ASSIGNMENT GLASS

In [1]:

```
import numpy as np
import pandas as pd
```

In [2]:

```
glass=pd.read_csv("glass.csv")
glass.head()
```

Out[2]:

	RI	Na	Mg	Al	Si	K	Ca	Ba	Fe	Type
0	1.52101	13.64	4.49	1.10	71.78	0.06	8.75	0.0	0.0	1
1	1.51761	13.89	3.60	1.36	72.73	0.48	7.83	0.0	0.0	1
2	1.51618	13.53	3.55	1.54	72.99	0.39	7.78	0.0	0.0	1
3	1.51766	13.21	3.69	1.29	72.61	0.57	8.22	0.0	0.0	1
4	1.51742	13.27	3.62	1.24	73.08	0.55	8.07	0.0	0.0	1

In [3]:

```
glass.Type.unique()
```

Out[3]:

```
array([1, 2, 3, 5, 6, 7], dtype=int64)
```

In [4]:

```
glass.isna().sum()
```

Out[4]:

```
RI      0
Na      0
Mg      0
Al      0
Si      0
K       0
Ca      0
Ba      0
Fe      0
Type    0
dtype: int64
```

In [6]:

```
x=glass.drop('Type',axis=1)
y=glass['Type']
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import chi2

bestfeatures = SelectKBest(score_func=chi2, k=4)
fit = bestfeatures.fit(x,y)
dfscores = pd.DataFrame(fit.scores_)
dfcolumns = pd.DataFrame(x.columns)

featureScores = pd.concat([dfcolumns,round(dfscores,2)],axis=1)
featureScores.columns = ['Specs','Score']
#print(featureScores)
print(featureScores)
print(featureScores.nlargest(6,'Score'))
print(featureScores.nsmallest(3,'Score'))
```

```
  Specs   Score
0    RI    0.00
1    Na    4.31
2    Mg  100.98
3    Al   16.98
4    Si    0.11
5     K   31.67
6    Ca    3.21
7    Ba  145.51
8    Fe    2.17
  Specs   Score
7    Ba  145.51
2    Mg  100.98
5     K   31.67
3    Al   16.98
1    Na    4.31
6    Ca    3.21
  Specs  Score
0    RI   0.00
4    Si   0.11
8    Fe   2.17
```

As we can clearly see that glass type is dependent very less on RI,Si,Na,Ca and Fe; So we drop these as our independent variables

In [7]:

```
x1=glass[['Mg','Al','K','Ba']]
y1=glass['Type']
```

In [8]:

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x1,y1,test_size=.2)
```

In [9]:

```
from sklearn.linear_model import LinearRegression
glass_model=LinearRegression()
glass_model.fit(x_train,y_train)
```

Out[9]:

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=Fal
se)
```

In [10]:

```
glass_pred=glass_model.predict(x_test)
glass_pred
```

Out[10]:

```
array([0.66532506, 4.59324767, 1.91743076, 1.90903306, 1.87958473,
       1.63294166, 1.34835355, 3.75859773, 2.83875001, 1.52000162,
       4.68504302, 2.73883252, 3.12335417, 2.19287987, 1.89357535,
       0.58695494, 1.895069  , 1.44533448, 3.08529189, 2.10964264,
       1.08335397, 2.04931636, 1.15594584, 2.17201787, 1.56427647,
       2.45048329, 3.1540649 , 1.68139544, 1.88816166, 1.32429567,
       6.17726749, 1.69433551, 6.41244105, 6.47543366, 2.46814976,
       6.91527347, 1.80504834, 2.09316094, 6.84764354, 1.54905275,
       2.47748238, 0.23964751, 2.42266284])
```

In [11]:

```
glass_model.score(x_train,y_train)
```

Out[11]:

```
0.6441361139700343
```

In [12]:

```
glass_model.score(x_test,y_test)
```

Out[12]:

```
0.7103011127407388
```

In [13]:

```
#L1 Regularization of coefficients
from sklearn.linear_model import Ridge

from sklearn.model_selection import cross_val_score

# Train model with alpha=1
ridge = Ridge(alpha=1).fit(x_train, y_train)
print("Ridge Training score is",ridge.score(x_train,y_train))
print("Ridge Test score is",ridge.score(x_test,y_test))

# get cross val scores
score_train1 = cross_val_score(ridge,x_train,y_train, cv=5, scoring='r2')
score_test1 = cross_val_score(ridge,x_test,y_test, cv=5, scoring='r2')

print('CV Mean Train: ', np.mean(score_train1),'CV Mean Test: ', np.mean(score_test1))
print('STD Train: ', np.std(score_train1),'STD Test: ', np.std(score_test1))
print('\n')
```

```
Ridge Training score is 0.6440243680757285
Ridge Test score is 0.7161272223622435
CV Mean Train:  0.5862581547188336 CV Mean Test:  0.7200005979908718
STD Train:  0.1675057656237211 STD Test:  0.23637203120874498
```

In [14]:

```
#L2 Regularization of Coefficients
from sklearn.linear_model import Lasso

# Train model with  alpha=.01
lasso = Lasso(alpha=.01).fit(x_train, y_train)
print("Lasso Training score is",lasso.score(x_train,y_train))
print("Lasso Test score is",lasso.score(x_test,y_test))


# get cross val scores
score_train2 = cross_val_score(lasso,x_train,y_train, cv=5, scoring='r2')
score_test2 = cross_val_score(lasso,x_test,y_test, cv=5, scoring='r2')
print('CV Mean Train: ', np.mean(score_train2),'CV Mean Test: ', np.mean(score_test2))
print('STD Train: ', np.std(score_train2),'STD Test: ', np.std(score_test2))
print('\n')
```

```
Lasso Training score is 0.6438973664670815
Lasso Test score is 0.7151262388836981
CV Mean Train:  0.5840194406530568 CV Mean Test:  0.69342902597724
STD Train:  0.16870036057250085 STD Test:  0.22985018842548252
```