The University of Hong Kong
Faculty of Engineering

Department of Civil Engineering Department

CIVL7018 Data Science for Civil Engineering

2023/24 2nd Semester

Assignment 1

Regression and Classification

Student Name: Ng Wing Fung

UID: 3035279208

Date of Submission: 29 February 2024

Q1

By Typed Calculation

Assume the simple linear model objective is Traffic Density (veh/km) and it takes in variable of only Traffic Volume (veh/hr). Let y be the objective and x be the variables.

$$X = [10, 35, 18, 45], Y = [500, 1200, 1000, 1800]$$

from equation 10 and 11 from lecture 2, w_0 and w_1 are estimated from the dataset X and Y:

$$w_1 = \frac{\sum_{n=1}^{N} x_n y_n - N\bar{x}\bar{y}}{\sum_{n=1}^{N} (x_n)^2 - N(\bar{x})^2}$$
$$w_0 = \bar{y} - w_1\bar{x}$$

Substituting the dataset of X and Y

$$w_1 = \frac{(10 \times 500 + 35 \times 1200 + 18 \times 1000 + 45 \times 1800) - 4 \times (\frac{10+35+18+45}{4}) \times (\frac{500+1200+1000+1800}{4})}{(10^2 + 35^2 + 18^2 + 45^2) - 4 \times (\frac{10+35+18+45}{4})^2}$$
$$= 32.322$$
$$w_0 = \frac{500 + 1200 + 1000 + 1800}{4} - 32.322 \times \frac{500 + 1200 + 1000 + 1800}{4}$$
$$= 252.309$$

By Jupyter Notebook

Assume the simple linear model objective is Traffic Density (veh/km) and it takes in variable of only Traffic Volume (veh/hr). Let y be the objective and x be the variables. Declare the x and y dataset with the values taken from Table 1:

In [1]:
```
X = [10,35,18,45]
Y = [500,1200,1000,1800]
```

Conduct some basic statistic analysis

In [2]:
```
x_mean = sum(X)/len(X)
y_mean = sum(Y)/len(Y)
N = len(X)
sum_xy = sum([i*j for i,j in zip(X,Y)])
sum_x_sq = sum([i**2 for i in X])
```

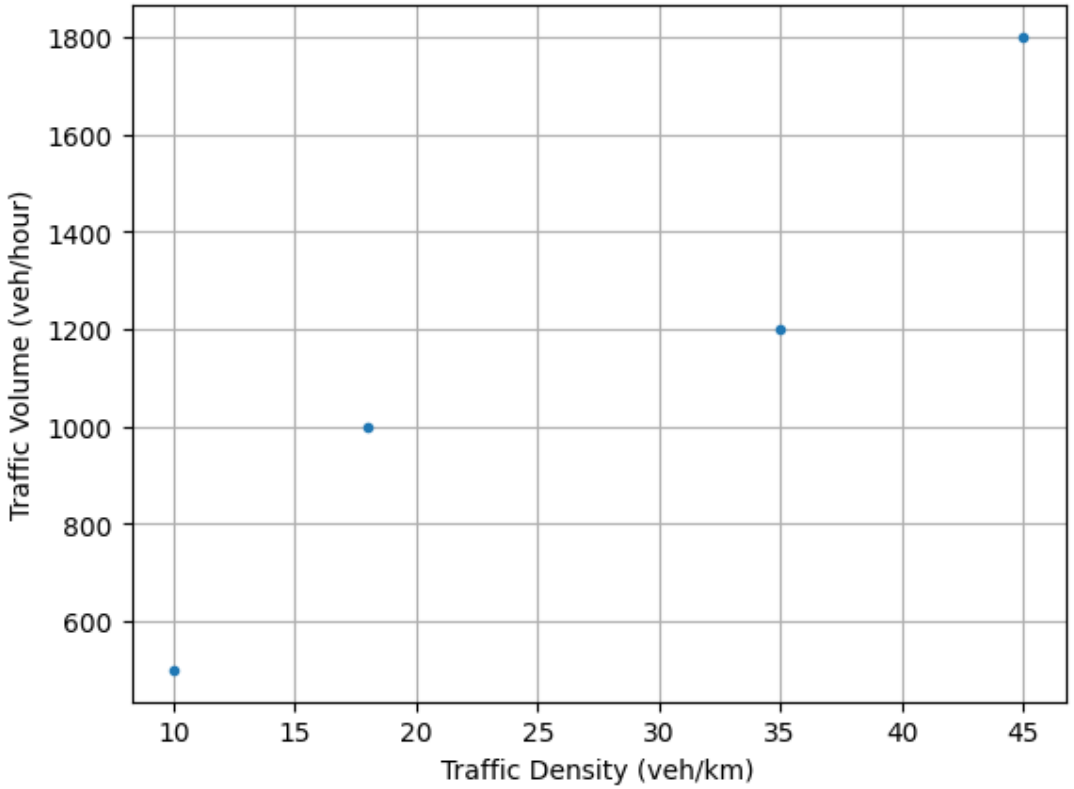
Using the closed form solution formula (Eq. 9 from Lec. 2 slide 9) to estimate the linear model parameter w0 and w1

