.iloc[:, 1:], train.Price)

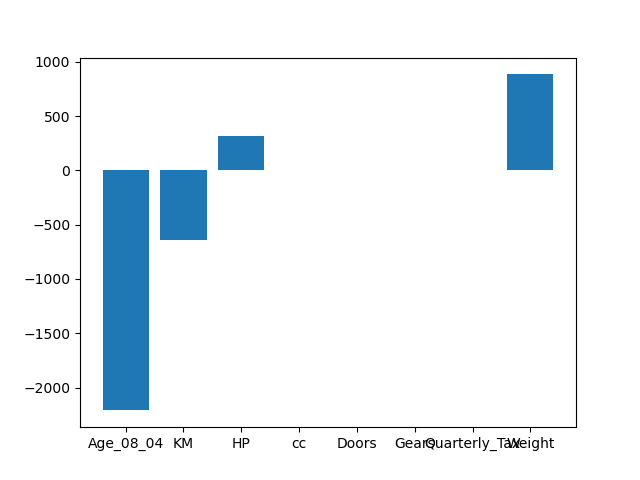
# Coefficient values for all independent variables#

lasso.coef\_



lasso.intercept\_

plt.bar(height = pd.Series(lasso.coef\_), x = pd.Series(train.columns[1:]))



###LASSO Evaluation on Test Data###

# Prediction

pred\_lasso\_test = lasso.predict(test.iloc[:, 1:])

# Error

resid\_test = pred\_lasso\_test - test.Price

# RMSE value for data

rmse\_lasso\_test = np.sqrt(np.mean(resid\_test \* resid\_test))

rmse\_lasso\_test # 1425.01

# Test data score in adusted r\_square term

lasso\_reg.score(test.iloc[:, 1:], test.Price) # 0.85

####### RIDGE REGRESSION MODEL #######

from sklearn.linear\_model import Ridge

help(Ridge)

ridge = Ridge()

# since in ridge have no problem of coeficient elimination we are going for a wide search of lambda value(tuning parameter)

p = []

x = 1e-320

for l in range(1,9,1):

if(x > 1e-50):

break

else:

p.append(x)

x \*= 1e+40

l=l+1

for i in range(1, 23, 1):

p.append(x)

x = x \* 100

if(x > 1):

break

i = i+1

for c in range(1,50,4):

p.append(c)

p

parameters = {'alpha': p}

ridge\_reg = GridSearchCV(ridge, parameters, scoring = 'r2', cv = 5)

ridge\_reg.fit(train.iloc[:, 1:], train.Price)

ridge\_reg.best\_params\_ # 49

ridge\_reg.best\_score\_ # 0.84

ridge\_pred\_train = ridge\_reg.predict(train.iloc[:, 1:])

# Adjusted r-square#

ridge\_reg.score(train.iloc[:, 1:], train.Price)

# RMSE

np.sqrt(np.mean((ridge\_pred\_train - train.Price)\*\*2)) # 1330.60

# ridge model for best alpha value

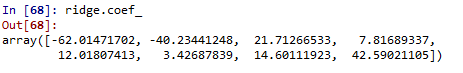
ridge = Ridge(alpha = 49, normalize = True)

# fit the train data with best alpha ridge model

ridge.fit(train.iloc[:, 1:], train.Price)

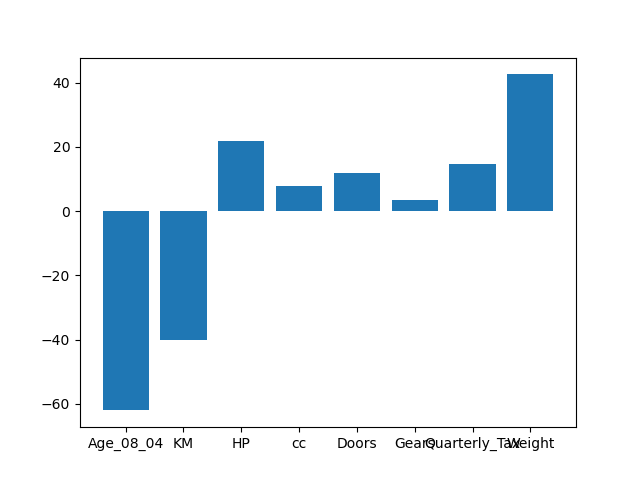
# Coefficient values for all independent variables#

ridge.coef\_



ridge.intercept\_

plt.bar(height = pd.Series(ridge.coef\_), x = pd.Series(train.columns[1:]))



