DATA7202 : Assessment 2
Name : Peng Yu
Student ID : 46635884

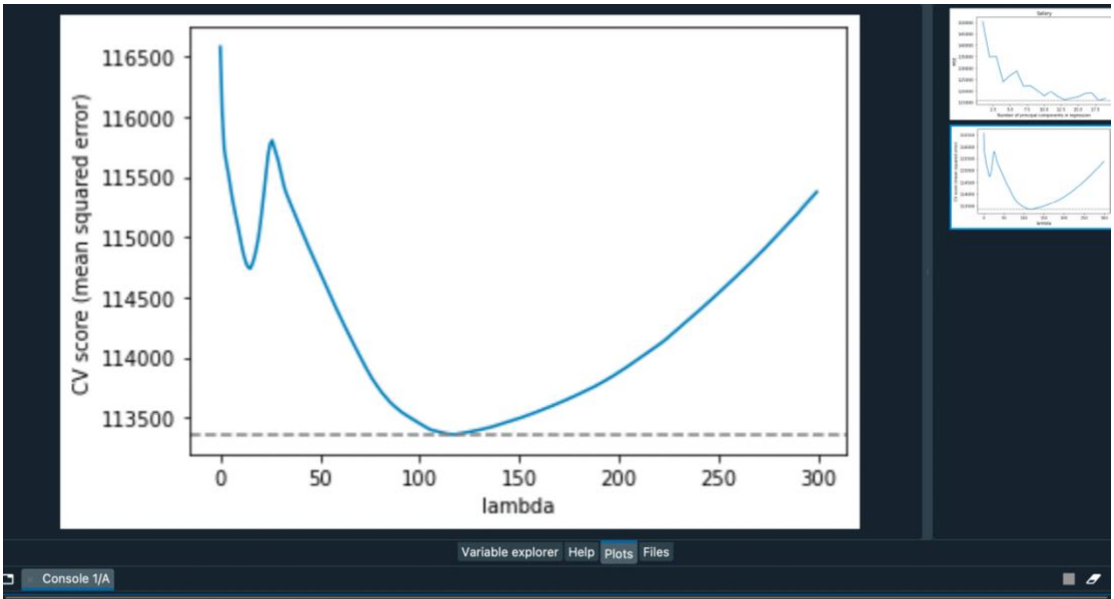
Q1:

Answer (a):

Through drawing and calculation, the optimal number of components is 18, and the corresponding mean squared error is 115733.66222777616.

Answer (b):

Through drawing and calculation, the best λ is 117, and the corresponding mean squared error is 113358.4033198368.




```
The best lambda is 117. The corresponding mean squared
error = 113358.4033198368
```

Q2:

Answer:

The coefficient for type is -0.2237, and the corresponding 95% Cis is (-0.317, -0.130).

The coefficient for construction is 0.3714, and the corresponding 95% Cis is (0.255, 0.488).

The coefficient for operation is 0.7680, and the corresponding 95% Cis is (0.567, 0.969).

The coefficient for months is 8.095* 10^{-5}, and the corresponding 95% Cis is (7.54*10^{-5}, 8.65*10^{-5}).

```
                 Generalized Linear Model Regression Results
==============================================================================
Dep. Variable:                 damage   No. Observations:                   34
Model:                            GLM   Df Residuals:                       30
Model Family:                 Poisson   Df Model:                            3
Link Function:                    log   Scale:                          1.0000
Method:                          IRLS   Log-Likelihood:                -145.96
Date:                Wed, 14 Apr 2021   Deviance:                       194.06
Time:                        21:01:19   Pearson chi2:                     178.
No. Iterations:                     6
Covariance Type:            nonrobust
================================================================================
                   coef    std err          z      P>|z|      [0.025      0.975]
--------------------------------------------------------------------------------
type            -0.2237      0.048     -4.693      0.000      -0.317      -0.130
construction     0.3714      0.060      6.231      0.000       0.255       0.488
operation        0.7680      0.103      7.471      0.000       0.567       0.969
months        8.095e-05   2.84e-06     28.487      0.000    7.54e-05    8.65e-05
================================================================================

***Repl Closed***
```

