Homework1 20200639 Chae Woo Jin

Chapter 2: #2, 3, 5, 8, 9

Chapter 3: #1, 4, 7, 13, 14

Chapter 2.

2.

(A) It is a **regression** problem, as it wants to estimate CEO salary (quantitative response). Also, this problem is interested in **inference**, since it is interested in factors affecting CEO salary. In this case, the number of predictors is 3, and the sample size is 500.

$(\boldsymbol{n = 500,\ p = 3})$

(B) It is a **classification** problem, as it wants to determine response between two categories: success, failure. Also, this problem is interested in **prediction** since it wants to know whether new launch will be success or failure. In this case, the number of predictors is 2, and the sample size is 20.

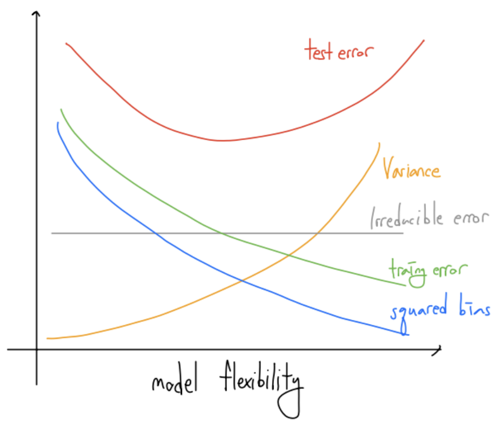
$(\boldsymbol{n = 20,\ p = 2})$

(C) It is a **regression** problem, because it wants to predict % change in the USD/Euro exchange rate in relation to the weekly changes in the world stock market, which is the quantitative factor. Also, this problem is interested in **prediction**. This is because it wants to predict future % change based on several factors. In this case, the number of predictors is 3, and the sample size is 52 (365/7≃52).

$(\boldsymbol{n = 52,\ p = 3})$

3.

(A)



(B) **Squared Bias:** Squared bias is an error that introduced by approximating a true relationship. However, it can be reduced as model becomes more flexible.

Variance: It refers to the amount by how estimator can be varied if it was estimated using different training dataset. As model becomes more flexible, it might capture to the noise of the data in the particular training dataset. Therefore, model has higher variance as flexibility increases.

Training error: Training error is the sum of square of distance between estimated output and true observation in the given dataset. As flexibility gets higher, it will approximate better to these data. Therefore, it reduces gradually as flexibility of the model increases.

Bayes error: The bayes error is kind of an irreducible error, which is constant, and it is the lowest achievable test error among all possible methods theoretically, which will be illustrated in the below.

Test error: Test error is given by the formula: variance + squared bias + bayes error.

$$E\left(y-\hat{f}(x_0)\right)^2 = Var\left(\hat{f}(x_0)\right) + \left[Bias\left(\hat{f}(x_0)\right)\right]^2 + Var(\varepsilon)$$

As model becomes more flexible, an increase of the variance roughly offsets the decrease of the squared bias. Generally, test error decreases as flexibility increases, and when the flexibility exceeds certain threshold, the test error starts to increase again.

5.

The advantages of a very flexible approach are that it yields less bias than less flexible one. Also, it provides better results when there are enough training data. However, the disadvantage of it is that it sacrifices more interpretability compared to less flexible approach. Furthermore, it has a risk of overfitting. Therefore, flexible approach is preferred when sample size is large and there are small number of predictors. However, it is preferred to use less flexible approach when sample size is small and the number of predictors is large.

8.

a)

```
[ ]  college = pd.read_csv('College.csv')
```

b)

```
[ ]  college2 = pd.read_csv('College.csv', index_col=0)
     college3 = college.rename({'Unnamed: 0': 'College'}, axis=1)
     college3 = college3.set_index('College')
     college = college3
```

c)

