In [1]:

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

In [2]:

```
df = pd.read_csv (r'C:\\Users\\rezam\\Desktop\\New folder\\Datasets\\Q3\Communities_Crime.csv', he
ader=None)
```

The cleaning process of the dataset:

- The dataset had missing values in some columns. The values have been replaced the avrage of the values of the column.

In [3]:

```
df.shape
df.dtypes
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1994 entries, 0 to 1993
Columns: 127 entries, 0 to 126
dtypes: float64(127)
memory usage: 1.9 MB
```

The cleaning process of the dataset 2:

- The columns (indexes 0 & 3) had int64 datatypes which have been modified to float.

- The column 29 had a missing value (object), and it has ben replaced the the avrage of related column.

- The loops below indicate the indexes of those values.

In [4]:

```
df.columns
for col in df.columns:
    if str(df.iloc[:,col].dtypes) == 'object':
        print(col)
# df.iloc[:,1]
```

In [5]:

```
df.columns
for col in df.columns:
    if str(df.iloc[:,col].dtypes) == 'int64':
        print(col)
# df.iloc[:,1]
```

Data preparation:

- The imported dataset has been devided in 80% train and 20% test.

- The train dataset will be used for cross validation and the test data set will remain untouche for the final validation.

In [6]:

```
from numpy.random import RandomState
rng = RandomState()
train = df.sample(frac=0.8, random_state=rng)
```

```
test = df.loc[~df.index.isin(train.index)]
```

In [7]:

```
print(df.shape)
```

```
(1994, 127)
```

Observation:

- The 80% train dataset has been allocated for cross validation.

- The target has been st to the last column.

In [8]:

```
X_train = train.iloc[:,0:126].to_numpy()
y_target = train.iloc[:,-1:].to_numpy()
```

In [9]:

```
print(X_train.shape)
print(y_target.shape)
```

```
(1595, 126)
(1595, 1)
```

In [10]:

```
from sklearn.linear_model import LinearRegression
def get_mse(x,y_true):
    model=LinearRegression().fit(x, y_true)
    y_pred = model.predict(x)
    mse = sum([(y_p - y_t)**2 for y_p, y_t in zip(y_pred, y_true)])
    return mse
```

In [11]:

```
from sklearn.linear_model import RidgeCV

def get_mse_ridg(a, b):

    alphas = [1, 1e1, 1e2, 1e3, 1e6]
    regressor = RidgeCV(alphas=alphas, store_cv_values=True)
    regressor.fit(a, b)
    y_pred = regressor.predict(a)
    mse = get_mse(b, y_pred)
    print(mse)
```

First division

In [12]:

```
c_train1 = X_train[0:319]
c_test1 = y_target[0:319]
print(c_train1.shape)
print(c_test1.shape)
```

```
(319, 126)
(319, 1)
```

In [13]:

```
get_mse(x=c_train1,y_true=c_test1)
```

Out[13]:

```
array([3.2037962])
```

In [14]:

```
get_mse_ridg(a=c_train1,b=c_test1)
```

```
[3.43679052]
```

Second division

In [15]:

```
c_train2 = X_train[319:638]
c_test2 = y_target[0:319]
print(c_train1.shape)
print(c_test1.shape)
```

```
(319, 126)
(319, 1)
```

In [16]:

```
get_mse(x=c_train2,y_true=c_test2)
```

Out[16]:

```
array([10.26699951])
```

In [17]:

```
get_mse_ridg(a=c_train2,b=c_test2)
```

```
[0.06604287]
```

Third division

In [18]:

```
c_train3 = X_train[638:957]
c_test3 = y_target[0:319]
print(c_train1.shape)
print(c_test1.shape)
```

```
(319, 126)
(319, 1)
```

In [19]:

```
get_mse(x=c_train3,y_true=c_test3)
```

Out[19]:

```
array([10.7699543])
```

In [20]:

```
get_mse_ridg(a=c_train3,b=c_test3)
```

```
[0.03143466]
```

Fourth division

In [21]:

```
c_train4 = X_train[957:1276]
c_test4 = y_target[0:319]
print(c_train1.shape)
print(c_test1.shape)
```

```
(319, 126)
(319, 1)
```

In [22]:

```
get_mse(x=c_train4,y_true=c_test4)
```

Out[22]:

```
array([11.10061915])
```

In [23]:

```
get_mse_ridg(a=c_train4,b=c_test4)
```

```
[0.02774564]
```

Fifth division

In [24]:

```
c_train5 = X_train[1276:1595]
c_test5 = y_target[0:319]
print(c_train1.shape)
print(c_test1.shape)
```

```
(319, 126)
(319, 1)
```

In [25]:

```
get_mse(x=c_train5,y_true=c_test5)
```

Out[25]:

```
array([10.90688371])
```

In [26]:

```
get_mse_ridg(a=c_train5,b=c_test5)
```

```
[0.31656732]
```

Best fit for the divided 20% of the imported dataset.

In [27]:

```
X_train_test = test.iloc[:,0:126].to_numpy()
y_target_test = test.iloc[:,-1:].to_numpy()
```

In [28]:

```
c_train_final = X_train_test[0:319]
c_test_final = y_target_test[0:319]
print(c_train1.shape)
print(c_test1.shape)
```

```
(319, 126)
(319, 1)
```

In [29]:

```
get_mse(x=c_train_final,y_true=c_test_final)
```

Out[29]:

```
array([2.9892639])
```

In [30]:

```
get_mse_ridg(a=c_train_final,b=c_test_final)
```

```
[3.17243433]
```