Q3:

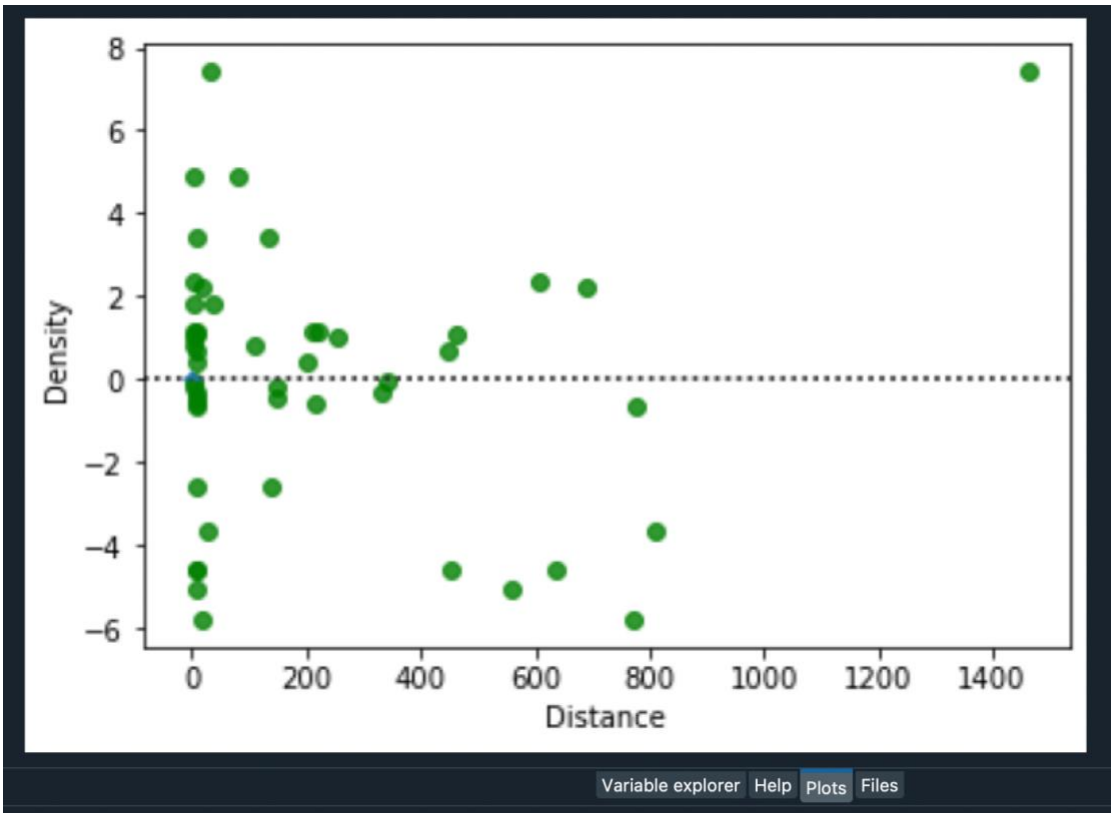
Answer (a):

The estimated residual standard deviation is 3.2594734475800946, the p-value for Cases is approximately 0, and the p-value for Distance is 0.001.
R-squared of this model is 0.960, it is mean that the fitting degree of this model can reach 96.0%, indicating that the fitting degree is good. P-value is $4.69*10^{-16}$, therefore, we can think that the number of cases of product stocked and the distance walked by the route driver can better predict the delivery time. The smaller p-value corresponding to Case indicates that the model with Cases have predictive advantages over the model without Cases, same as the model with Distance have predictive advantages over the model without Distance.