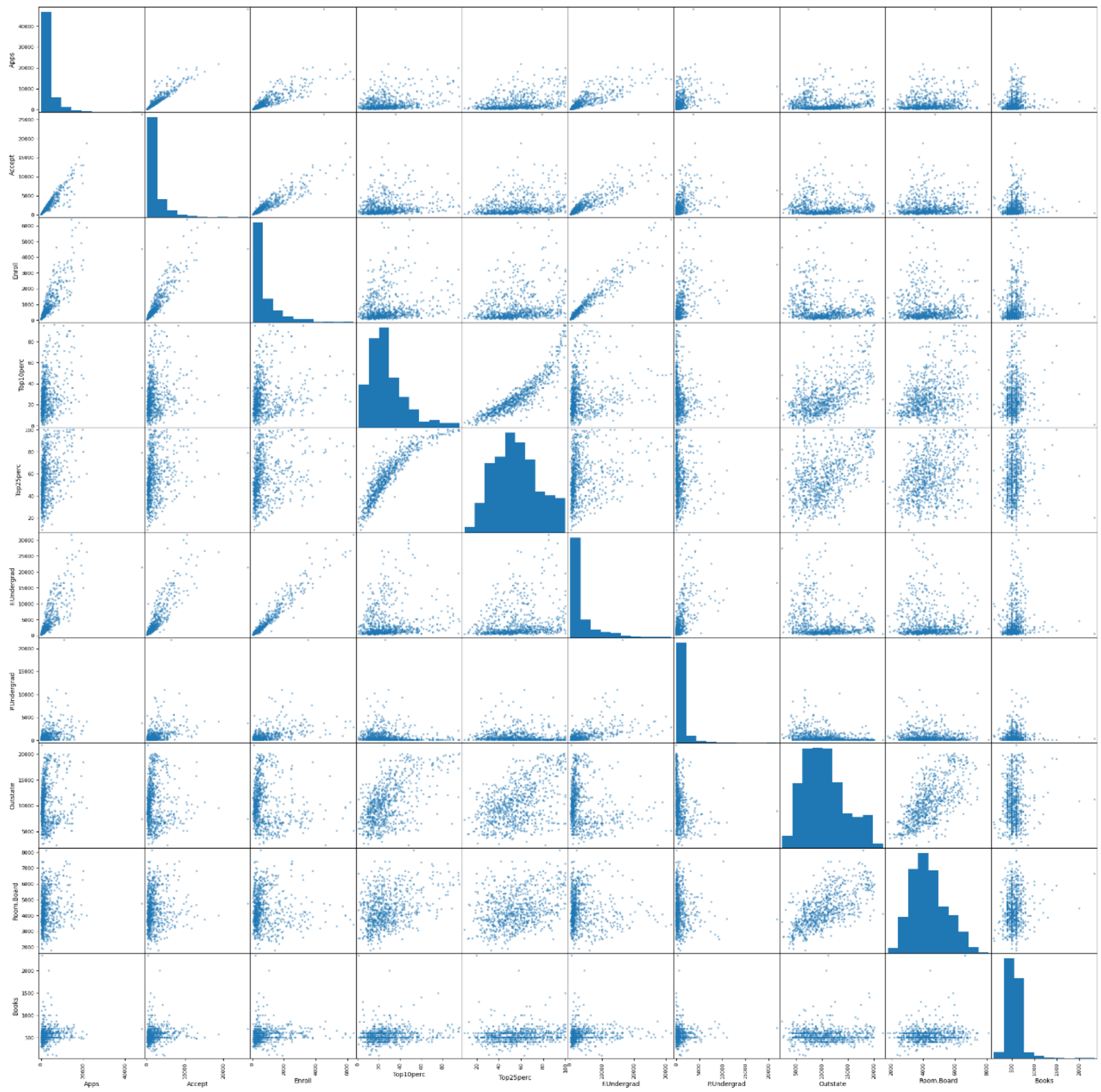
i.

```
college.describe()
```

	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	Outstate	Room.Board	Books	Personal	PhD	Terminal	S.F.
count	777.00	777.00	777.00	777.00	777.00	777.00	777.00	777.00	777.00	777.00	777.00	777.00	777.00	
mean	3,001.64	2,018.80	779.97	27.56	55.80	3,699.91	855.30	10,440.67	4,357.53	549.38	1,340.64	72.66	79.70	
std	3,870.20	2,451.11	929.18	17.64	19.80	4,850.42	1,522.43	4,023.02	1,096.70	165.11	677.07	16.33	14.72	
min	81.00	72.00	35.00	1.00	9.00	139.00	1.00	2,340.00	1,780.00	96.00	250.00	8.00	24.00	
25%	776.00	604.00	242.00	15.00	41.00	992.00	95.00	7,320.00	3,597.00	470.00	850.00	62.00	71.00	