###RIDGE Evaluation on Test Data###

# Prediction

pred\_ridge\_test = ridge.predict(test.iloc[:, 1:])

# Error

resid\_test = pred\_ridge\_test - test.Price

# RMSE value for data

rmse\_ridge\_test = np.sqrt(np.mean(resid\_test \* resid\_test))

rmse\_ridge\_test # 3537.21

# Test data score in adusted r\_square term

ridge\_reg.score(test.iloc[:, 1:], test.Price) # 0.85

ridge.score(test.iloc[:, 1:], test.Price)

**Summary:-**

****

* Both models showing almost similar accuracies on test & train datas
* But compare to Ridge model Lasso showing better performance in the case of RMSE error value(less error value for test and train) for test and train datas as because they eliminate 4 less informative input independent feature.
* But we cant say Lasso model is completely better, because the eliminated input features may have chanages in their influence for the prediction of output dependant variable in future. As time changes the the strength of features influences may also got change.

Business benefit:-

It helps the client to predict the prices of the customers cars from the given datas. There by provide an accurate and enhance or fast system to get a result within times for both seller and buyer. Accurate and fast results will catch up or attract more customers usage of the service. This will helps the client to improves the business.

**Business Problem:-**

Data of various countries and the factors affecting their life expectancy has been recorded over the past few decades. An analytics firm would like to know how it varies country wise and what factors are influential. Use your skills to analyze the data and build a Lasso and Ridge Regression model and summarize the output. Snapshot of the dataset is given below.

**What is the business objective?**

Build an automated model which can predict the life expectancy from the data. And to know how it varies country wise and what factors are influential.

**Are there any constraints?**

**Maximize:** accuracy of the model

**Python Code:-**

# Multilinear Regression with Regularization using L1 and L2 norm

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import statsmodels.formula.api as smf

import seaborn as sns

from sklearn.preprocessing import LabelEncoder

from sklearn.preprocessing import StandardScaler, MinMaxScaler,RobustScaler

# loading the data

df = pd.read\_csv("C:/Users/user/Downloads/lasso ridge/Life\_expectencey\_LR.csv")

#creating one copy

df1=df.copy(deep=True)

df1.columns

# rearranging the columns

df1 = df1[['Life\_expectancy','Country','Status', 'Year','Adult\_Mortality',

'infant\_deaths', 'Alcohol', 'percentage\_expenditure', 'Hepatitis\_B',

'Measles', 'BMI', 'under\_five\_deaths', 'Polio', 'Total\_expenditure',

'Diphtheria', 'HIV\_AIDS', 'GDP', 'Population', 'thinness',

'thinness\_yr', 'Income\_composition', 'Schooling']]

###### Null value Treatment ########

df1.isna().sum()

df1.dropna(axis = 0, inplace = True) ## drop na values

df1.info()

df1.columns

#summary

df1.describe()

# converting ouput variable to numeric binary dummy variable format

lb = LabelEncoder()

df1['Country'] =lb.fit\_transform(df1['Country'])

df1['Status'] =lb.fit\_transform(df1['Status'])

### standardization scaling ###

#Importing the Libraries

from sklearn.preprocessing import StandardScaler, MinMaxScaler,RobustScaler

# define standard scaler

scaler = StandardScaler() # Standard Scaler or Standardization

# Transform data ## "TYPE" column is considered as ouput. so not gonna do any scaling there

df1.iloc[:,1:] = scaler.fit\_transform(df1.iloc[:,1:]) #Fit to data, then transform it.

df1.describe()

df1.columns

df1.head()

# Correlation matrix

a = df1.corr()

a

# Scatter plot and histogram between variables

sns.pairplot(df1) # having any multicolinearity issue between GDP & 'percentage\_expenditure','thinness'&

# 'thinness\_yr' , 'infant\_deaths'&'under\_five\_deaths'independant input variables

# so actually we have to treat and eliminate the corresponding problem ;

# but here we look up only on lasso & ridge method results



#train test split

from sklearn.model\_selection import train\_test\_split

train, test = train\_test\_split(df1, test\_size = 0.2, random\_state=42)

# Preparing the model on train data

model = smf.ols('Life\_expectancy ~ Country + Status + Year + Adult\_Mortality + infant\_deaths + Alcohol + percentage\_expenditure + Hepatitis\_B + Measles + BMI + under\_five\_deaths + Polio + Total\_expenditure +Diphtheria + HIV\_AIDS + GDP + Population + thinness + thinness\_yr + Income\_composition + Schooling', data = train).fit()

model.summary()

#evaluation on test data

# Prediction

pred\_test = model.predict(test)