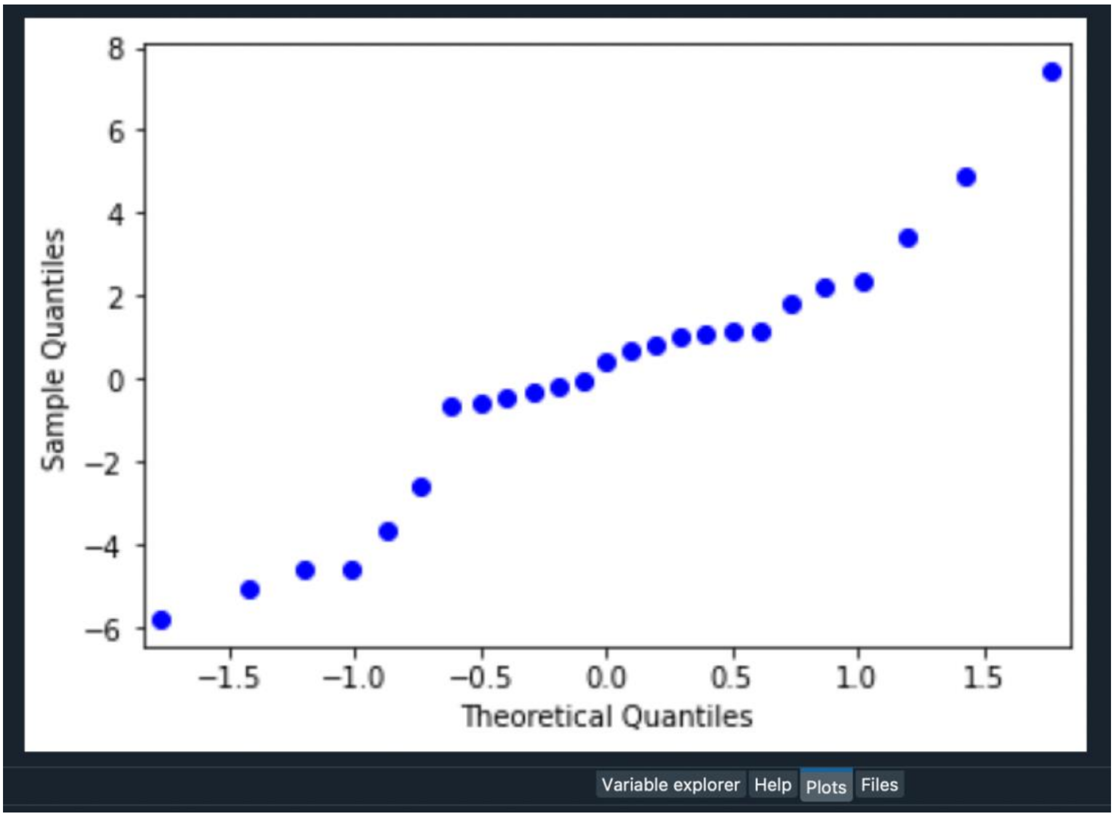
```
                            OLS Regression Results
==============================================================================
Dep. Variable:                   Time   R-squared:                       0.960
Model:                            OLS   Adj. R-squared:                  0.956
Method:                 Least Squares   F-statistic:                     261.2
Date:                Wed, 14 Apr 2021   Prob (F-statistic):           4.69e-16
Time:                        21:17:41   Log-Likelihood:                -63.415
No. Observations:                  25   AIC:                             132.8
Df Residuals:                      22   BIC:                             136.5
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      2.3412      1.097      2.135      0.044       0.067       4.616
Cases          1.6159      0.171      9.464      0.000       1.262       1.970
Distance       0.0144      0.004      3.981      0.001       0.007       0.022
==============================================================================
Omnibus:                        0.421   Durbin-Watson:                   1.170
Prob(Omnibus):                  0.810   Jarque-Bera (JB):                0.010
Skew:                           0.032   Prob(JB):                        0.995
Kurtosis:                       3.073   Cond. No.                         873.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
The standard error of estimate:  3.2594734475800946
```