50%	1,558.00	1,110.00	434.00	23.00	54.00	1,707.00	353.00	9,990.00	4,200.00	500.00	1,200.00	75.00	82.00	
75%	3,624.00	2,424.00	902.00	35.00	69.00	4,005.00	967.00	12,925.00	5,050.00	600.00	1,700.00	85.00	92.00	
max	48,094.00	26,330.00	6,392.00	96.00	100.00	31,643.00	21,836.00	21,700.00	8,124.00	2,340.00	6,800.00	103.00	100.00	

ii.

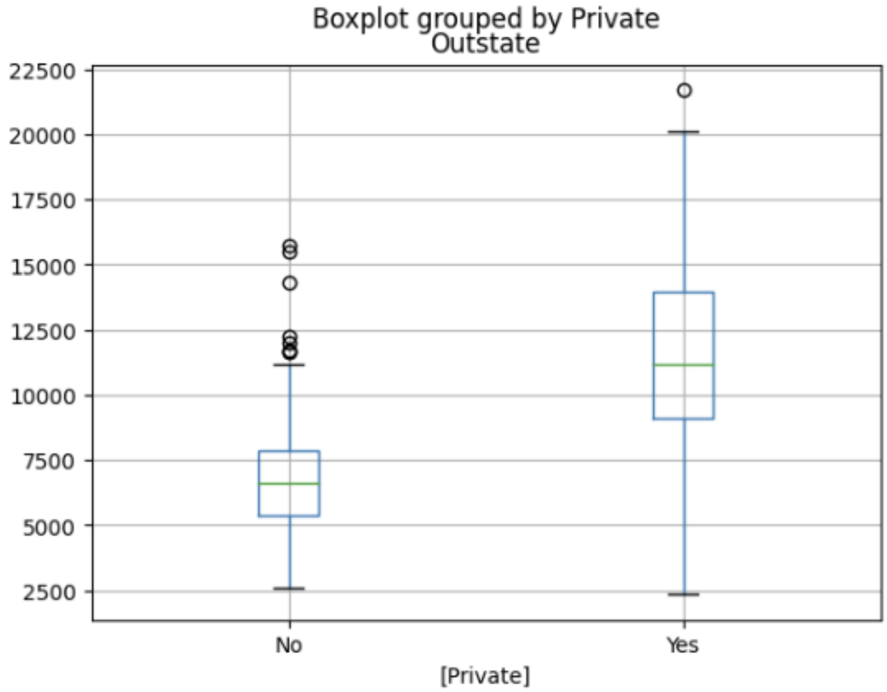
```
pd.plotting.scatter_matrix(college.iloc[:, 1:11], alpha=0.5, figsize = (30, 30))
```



iii.

```
[ ]  college.boxplot(column=['Outstate'], by=['Private'])
```

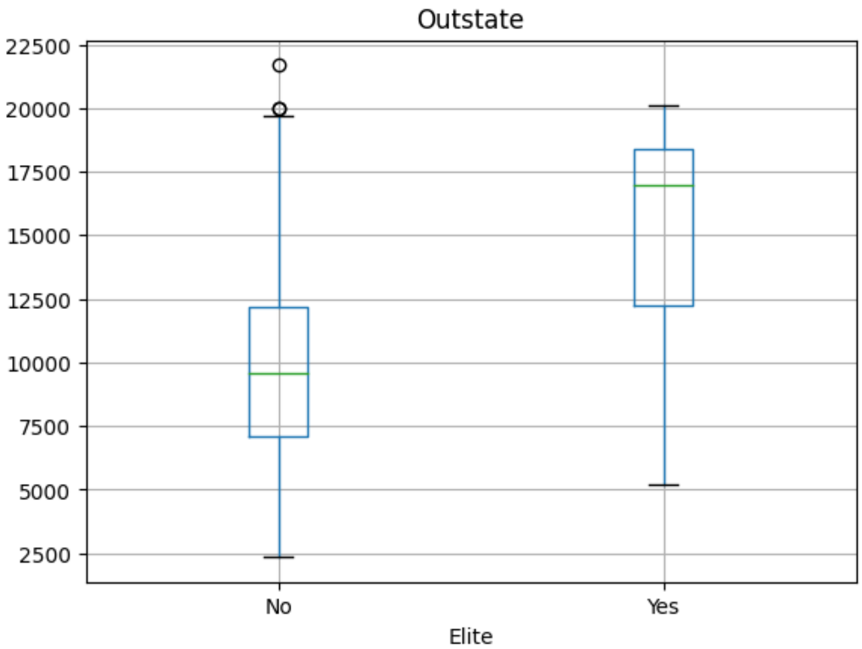
```
<Axes: title={'center': 'Outstate'}, xlabel='[Private]'>
```