In [8]:
```
w1 = (sum_xy - N*x_mean*y_mean)/(sum_x_sq-N*x_mean*x_mean)
w0 = y_mean - w1*x_mean
print(f'coefficient w0 and w1 are {w0:.3f} and {w1:.3f} respectively')
```

```
coefficient w0 and w1 are 252.309 and 32.322 respectively
```

Plot out data and the linear fitted model

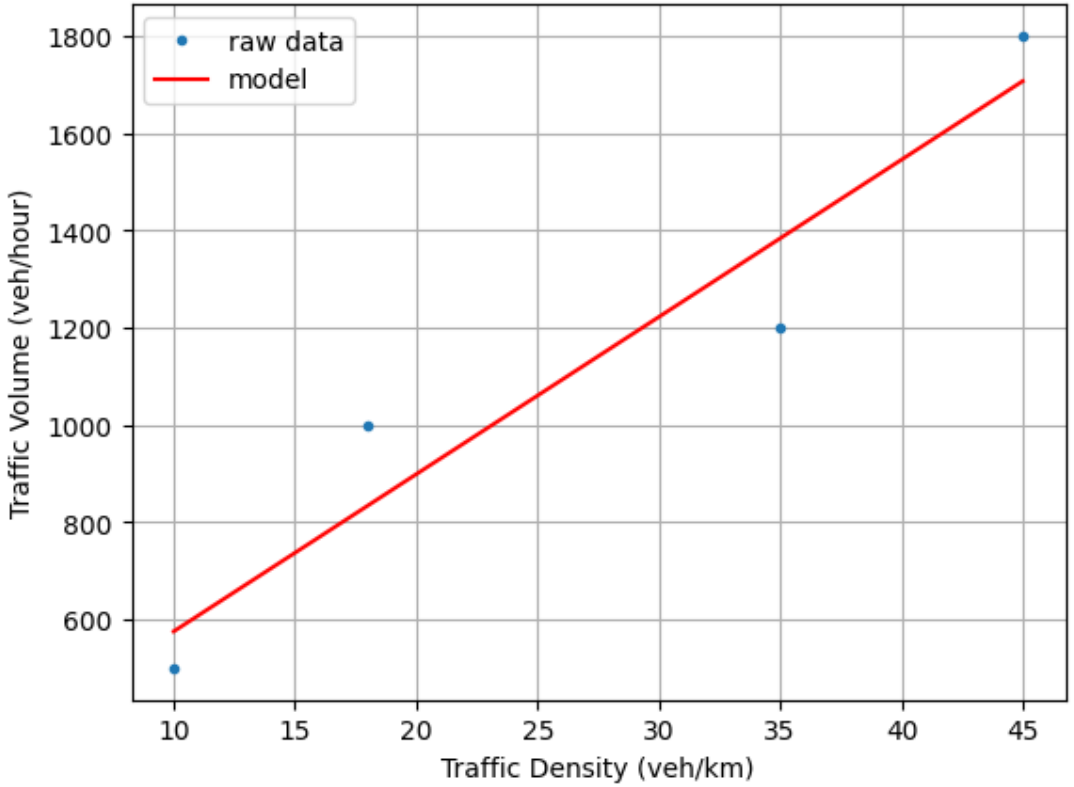
In [4]:
```
import matplotlib.pyplot as plt

plt.plot(X, Y, '.', label='raw data')
plt.xlabel('Traffic Density (veh/km)')
plt.ylabel('Traffic Volume (veh/hour)')
plt.grid()
```



Adding model in red for comparison

In [5]:
```
plt.plot(X, Y, '.', label='raw data')
plt.plot([min(X),max(X)], [min(X)*w1+w0,max(X)*w1+w0], '-r', label='model')
plt.xlabel('Traffic Density (veh/km)')
plt.ylabel('Traffic Volume (veh/hour)')
plt.grid()
plt.legend(loc=0)
```

Out[5]: <matplotlib.legend.Legend at 0x2534a33a560>



If the predicted traffic using this arterial is 1000 veh/hour, what is the estimated traffic density?