Answer (b):

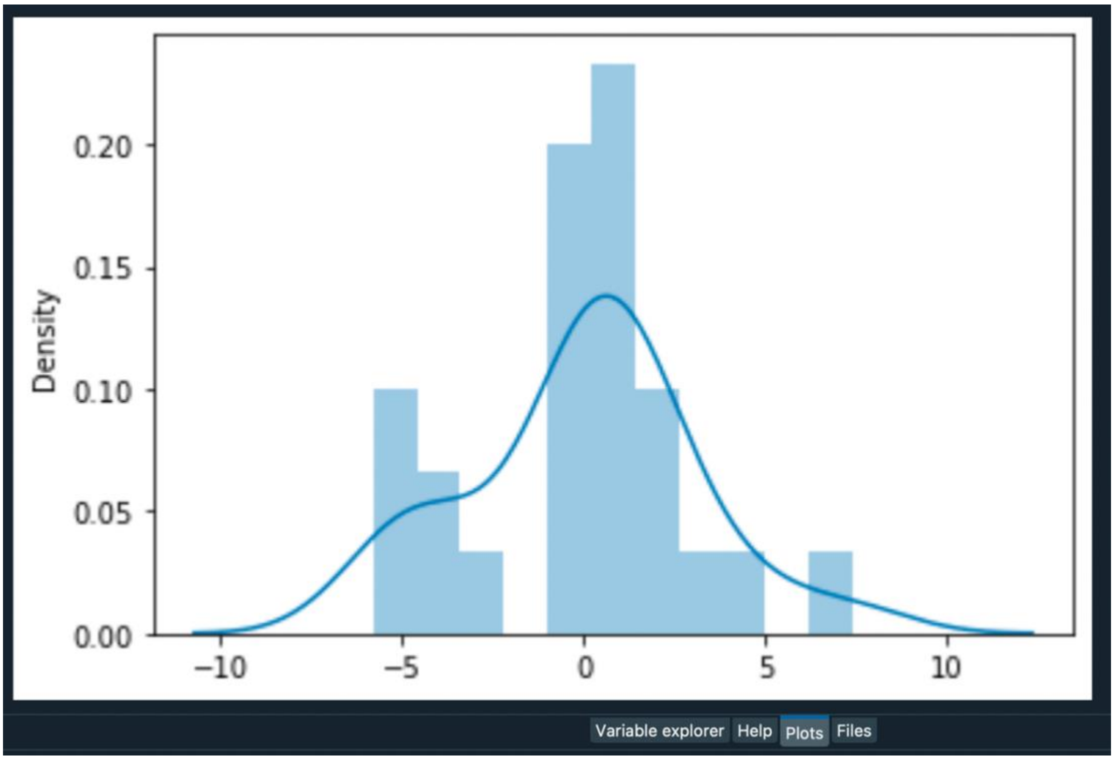
For a good model, its residuals should be points with randomness and unpredictability, forming a graph close to the normal distribution. However, in our residual graph, the residuals are not distributed very evenly on both sides of 0.



So we plotted q-qplot. If the residuals conform to the normal distribution, the points in q-qplot would form a straight line state. But our point is obviously not, so our residuals don't fit a normal distribution.