iv.

```
college.value_counts(subset=['Elite'])
```

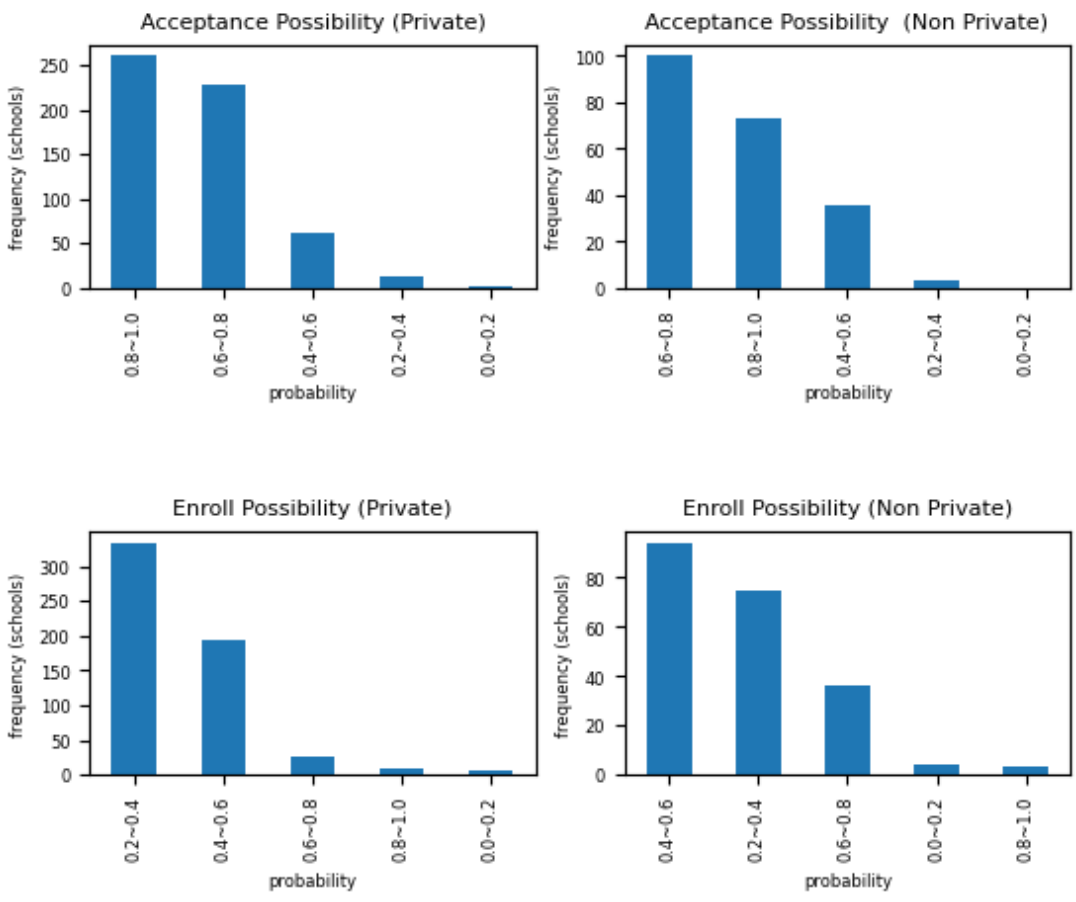
```
Elite
No     699
Yes     78
dtype: int64
```