In [10]:
```
print(f"estimated traffic density for traffic volume of 1000 veh/hour is {1000*w1+w0:.3f} veh/km")
```

```
estimated traffic density for traffic volume of 1000 veh/hour is 32574.208 veh/km
```

Q 2.1.1

From dataset the dependent variable of the samples y_n is the total in flow for each station

$$Y = \begin{bmatrix} 3459623 & 3914019 & 8100630 & 13460142 & 2535732 \end{bmatrix}$$

the independent variable of the samples x_n consists of 4-dimensional data& which extends its value in the axis of Total Population near each station (x_{n1}) number of households that own 0 vehicles (x_{n2}) total employment (x_{n3}) and total road network density (x_{n4}). In each axis& the x_{nj} is a column vector containing the data value of each sample in the dataset.

$$X = \begin{bmatrix} x_{n1}^T & x_{n2}^T & x_{n3}^T & x_{n4}^T \end{bmatrix}$$
$$= \begin{bmatrix} 50383 & 4784 & 28318 & 28.7 \\ 11084 & 1664 & 33120 & 42.23 \\ 51122 & 16059 & 61815 & 36.3 \\ 25970 & 5383 & 181995 & 40.15 \\ 29222 & 2891 & 23981 & 31.3 \end{bmatrix}$$

Adding a column vector of 1 to the independent variable matrix to introduct an intercept to the model

$$X = \begin{bmatrix} 1 & 50383 & 4784 & 28318 & 28.7 \\ 1 & 11084 & 1664 & 33120 & 42.23 \\ 1 & 51122 & 16059 & 61815 & 36.3 \\ 1 & 25970 & 5383 & 181995 & 40.15 \\ 1 & 29222 & 2891 & 23981 & 31.3 \end{bmatrix}$$

Since the model is built as a linear regression model, there is the coefficient matrix w, which by taking dot product to the independent variable will give a close proximation to the dependent variable

$$y_n \approx w^T x_n$$

We obtains the close proximation of the model by maximum likeihood estimate, i.e. minimizing the error function

$$E(w) = \frac{1}{2} \sum_{n=1}^{N} [y_n - w^T x_n]^2$$

Using the close form solution formula (25) for lecture 2, obtained by taking derivitive to the error function and setting it equal 0

$$w = (X^T X)^{-1} X^T Y$$
$$= [5288434.545, 39.269, 127.973, 59.180, 156149.881]^T$$

Q2.1.2

Declare the objective matrix and the sample matrix

In [1]:
```
import numpy as np
np.set_printoptions(suppress=True)

X1 = np.array([[1,50383,4784,28318,28.7],[1,11084,1664,33120,42.23],
               [1,51122,16059,61815,36.3],[1,25970,5383,181995,40.15],
               [1,29222,2891,23981,31.3]])
Y1 = np.array([3459623,3914019,8100630,13460142,2535732]).reshape(-1,1)
```

Solving the MLE with the matrix product

In [6]:
```
g1 = X1.T@X1
g2 = np.linalg.inv(g1)
g3 = X1.T @ Y1
w = np.linalg.inv(X1.T@X1)@X1.T @ Y1
print("The w matrix with 5 variables [w0, w1, w2, w3, w4] is " , w.flatten())
```

```
The w matrix with 5 variables [w0, w1, w2, w3, w4] is  [-5288434.54549981       39.26866197      127.97287239
59.18005265
   156149.88142737]
```

Solving the MLE with the sklearn regression model

In [7]:
```
from sklearn.linear_model import LinearRegression

X2 = np.array([[50383,4784,28318,28.7],[11084,1664,33120,42.23],
               [51122,16059,61815,36.3],[25970,5383,181995,40.15],
               [29222,2891,23981,31.3]])
Y2 = np.array([3459623,3914019,8100630,13460142,2535732])

reg = LinearRegression().fit(X2, Y2)

print('[w1, w2, w3, w4] is', reg.coef_)
print('w0 is', reg.intercept_)
```

```
[w1, w2, w3, w4] is [    39.26866197    127.97287239     59.18005265 156149.88142737]
w0 is -5288434.545499415
```

Q2.1.3

In [4]:
```
X3 = np.array([1, 34689, 9443, 148355, 38.9])
Y3 = w.T@X3
print('estimated rideship inflow at Montgomery is', Y3[0])
```

```
estimated rideship inflow at Montgomery is 12136091.00253777
```

Q2.2.1

True

Q2.2.2

True

Q2.2.3

from definition, $\hat{y} = Xw$

the difference vector of $\hat{y}$ and y = $y - \hat{y} = y - Xw$

Since $y - \hat{y}$ is orthogonal to X,

$$X^T(y - \hat{y}) = 0$$
$$X^T(y - Xw) = 0$$
$$X^T y - X^T Xw = 0$$
$$X^T Xw = X^T y$$
$$w = (X^T X)^{-1} X^T y$$