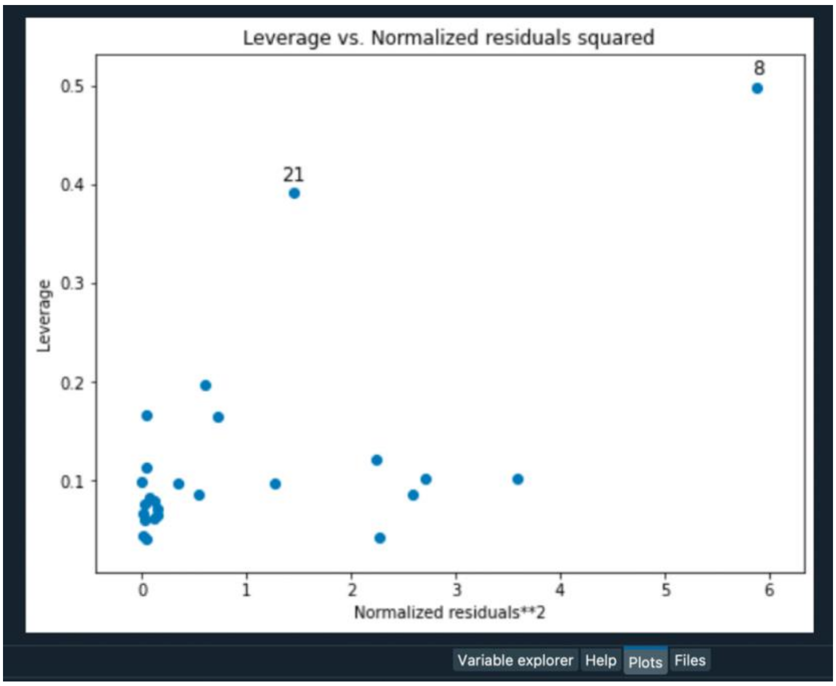


Finally, we draw the histogram of the residuals to prove our conclusion.



Answer (c):

The most influential observation is 8, and the second most influential observation is number 21.



Q4:

Answer:

$\mathbb{P}$(y|θ) = $\frac{1}{\theta}$

$\mathbb{P}$(Y|θ) = $\prod_{i=1}^{n} \mathbb{P}(y_i|\theta)$ = $\prod_{i=1}^{n} \frac{1}{\theta}$ = $\frac{1}{\theta^n}$

$\mathbb{P}$(θ) = $\frac{\alpha\theta_m^\alpha}{\theta^{\alpha+1}}$ $\quad \theta \geq \theta_m$

0 $\quad \theta < \theta_m$

$\mathbb{P}$(θ|Y) = $\frac{\mathbb{P}(Y|\theta) * \mathbb{P}(\theta)}{\mathbb{P}(Y)}$

$\propto \mathbb{P}(Y|\theta) * \mathbb{P}(\theta)$

= $\frac{\alpha\theta_m^\alpha}{\theta^{\alpha+1}} * \frac{1}{\theta^n}$

= $\frac{\alpha\theta_m^\alpha}{\theta^{\alpha+n+1}}$

= $\frac{(\alpha+n)(\theta_m)^{(\alpha+n)}}{\theta^{(\alpha+n)+1}} * \frac{\alpha}{(\alpha+n)\theta_m^n}$

$\propto \frac{(\alpha+n)(\theta_m)^{(\alpha+n)}}{\theta^{(\alpha+n)+1}}$ $\quad \theta \geq \theta_m$

0 $\quad \theta < \theta_m$

Therefore, derive the posterior distribution of θ is the Pareto distribution Pareto($\alpha + n, \theta_m$).

Q5:

Answer (a):

$$f_X(x) = \int_0^{+\infty} f_{XY}(x,y)\,dy = \int_0^{+\infty} ce^{-(xy+x+y)}\,dy = \frac{ce^{-x}}{x+1}$$

$$f_Y(y) = \int_0^{+\infty} f_{XY}(x,y)\,dx = \int_0^{+\infty} ce^{-(xy+x+y)}\,dx = \frac{ce^{-y}}{y+1}$$

$$f_{X|Y}(x|y) = \frac{f_{XY}(x,y)}{f_Y(y)} = \frac{ce^{-xy-x-y}}{\frac{ce^{-y}}{y+1}} = (y+1)e^{-x(y+1)}$$

