In [1]:
```python
"""
Landon Buell
Marek Petrik
CS 750.01
18 Feb 2020
"""

import numpy as np
import pandas as pd
import sklearn.metrics as metrics
```

Problem 1 [25%]

In this exercise, we will predict the number of applications received using the other variables in the College (ISLR::College) data set.

In [2]:
```python
# Load in data set, print out head
college = pd.read_csv('college.csv',index_col=0)
college['Private'] = college['Private'].map({'Yes':1,'No':0})
print(college.head())

# create X matrix & target vector
X = college.drop(['Apps','Private'],axis=1)
y = college['Apps']
X.to_numpy()
y.to_numpy()
print(X.shape,y.shape)

# split into train and test sets
from sklearn.model_selection import train_test_split
xtrain,xtest,ytrain,ytest = train_test_split(X,y,test_size=0.2)
```

```
                              Private  Apps  Accept  Enroll  Top10perc  \
Abilene Christian University        1  1660    1232     721         23
Adelphi University                  1  2186    1924     512         16
Adrian College                      1  1428    1097     336         22
Agnes Scott College                 1   417     349     137         60
Alaska Pacific University           1   193     146      55         16

                              Top25perc  F.Undergrad  P.Undergrad  Outstate
\
Abilene Christian University         52         2885          537      7440
Adelphi University                   29         2683         1227     12280
Adrian College                       50         1036           99     11250
Agnes Scott College                  89          510           63     12960
Alaska Pacific University            44          249          869      7560

                              Room.Board  Books  Personal  PhD  Terminal  \
Abilene Christian University        3300    450      2200   70        78
Adelphi University                  6450    750      1500   29        30
Adrian College                      3750    400      1165   53        66
Agnes Scott College                 5450    450       875   92        97
Alaska Pacific University           4120    800      1500   76        72

                              S.F.Ratio  perc.alumni  Expend  Grad.Rate
Abilene Christian University       18.1           12    7041         60
Adelphi University                 12.2           16   10527         56
Adrian College                     12.9           30    8735         54
Agnes Scott College                 7.7           37   19016         59
Alaska Pacific University          11.9            2   10922         15
(777, 16) (777,)
```

1. Fit a linear model using least squares on the training set, and report the test error obtained.

In [3]:
```
from sklearn.linear_model import LinearRegression

# Fit least sq. model
linreg = LinearRegression()
linreg.fit(xtrain,ytrain)

ypred = linreg.predict(xtest)
linreg_MSE = metrics.mean_squared_error(ytest,ypred)
print("Testing Error determined by MSE:",linreg_MSE)
print("This seems unreasonably large!")
```

```
Testing Error determined by MSE: 1078327.8649763155
This seems unreasonably large!
```

2. Use best subset selection with cross-validation. Report the test error obtained.

In [4]:
```
from sklearn.feature_selection import SelectKBest , f_regression
# I wasn't sure how to do this with X-val in python b/c of the arguments requi
red!

Kbest = SelectKBest(score_func=f_regression,k=4)
Kbest.fit(xtrain,ytrain)

# create new training data set
xtrain_new_tp = xtrain.transpose()[Kbest.get_support()]
xtrain_new = xtrain_new_tp.transpose()
xtrain_new.head()
```

Out[4]:

	Accept	Enroll	F.Undergrad	PhD
Virginia Tech	11719.0	4277.0	18511.0	85.0
Chapman University	771.0	351.0	1662.0	72.0
King's College	1053.0	381.0	500.0	66.0
Wisconsin Lutheran College	128.0	75.0	282.0	48.0
University of Massachusetts at Dartmouth	2597.0	1006.0	4664.0	74.0

In [5]:
```python
# Create new linear regression class
linreg2 = LinearRegression()
linreg2.fit(xtrain_new,ytrain)

# new testing data set
xtest_new_tp = xtest.transpose()[Kbest.get_support()]
xtest_new = xtest_new_tp.transpose()

# new prediction on data subset
ypred2 = linreg2.predict(xtest_new)
linreg2_MSE = metrics.mean_squared_error(ytest,ypred2)
print("Testing Error determined by MSE:",linreg2_MSE)
print("It's increased! What have I done wrong???")
```

```
Testing Error determined by MSE: 1428472.7131221814
It's increased! What have I done wrong???
```

3. Fit a ridge regression model on the training set, with λ chosen by cross-validation.

In [ ]:
```python

```

4. Fit a lasso model on the training set, with λ chosen by cross-validation. Report the test error obtained, along with the number of non-zero coefficient estimates.

In [ ]:
```python

```

5. Briefly comment on the results obtained. How accurately can we predict the number of college applications received? Is there much difference among the test errors resulting from these approaches?

Problem 2 [25%]

We will try to predict per capita crime rate in the Boston dataset.

In [6]:
```python
from sklearn.datasets import load_boston

# load in Data set & make frame
boston = load_boston()
X = pd.DataFrame(boston['data'],columns=boston['feature_names'])

# make intor X & y objects
y = X['CRIM']
X = X.drop(['CRIM'],axis=1)
xtrain,xtest,ytrain,ytest = train_test_split(X,y,test_size=0.2)
```

1. Try out best subset selection, the lasso, ridge regression, and PCR on this problem. Present and discuss results for the approaches that you consider.