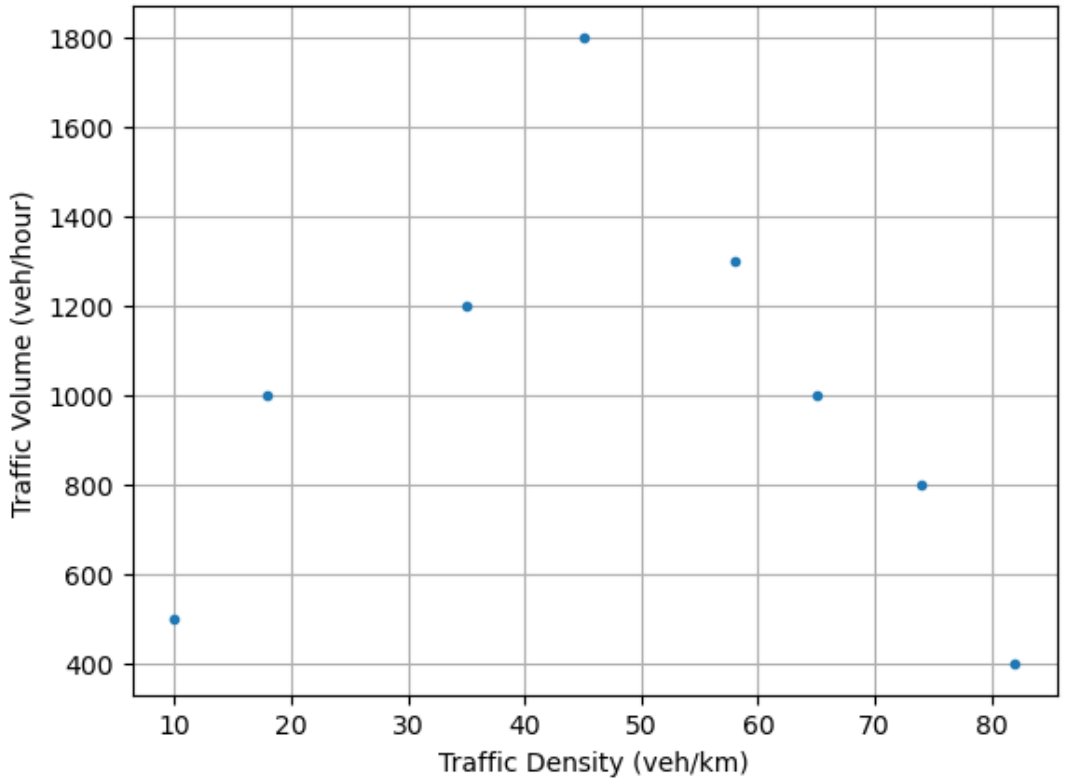
Q3

Update the sample data set inherited from Q1.

In [2]:
```
import numpy as np
X = np.array([10,35,18,45,58,65,82,74])
Y = np.array([500,1200,1000,1800,1300,1000,400,800])
```

Ploting the data

In [3]:
```
import matplotlib.pyplot as plt

plt.plot(X, Y, '.', label='raw data')
plt.xlabel('Traffic Density (veh/km)')
plt.ylabel('Traffic Volume (veh/hour)')
plt.grid()
```



Since the model to be built is in 2nd order derivitive, create the design matrix up to 2nd order

In [6]:
```
PHI = np.vander(X,3,increasing=True)
print (PHI)
```

```
[[   1   10  100]
 [   1   35 1225]
 [   1   18  324]
 [   1   45 2025]
 [   1   58 3364]
 [   1   65 4225]
 [   1   82 6724]
 [   1   74 5476]]
```

Using the close form matrix solution to solve for w

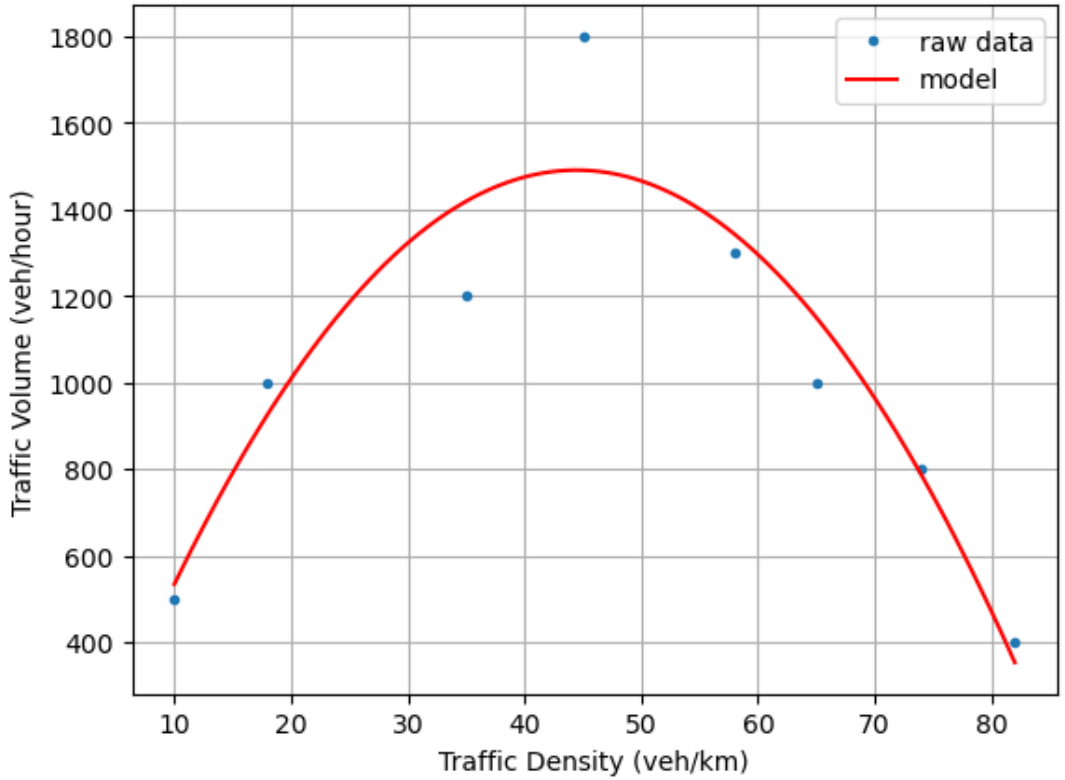
In [7]:
```
w = np.linalg.inv(PHI.T@PHI)@PHI.T @ Y.reshape(-1,1)
print(w)
```

```
[[-102.69284032]
 [  71.70391366]
 [  -0.8067115 ]]
```