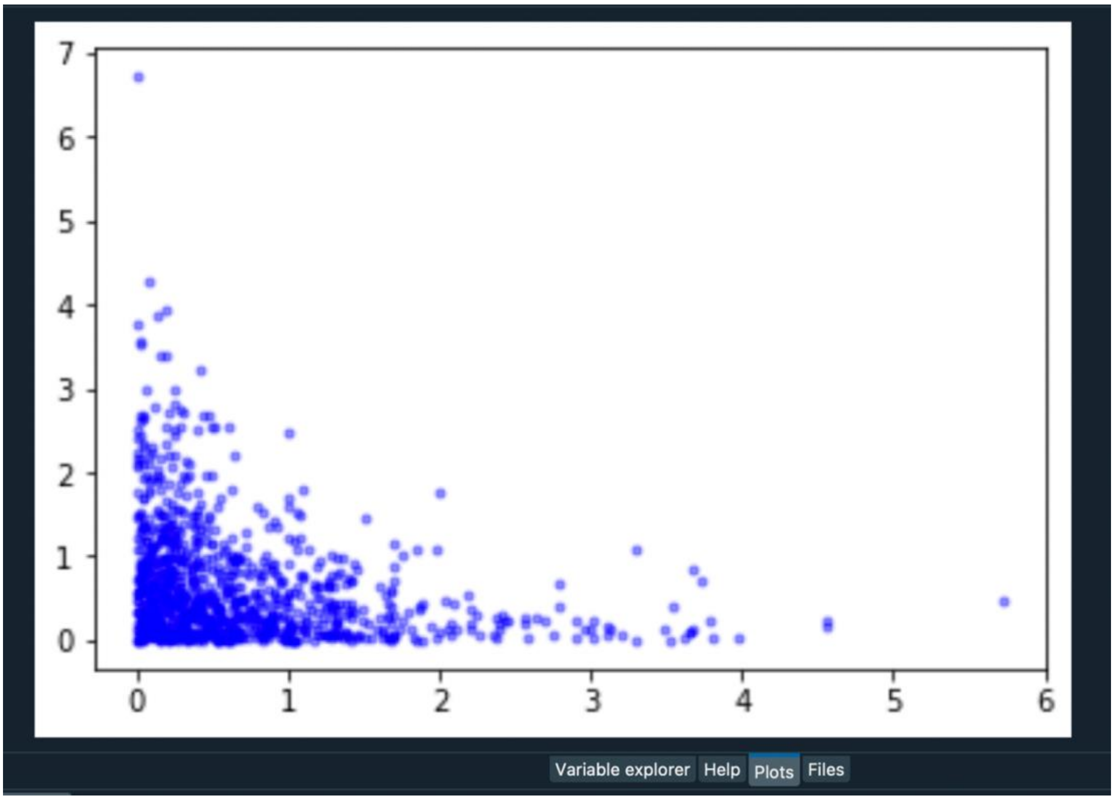
$$f_{Y|X}(y|x) = \frac{f_{XY}(x,y)}{f_X(x)} = \frac{ce^{-xy-x-y}}{\frac{ce^{-x}}{x+1}} = (x+1)e^{-y(x+1)}$$

Answer (b):

$$F_{X|Y}(x|y) = \int_0^x f_{X|Y}(x|y)\,dx \;=\; \int_0^x (y+1)e^{-x(y+1)}\,dx \;=\; 1 - e^{-x(y+1)}$$

$$F_{Y|X}(y|x) = \int_0^y f_{Y|X}(y|x)\,dy \;=\; \int_0^y (x+1)e^{-y(x+1)}\,dy \;=\; 1 - e^{-y(x+1)}$$

From the above formula, it can be seen that $F_{X|Y}(x|y)$ conforms to the exponential distribution of the parameter $\lambda = (y+1)$, the same as $F_{Y|X}(y|x)$ conforms to the exponential distribution of the parameter $\lambda = (x+1)$.