# Error

resid\_test = pred\_test - test.Life\_expectancy

# RMSE value for data

rmse\_test = np.sqrt(np.mean(resid\_test \* resid\_test))

rmse\_test # 3.61

#evaluation on train data

# Prediction

pred\_train = model.predict(train)

# Error

resid\_train = pred\_train - train.Life\_expectancy

# RMSE value for data

rmse\_train = np.sqrt(np.mean(resid\_train \* resid\_train))

rmse\_train # 3.52

# To overcome the issues(reduce error value OR over fit problem), LASSO and RIDGE regression are used

######## LASSO REGRESSION MODEL ##########

from sklearn.model\_selection import GridSearchCV

from sklearn.linear\_model import Lasso

help(Lasso)

lasso = Lasso()

parameters\_l = {'alpha': [1e-15, 1e-10, 1e-8, 1e-4, 1e-3, 1e-2, 1, 5 ,10, 20]}

lasso\_reg = GridSearchCV(lasso, parameters\_l, scoring = 'r2', cv = 5)

lasso\_reg.fit(train.iloc[:, 1:], train.Life\_expectancy)

lasso\_reg.best\_params\_ # 1e-08

lasso\_reg.best\_score\_ # 0.83

lasso\_pred\_train = lasso\_reg.predict(train.iloc[:, 1:])

# Adjusted r-square#

lasso\_reg.score(train.iloc[:, 1:], train.Life\_expectancy) #0.84

# RMSE

np.sqrt(np.mean((lasso\_pred\_train - train.Life\_expectancy)\*\*2)) # 3.524

# lasso model for best alpha value

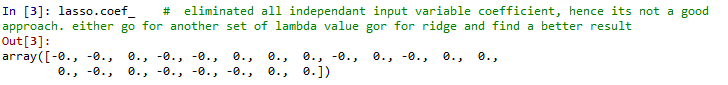
lasso = Lasso(alpha = 49, normalize = True)

# fit the train data with best alpha lasso model

lasso.fit(train.iloc[:, 1:], train.Life\_expectancy)

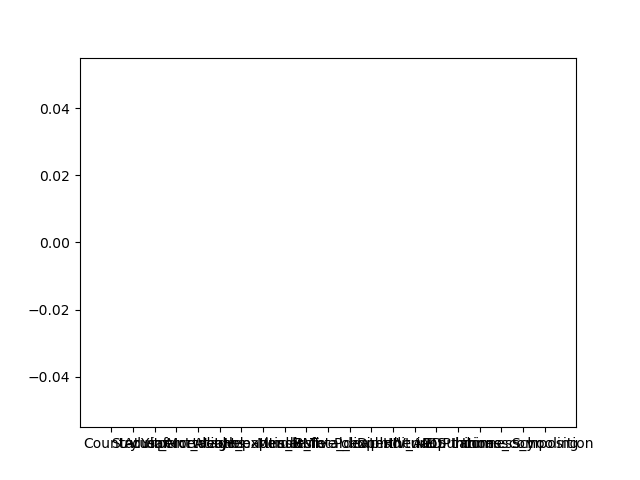
# Coefficient values for all independent variables#

lasso.coef\_ # eliminated all independant input variable coefficient, hence its not a good approach. either go for another set of lambda value gor for ridge and find a better result



lasso.intercept\_

plt.bar(height = pd.Series(lasso.coef\_), x = pd.Series(train.columns[1:]))



###LASSO Evaluation on Test Data###

# Prediction

pred\_lasso\_test = lasso.predict(test.iloc[:, 1:])

# Error

resid\_test = pred\_lasso\_test - test.Life\_expectancy

# RMSE value for data

rmse\_lasso\_test = np.sqrt(np.mean(resid\_test \* resid\_test))

rmse\_lasso\_test # 8.46

# Test data score in adusted r\_square term

lasso\_reg.score(test.iloc[:, 1:], test.Life\_expectancy) # 0.82

####### RIDGE REGRESSION MODEL #######

from sklearn.linear\_model import Ridge

help(Ridge)

ridge = Ridge()

# since in ridge have no problem of coeficient elimination we are going for a wide search of lambda value(tuning parameter)

p = []

x = 1e-320

for l in range(1,9,1):

if(x > 1e-50):

break

else:

p.append(x)

x \*= 1e+40

l=l+1

for i in range(1, 23, 1):

p.append(x)

x = x \* 100

if(x > 1):

break

i = i+1

for c in range(1,50,4):

p.append(c)

p

parameters = {'alpha': p}

ridge\_reg = GridSearchCV(ridge, parameters, scoring = 'r2', cv = 5)

ridge\_reg.fit(train.iloc[:, 1:], train.Life\_expectancy)

ridge\_reg.best\_params\_ # 0.009999888671826829

ridge\_reg.best\_score\_ # 0.83

ridge\_pred\_train = ridge\_reg.predict(train.iloc[:, 1:])