Plot out the data

In [8]:
```
import matplotlib.pyplot as plt

plt.plot(X, Y, '.', label='raw data')

# create 1000 equally spaced points between -10 and 10
px = np.linspace(min(X), max(X), 100)
py = np.vander(px,3,increasing=True)@w
plt.plot(px,py, '-r', label='model')
plt.xlabel('Traffic Density (veh/km)')
plt.ylabel('Traffic Volume (veh/hour)')
plt.grid()
plt.legend(loc=0)
```

Out[8]: <matplotlib.legend.Legend at 0x2545178a650>



If the predicted traffic using this arterial is 1000 veh/hour, what will be the estimated traffic density?

In [12]:
```
quad_equation=np.flip(w.reshape(-1))-[0,0,1000]
ans1 = np.roots(quad_equation)
print(f'The traffic density for that predict the traffic volume at 1000 veh/hour are {ans1[0]:.3f} veh/km and {ans1[1]:.3
```

```
The traffic density for that predict the traffic volume at 1000 veh/hour are 69.104 veh/km and 19.780 veh/km
```

What will be the traffic volume if the density goes to 90 veh/km?

In [10]:
```
ans2 = np.vander(np.full(1,90),3,increasing=True)@w
print(ans2[0])
```

```
[-183.70373998]
```

The predicted traffic volume is outside the estimated maximum traffic capacity of the road

Q4.1

In [7]:
```
from sklearn import linear_model
from sklearn.preprocessing import minmax_scale
import pandas as pd
import numpy as np
```

In [8]:
```
data_BART_sld = pd.read_csv('data_X.csv').iloc[:48, 1:]
```

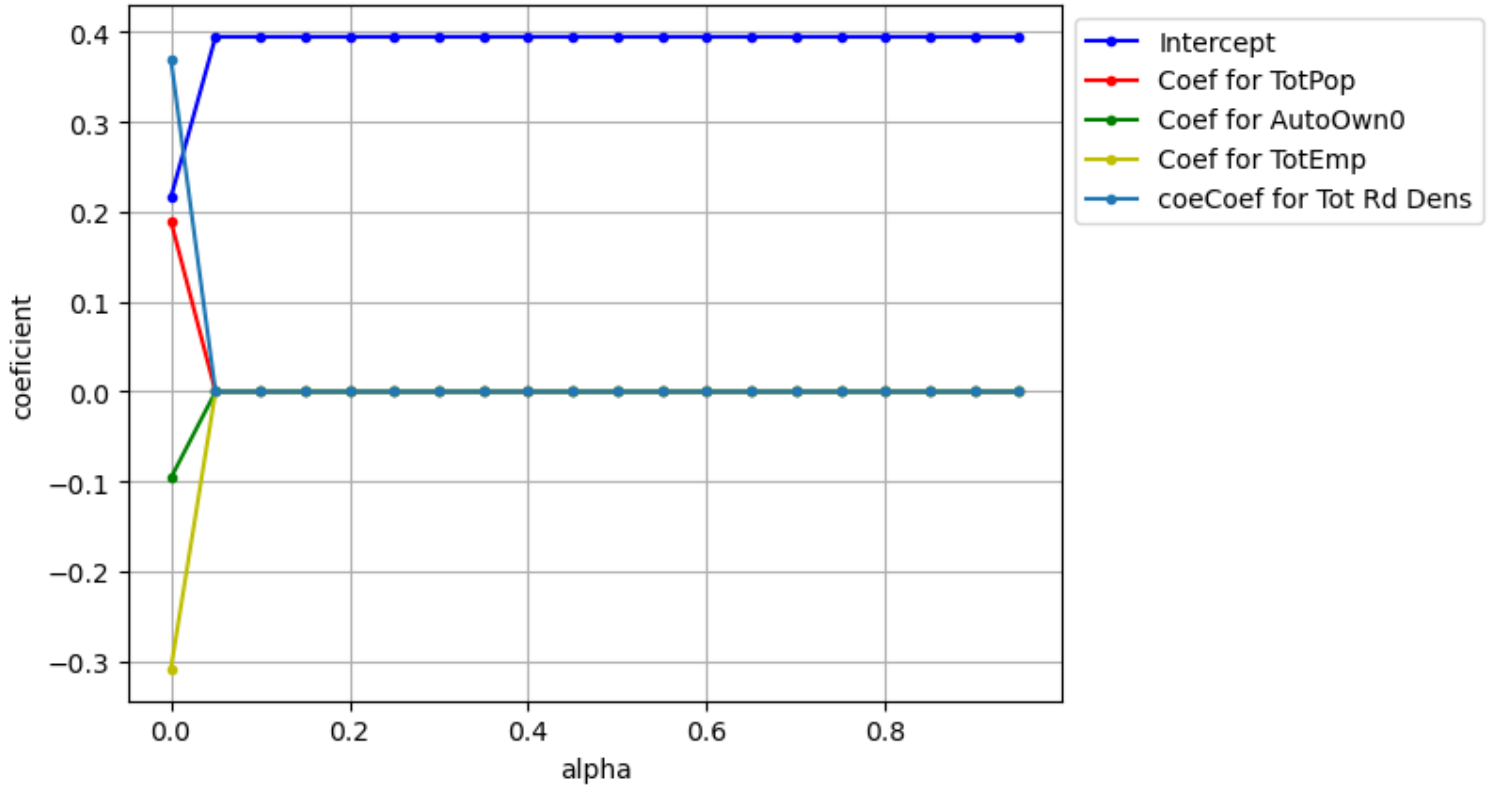
In [9]:
```
OD_BART = np.load('3d_daily.npy').sum(axis=2)[17, :48]
```