Code Appendic

Question 1

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn import model_selection
from sklearn.decomposition import PCA
from sklearn.linear_model import LinearRegression, Lasso
from sklearn.preprocessing import LabelEncoder
import warnings
warnings.filterwarnings("ignore")


df = pd.read_csv('Hitters.csv')
for cname in df.columns:
	if(df[cname].dtype == 'object'):
		df[cname][df[cname] == "?"] = np.nan
df = df.dropna()

lb_make = LabelEncoder()
df.League = lb_make.fit_transform(df.League)
df.Division = lb_make.fit_transform(df.Division)
df.NewLeague = lb_make.fit_transform(df.NewLeague)

Y = df.Salary
X = df.drop(['Salary'],axis=1)


#(a)

mse_min=[]
model = LinearRegression()
pca = PCA()
for i in range(1,20):
	pca.n_components = i
	mse = []
	X_pca = pca.fit_transform(X)
```

```
    mse = -model_selection.cross_val_score(model, X_pca, Y,scoring='neg_mean_squared_error',cv=10)
    mse_min.append(np.mean(mse))

print('The optimal number of components is '+ str(mse_min.index(min(mse_min))+1) + ". Linear regression cross validation error = "+ str(min(mse_min)))
print('-'*50)
plt.plot(range(1, len(mse_min) + 1), mse_min)
plt.xlabel('Number of principal components in regression')
plt.ylabel('MSE')
plt.title('Salary')
plt.axhline(np.min(mse_min), linestyle='--', color='.5')
plt.show()


#(b)

scores = list()
lasso = Lasso(fit_intercept=True)
alphas = np.arange(0,300,1)

for alpha in alphas:
    lasso.alpha = alpha
    this_scores = -model_selection.cross_val_score(lasso, X, Y,scoring='neg_mean_squared_error',cv=10)
    scores.append(np.mean(this_scores))

plt.plot(alphas, scores)
plt.ylabel('CV score (mean squared error)')
plt.xlabel('lambda')
plt.axhline(np.min(scores), linestyle='--', color='.5')
print('The best lambda is '+ str(alphas[scores.index(min(scores))]) + ". The corresponding mean squared error = "+ str(min(scores)))
```

Question 2

```
import pandas as pd
import statsmodels.api as sm

df = pd.read_csv("ships.csv")
```

```
Y = df.damage
X = df.drop(['damage'], axis = 1)

model = sm.GLM(Y, X, family = sm.families.Poisson())
results = model.fit()
print(results.summary())
```

Question 3

```
import numpy as np
from statsmodels.graphics.regressionplots import plot_leverage_resid2
import statsmodels.formula.api as smf
import statsmodels.api as sm
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings("ignore")

df = pd.read_csv('softdrink.csv')
Y = df.Time
X = df.drop(['Time'], axis = 1)

#(a)
results = smf.ols('Time ~ Cases + Distance', data=df).fit()
print(results.summary())
K = 2
dfd = len(Y) - K - 1
yhat = results.predict(X).values
se = np.sqrt(np.sum((np.array(Y) - yhat) ** 2) / (dfd))
print('The standard error of estimate: ',se)


#(b)
results_his = results
sns.residplot(df.Cases, results.resid, lowess=False, color="g")
sns.residplot(df.Distance, results.resid, lowess=False, color="g")
sns.distplot(results.resid)
plt.show()
```

```
fig = sm.qqplot(results.resid)
plt.show()
sns.distplot(results_his.resid)
plt.show()


#(c)
fig, ax = plt.subplots(figsize=(8,6))
fig = plot_leverage_resid2(results, ax = ax)
```

Question 5

```
import random
import matplotlib.pyplot as plt
N = 1000
x_res = []
y_res = []
x0 = 1
y0 = 1
y = y0
random.seed(N)
for i in range(N*2):
	x = random.expovariate(y + 1)
	y = random.expovariate(x + 1)
	x_res.append(x)
	y_res.append(y)
x_res = x_res[-N:]
y_res = y_res[-N:]
plt.clf()
plt.plot(x_res, y_res, 'b.', alpha = 0.4)
plt.show()
```