In [7]:
```
Kbest = SelectKBest(score_func=f_regression,k=4)
Kbest.fit(xtrain,ytrain)

xtrain_new_tp = xtrain.transpose()[Kbest.get_support()]
xtrain_new = xtrain_new_tp.transpose()
xtest_new_tp = xtest.transpose()[Kbest.get_support()]
xtest_new = xtest_new_tp.transpose()
```

In [8]:
```
from sklearn.linear_model import Lasso

# train Lasso Instance on initial dataset
lasso_1 = Lasso()
lasso_1.fit(xtrain,ytrain)
ypred_1 = lasso_1.predict(xtest)
print("Testing Error determined by MSE:",metrics.mean_squared_error(ytest,ypre
d_1))

# train Lasso Instance on Kbest dataset
lasso_2 = Lasso()
lasso_2.fit(xtrain_new,ytrain)
ypred_2 = lasso_2.predict(xtest_new)
print("Testing Error determined by MSE:",metrics.mean_squared_error(ytest,ypre
d_2))
```

```
Testing Error determined by MSE: 62.62231547352068
Testing Error determined by MSE: 64.42077388420705
```

In [9]:
```
from sklearn.linear_model import Ridge

# train Ridge Instance on initial dataset
ridge_1 = Ridge()
ridge_1.fit(xtrain,ytrain)
ypred_1 = ridge_1.predict(xtest)
print("Testing Error determined by MSE:",metrics.mean_squared_error(ytest,ypre
d_1))

# train Ridge Instance on Kbest dataset
ridge_2 = Lasso()
ridge_2.fit(xtrain_new,ytrain)
ypred_2 = ridge_2.predict(xtest_new)
print("Testing Error determined by MSE:",metrics.mean_squared_error(ytest,ypre
d_2))
```

```
Testing Error determined by MSE: 61.679342154327976
Testing Error determined by MSE: 64.42077388420705
```

2. Propose a model (or set of models) that seem to perform well on this data set, and justify your answer. Make sure that you are evaluating model performance using validation set error, cross-validation, or some other reasonable alternative, as opposed to using training error.

Both Ridge and Lasso seem to work quite well on this model. Unforuately, The K best features from the data set for K = 4 seems to offer little, or occassionally no improvement from just the base data set.

Problem 3 [25%]

Suppose we have a linear regression problem with P features. We estimate the coefficients in the linear regression model by minimizing the RSS for the first p features:

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2$$

Where $p \leq P$ for parts (1) through (5) , indicate which is correct. Briefly justify your answer.

1. As we increase p from 1 to P, the training RSS will typically:

As we add more features, our training error go down. For a short while, It will likely produce a better fit for the data and perform well on the testing or validation set. Thus with each sucuessive p added, the difference between each y_i and $\hat{y}_i$ decreases until we run into the problem of overfitting the data set. Thus, the RSS will (v.) Decrease initially, and then eventually start increasing in a U shape.

2. As we increase p from 1 to P, the training MSE will typically:

Just like RSS, adding more features decreases the training error, at a certain point, adding more and more features will cause the training set to be overfitted and thus perform poorly on the testing or validation set. Again, the MSE will (v.) Decrease initially, and then eventually start increasing in a U shape.(ISL, fig. 2.12)

3. As we increase p from 1 to P, the training squared bias will typically:

The squared bais is the expectation value of the difference between the predicted and true output of a model. With each sucessive feature, the model is prone of overfitting and thus performs worse and worse on the testing set. This when paired with the fact that we are squaring the result, always produces a sucessivly larger positive number. Thus, the squared bias will (iii.) Steadily decrease.(ISL, fig. 2.12)

4. As we increase p from 1 to P, the training variance will typically:

Adding more features will generally cause the variance to (ii.) steadily increase (ISL, fig. 2.12)

5. As we increase p from 1 to P, the irreducible error (Bayes error) will typically:

Adding more features will cause the Bayes Error to reduce initially,, because the irreducible error always seeks the minimum possible value. However, when overfitting is comes into play, we can then say that the error will (v.) Decrease initially, and then eventually start increasing in a U shape.

Problem 4 [25%]

Suppose we estimate the regression coefficients in a linear regression model by minimizing:

$$\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2$$

Subject to:

$$\sum_{j=1}^{p}|\beta_j|^2 \geq s$$

for a particular value of s. For parts (1) through (5), indicate which of i. through v. is correct. Justify your answer.

1. As we increase s from 0, the training RSS will typically:

Increasing s to a sufficiently large value, then the abpove equation will produce a simple least squares fit. Thus, the training error will always (iii.) steadily decrease. (ISL,221)

2. As we increase s from 0, the testing RSS will typically:

Increasing s to a sufficiently large value, then the abpove equation will produce a simple least squares fit. Thus, the testing error will (v.) Decrease initially, and then eventually start increasing in a U shape, as overfitting takes place.

3. As we increase s from 0, the squared bias will typically:

Increasing s will increase the constraint value on the possible values of β_j. also increases. This causes the bais squared value to also (ii.) steadily increase.

4. As we increase s from 0, the variance will typically:

Increasing s will generally cause the variance to (ii.) steadily increase

5. As we increase s from 0, the Bayes Error will typically:

Increasing s will cause the Bayes Error to reduce initially,, because the irreducible error always seeks the minimum possible value. However, when overfitting is comes into play, we can then say that the error will (v.) Decrease initially, and then eventually start increasing in a U shape.