In [10]:
```
def get_w(alpha):
    X = minmax_scale(data_BART_sld)
    y = minmax_scale(OD_BART)

    reg = linear_model.ElasticNet(alpha=alpha)
    reg.fit(X, y)

    return reg.intercept_, *reg.coef_[:4]
```

In [20]:
```
import warnings
warnings.filterwarnings('ignore')

cols = ['Intercept', 'Coef for TotPop', 'Coef for AutoOwn0', 'Coef for TotEmp', 'coeCoef for Tot Rd Dens']
w = [[] for _ in cols]

alpha_set = np.arange(0, 1.0, 0.05)
for alpha in alpha_set:
    for t,l in zip(get_w(alpha), w):
       l.append(t)
```

In [21]:
```
import matplotlib.pyplot as plt

style = ['b.-', 'r.-', 'g.-', 'y.-', '.-']
for i, d in enumerate(w):
    plt.plot(alpha_set, d, style[i], label=cols[i])
plt.grid()
plt.legend(bbox_to_anchor=(1,1))
plt.xlabel('alpha')
plt.ylabel('coeficient')
```

Out[21]:
```
Text(0, 0.5, 'coeficient')
```




```
# Q4.2
```

No, as shown in the example, the coefficient all reduce to 0 if we apply a strong regularization, render the model to only gives out constant as regression result.

5.1

feature of model -> The output is discrete, only binary output,

output -> the class of the input sample, in only true of false (1 or 0)

discriminant function ->

$$g(x) = w^T x = w_0 + w_1 x_1 + w_2 x_2$$

where y_n is the dependent variable of the sample being assessed as congested

x_n is for independent variable of the sample (x_0 as intercept (=1 for all cases), x_1 for Traffic Density, x_2 for Traffic Volume)

w is the parameters (w_0 as intercept, remaining w_n correspond to their specific x_n)

5.2

from the model parameter, it can be deduce that

$$y_n = 1, x_0 = 1, x_1 = 30, x_2 = 1000$$

The equation of the model for the probability is

$$P(y_n = 1|x_n) = \frac{e^{w_0 + w^T x_n}}{1 + e^{w_0 + w^T x_n}} = \frac{e^{w_0 + 30w_1 + 1000w_2}}{1 + e^{w_0 + 30w_1 + 1000w_2}}$$

5.3

Joint probability

$$L(w_0, w_1, w_2) = \frac{e^{w_0+50w_1+2000w_2}}{1 + e^{w_0+50w_1+2000w_2}} \cdot \frac{1}{1 + e^{w_0+20w_1+900w_2}} \cdot \frac{e^{w_0+40w_1+1500w_2}}{1 + e^{w_0+40w_1+1500w_2}} \cdot \frac{1}{1 + e^{w_0+10w_1+500w_2}} \cdot \frac{1}{1 + e^{w_0+5w_1+300w_2}}$$