# Adjusted r-square#

ridge\_reg.score(train.iloc[:, 1:], train.Life\_expectancy) # 0.84

# RMSE

np.sqrt(np.mean((ridge\_pred\_train - train.Life\_expectancy)\*\*2)) # 3.52

# ridge model for best alpha value

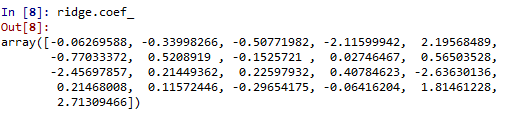
ridge = Ridge(alpha = 0.009999888671826829, normalize = True)

# fit the train data with best alpha ridge model

ridge.fit(train.iloc[:, 1:], train.Life\_expectancy)

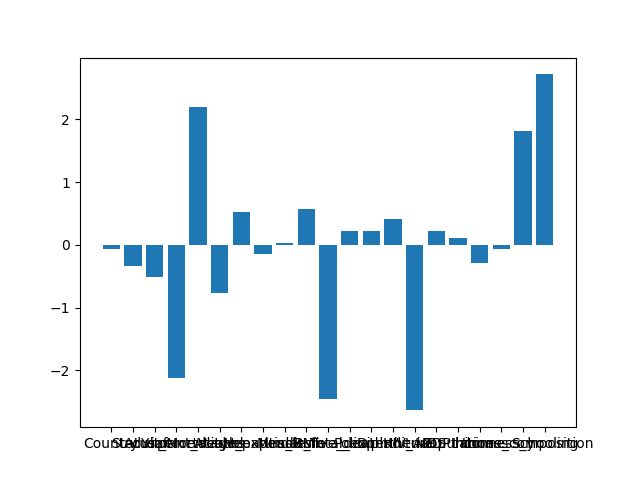
# Coefficient values for all independent variables#

ridge.coef\_



ridge.intercept\_

plt.bar(height = pd.Series(ridge.coef\_), x = pd.Series(train.columns[1:]))



###RIDGE Evaluation on Test Data###

# Prediction

pred\_ridge\_test = ridge.predict(test.iloc[:, 1:])

# Error

resid\_test = pred\_ridge\_test - test.Life\_expectancy

# RMSE value for data

rmse\_ridge\_test = np.sqrt(np.mean(resid\_test \* resid\_test))

rmse\_ridge\_test # 3.64

# Test data score in adusted r\_square term

ridge\_reg.score(test.iloc[:, 1:], test.Life\_expectancy) # 0.82

**Summary:-**

****

* Both models showing almost similar accuracies on test & train datas
* But compare to Lasso model, Ridge showing better performance in the case of RMSE error value(less error value for test and train) for test and train datas
* Here Lasso model would perform like base model as because there happened complete elimination input independent feature coefficients..
* So in every aspects Ridge model ismore better than Lasso model for this perticular data.

Business benefit:-

It helps to find the influence of different features upon predicting the life expectancy of peoples. As because we given country name as one of the input feature we can see how life expentancy changes with countries for other input features. So that will helps to predict the different features influence upon life expectancy calculation on different countries. So the corresponding authority can took the remedial actions based on the input features.

**Problem Statements:**

1. Officeworks is a leading retail store in Australia, with numerous outlets around the country. The manager would like to improve the customer experience by providing them online predictive prices for their laptops if they want to sell them. To improve this experience the manager would like us to build a model which is sustainable and accurate enough. Apply Lasso and Ridge Regression model on the dataset and predict the price, given other attributes. Tabulate R squared, RMSE, and correlation values.





1. An online car sales platform would like to improve its customer base and their experience by providing them an easy way to buy and sell cars. For this, they would like to have an automated model which can predict the price of the car once the user inputs the required factors. Help the business achieve the objective by applying Lasso and Ridge Regression on it. Please use the below columns for the analysis: Price, Age\_08\_04, KM, HP, cc, Doors, Gears, Quarterly\_Tax, Weight.



1. Data of various countries and the factors affecting their life expectancy has been recorded over the past few decades. An analytics firm would like to know how it varies country wise and what factors are influential. Use your skills to analyze the data and build a Lasso and Ridge Regression model and summarize the output. Snapshot of the dataset is given below.

A screenshot of a cell phone

Description automatically generated