5.4

Cross-entropy

$$E(w_0, w_1, w_2) =$$
$$-logL(w_0, w_1, w_2) =$$
$$-[log(\frac{e^{w_0+50w_1+2000w_2}}{1+e^{w_0+50w_1+2000w_2}}) + log(\frac{1}{1+e^{w_0+20w_1+900w_2}}) + log(\frac{e^{w_0+40w_1+1500w_2}}{1+e^{w_0+40w_1+1500w_2}}) + log(\frac{1}{1+e^{w_0+10w_1+500w_2}}) + log(\frac{1}{1+e^{w_0+5w_1+300}}$$

Taking derivitive to the cross-entropy, and using equation (21) from lecture 3

$$\frac{\partial E}{\partial w_j} = \sum_{n=1}^{n=5} (y_n - z_n)x_{nj}$$

Calculate the individual w_n from 0 to 2

$$\frac{\partial E}{\partial w_0} = x_0[(1-z_1) + (-z_2) + (1-z_3) + (-z_4) + (-z_5)]$$
$$= (\frac{1}{1+e^{w_0+50w_1+2000w_2}} - \frac{e^{w_0+20w_1+900w_2}}{1+e^{w_0+20w_1+900w_2}} + \frac{1}{1+e^{w_0+40w_1+1500w_2}} - \frac{e^{w_0+10w_1+500w_2}}{1+e^{w_0+10w_1+500w_2}} - \frac{e^{w_0+5w_1+300w_2}}{1+e^{w_0+5w_1+300w_2}})$$

$$\frac{\partial E}{\partial w_1} = [x_{11}(1-z_1) + x_{21}(-z_2) + x_{31}(1-z_3) + x_{41}(-z_4) + x_{51}(-z_5)]$$
$$= (\frac{50}{1+e^{w_0+50w_1+2000w_2}} - \frac{20e^{w_0+20w_1+900w_2}}{1+e^{w_0+20w_1+900w_2}} + \frac{40}{1+e^{w_0+40w_1+1500w_2}} - \frac{10e^{w_0+10w_1+500w_2}}{1+e^{w_0+10w_1+500w_2}} - \frac{5e^{w_0+5w_1+300w_2}}{1+e^{w_0+5w_1+300w_2}})$$

$$\frac{\partial E}{\partial w_2} = [x_{12}(1-z_1) + x_{22}(-z_2) + x_{32}(1-z_3) + x_{42}(-z_4) + x_{52}(-z_5)]$$
$$= (\frac{2000}{1+e^{w_0+50w_1+2000w_2}} - \frac{900e^{w_0+20w_1+900w_2}}{1+e^{w_0+20w_1+900w_2}} + \frac{1500}{1+e^{w_0+40w_1+1500w_2}} - \frac{500e^{w_0+10w_1+500w_2}}{1+e^{w_0+10w_1+500w_2}} - \frac{300e^{w_0+5w_1+300w_2}}{1+e^{w_0+5w_1+300w_2}})$$

5.5

Initalize

$$w_0 = 1, w_1 = 1, w_2 = 1,$$

discriminant function

$$g(x) = w_0 + w_1x_1 + w_2x_2$$

and

$$z = \frac{1}{1 + e^{-(w_0 + w_1 x_1 + w_2 x_2)}}$$

Setting learning rate

$$\eta = 0.01$$

calculate the cross-entropy loss to w,

$$\Delta w_0 = \eta \sum_{n=1}^{5} (y_n - z_n) = 0.01([1 - \frac{1}{1 + e^{-(1+50+2000)}}] + [0 - \frac{1}{1 + e^{-(1+20+900)}}] + [1 - \frac{1}{1 + e^{-(1+40+1500)}}] + [0 - \frac{1}{1 + e^{-(1+10+500)}}]$$
$$+ [0 - \frac{1}{1 + e^{-(1+5+300)}}])$$
$$= 0.01 \times -3 = -0.03$$
$$\Delta w_1 = \eta \sum_{n=1}^{5} (y_n - z_n) x_{n1} = 0.01(50[1 - \frac{1}{1 + e^{-(1+50+2000)}}] + 20[0 - \frac{1}{1 + e^{-(1+20+900)}}] + 40[1 - \frac{1}{1 + e^{-(1+40+1500)}}] + 10[0 - \frac{1}{1 + e^{-(}}$$
$$+ 5[0 - \frac{1}{1 + e^{-(1+5+300)}}])$$
$$= 0.01 \times -35 = -0.35$$
$$\Delta w_2 = \eta \sum_{n=1}^{5} (y_n - z_n) x_{n2} = 0.01(2000[1 - \frac{1}{1 + e^{-(1+50+2000)}}] + 900[0 - \frac{1}{1 + e^{-(1+20+900)}}] + 1500[1 - \frac{1}{1 + e^{-(1+40+1500)}}]$$
$$+ 500[0 - \frac{1}{1 + e^{-(1+10+500)}}] + 300[0 - \frac{1}{1 + e^{-(1+5+300)}}])$$
$$= 0.01 \times -1700 = -17$$

At the 2nd iteration, w_0, w_1, w_2 are 0.97,0.65 and -16 respectively

6.1

1. the data point lie on the margin in the correct side
2. the data point lie between the hyperpane and the margin but in the correct side
3. the data point lie on the wrong side of the hyperpane
4. left: C=0.1, right: C=10

6.2

1. support vector are the vectors of the data points which has the closest distance to the hyperplane for classification of data.
2. since x_0 and x_1 lies on the hyperpane H, they satisfy the definition of H, i.e.

$$w^T x_0 = b$$
$$w^T x_1 = b$$

consider the dot product of vectors $(x_1 - x_0)$ and w,

$$w \cdot (x_1 - x_0) = w \cdot x_1 - w \cdot x_0 = w^T x_0 - w^T x_1 = b - b = 0$$

which by definition dot product for 2 vectors $A \cdot B = |A| cos\theta |B|$

Assume the Hyperplane H is properly defined, hence $w \neq (0, 0, 0)$,

and the two points x_0 and x_1 are two seperate points,

The two vector in length, $|(x_1 - x_0)|$ and $|w|$ must be greater than 0,

thus $w \cdot (x_1 - x_0) = |w| cos\theta |x_1 - x_0|$ must imply the angle between $(x_1 - x_0)$ and w are perpendicular such that $cos\theta = 0$

3. since distance D between a point $z(x_0, x_1, x_2)$ and a plane H is measure on a line passing through z and perpendicular to plane H,

we first find the vector normal to the plane H,

which from Q2., vector w is normal to any two points given on the plane.

Reduce the length of the noraml vector to 1 such that it forms the unit vector in the direction normal to the polane H

$$n = \frac{w}{|w|}$$

Next consider a vector A starting from any arbitary point $p(x_1, y_1, z_1)$ on the plane H pointing towards z

$$A = p - z = (x_1 - x_0, y_1 - y_0, z_1 - z_0)$$

we project the vector A to the normal vector n by taking dot product of A and the normal vector n, which will give out the distance D between the point z to on a line perpendicular to plane H as we are finding

$$\begin{aligned} D &= |A \cdot n| \\ &= \frac{|(p-z) \cdot w|}{|w|} \\ &= \frac{|w^T p - w^T z|}{|w|} \end{aligned}$$

Since p lies on plane H, by the plane definition $w^T p = b$

$$\begin{aligned} D &= \frac{|-b - w^T z|}{|w|} \\ &= \frac{|b + w^T z|}{|w|} \end{aligned}$$

Assume w is defined as (p, q, r), we can expand the distance formula as follow:

$$\begin{aligned} D &= \frac{|b + [p, q, r]^T [x_0, y_0, z_0]|}{\sqrt{p^2 + q^2 + r^2}} \\ &= \frac{px_0 + qy_0 + rz_0 + b}{\sqrt{p^2 + q^2 + r^2}} \end{aligned}$$

7.1

the depth d tree contain d+1 layers of node with layer 0 as the initial status with only one root node.

for the progression of next layer, each node of the preceding layer may assessed and if not meeting a predefined criterion, split into two new nodes

Suppose in the Layer L_d at depth d there are $n_d = g(d)$ nodes , the next layer L_{d+1} will have at most $2n$ nodes assuming every decision node in L_d is being split.

therefore at the decision tree with most nodes at L_d

$$n_d = \sum_{d=0}^{i=d} g(i) = g(d) + \sum_{d=0}^{i=d-1} g(i)$$

iterate this recurrsively and at $d = 0$, we have the root node only hence $g(0) = 1$, we have

$$\begin{aligned} n_d &= g(0) + g(1) + \ldots + g(d) \\ &= g(0) + 2g(0) + \ldots + 2^d g(0) \\ &= \sum_{d=0}^{i=d} 2^i \end{aligned}$$

Using the power sum formula

$$n_d = \frac{2^{d+1} - 1}{2 - 1} = 2^{d+1} - 1$$

7.2

for the layer L_d at depth d having the lower bound m nodes,

the layer will have its maximum node number greater than m,

$$\begin{aligned} n_{d-1} = 2^{d+1} - 1 &> m \\ 2^{d+1} &> m + 1 \end{aligned}$$

Taking log on both sides and rearrange the equation

$$d > \log_2(m + 1) - 1$$

7.3

Consider a tree which at each level, there is only one node being bifurcated, and there is one node being bifurcated contains only one element. Doing so will eliminate the least element out from its mother node, and the remaining root will have the most element for further proceeding of depth. Since for each level, the node with max number of element is 1 element less than the previous depth, it take m-1 depth to deplet all nodes with more than 1 element into node with 1 element. Hence the max depth for m samples is d+1.

For the lowest depth possible of a tree, each depth processed will double the node number as in the previous depth. At depth of d, there will be at most 2^d nodes. Thus for a sample size of m, it will require at least $m \leq 2^d$ depth for all samples to be isolated in its own node.

In short, the range of d for sample size of m will be in range:

$$log_2(m) \leq d \leq m - 1$$