v.

```
[ ] private = college[college['Private'] == 'Yes'].copy()
    nonprivate = college[college['Private'] == 'No'].copy()

    private['Acceptance Possibility (Private)'] = pd.cut(private['Accept'] / private['Apps'], bins=np.arange(0, 1.2, 0.2), labels=['0.0~0.2', '0.2~0.4', '0.4~0.6', '0.6~0.8', '0.8~1.0'])
    nonprivate['Acceptance Possibility (Non Private)'] = pd.cut(nonprivate['Accept'] / nonprivate['Apps'], bins=np.arange(0, 1.2, 0.2), labels=['0.0~0.2', '0.2~0.4', '0.4~0.6', '0.6~0.8', '0.8~1.0'])
    private['Enroll Possibility (Private)'] =  pd.cut(private['Enroll'] / private['Accept'], bins=np.arange(0, 1.2, 0.2), labels=['0.0~0.2', '0.2~0.4', '0.4~0.6', '0.6~0.8', '0.8~1.0'])
    nonprivate['Enroll Possibility (Non Private)'] = pd.cut(nonprivate['Enroll'] / nonprivate['Accept'], bins=np.arange(0, 1.2, 0.2), labels=['0.0~0.2', '0.2~0.4', '0.4~0.6', '0.6~0.8', '0.8~1.0'])
```

```
[ ] fig, axes = plt.subplots(nrows=2, ncols=2)

    private['Acceptance Possibility (Private)'].value_counts().plot(ax=axes[0, 0], kind='bar')
    nonprivate['Acceptance Possibility (Non Private)'].value_counts().plot(ax=axes[0, 1], kind='bar')
    private['Enroll Possibility (Private)'].value_counts().plot(ax=axes[1, 0], kind='bar')
    nonprivate['Enroll Possibility (Non Private)'].value_counts().plot(ax=axes[1, 1], kind='bar')

    axes[0, 0].set_title('Acceptance Possibility (Private)', fontsize=8)
    axes[0, 1].set_title('Acceptance Possibility  (Non Private)', fontsize=8)
    axes[1, 0].set_title('Enroll Possibility (Private)', fontsize=8)
    axes[1, 1].set_title('Enroll Possibility (Non Private)', fontsize=8)

    for i in range(2):
        for j in range(2):
            axes[i, j].xaxis.set_tick_params(labelsize=6)
            axes[i, j].yaxis.set_tick_params(labelsize=6)
            axes[i, j].set_xlabel('probability', fontsize=6)
            axes[i, j].set_ylabel('frequency (schools)', fontsize=6)

    plt.subplots_adjust(hspace=1)
```



vi.

According to v., acceptance possibility (acceptance rate) seems higher for private school than non-private school. However, enroll possibility (enroll rate) was much higher for non-private school than private one.

9.

(a)

(a) Qualitative and qualitative predictors

```
[31] auto.head()
```

	mpg	cylinders	displacement	horsepower	weight	acceleration	year	origin	name
0	18.00	8	307.00	130	3504	12.00	70	1	chevrolet chevelle malibu
1	15.00	8	350.00	165	3693	11.50	70	1	buick skylark 320
2	18.00	8	318.00	150	3436	11.00	70	1	plymouth satellite
3	16.00	8	304.00	150	3433	12.00	70	1	amc rebel sst
4	17.00	8	302.00	140	3449	10.50	70	1	ford torino

Quantitative Predictors

```
[32] quantitative = auto.select_dtypes(include=['number']).columns
     quantitative

Index(['mpg', 'cylinders', 'displacement', 'horsepower', 'weight',
       'acceleration', 'year', 'origin'],
      dtype='object')
```

Qualitative Predictors

```
[33] qualitative = auto.select_dtypes(exclude=['number']).columns
     qualitative

Index(['name'], dtype='object')
```

'mpg', 'cylinders', 'displacement', 'horsepower', 'weight', 'acceleration', 'year', and 'origin' are quantitative predictors, while 'name' is a qualitative predictor.

(b)

(b) Range of each quantitave predictor

```
[34] auto[quantitative].apply(max) - auto[quantitative].apply(min)

mpg               37.60
cylinders          5.00
displacement     387.00
horsepower       184.00
weight         3,527.00
acceleration      16.80
year              12.00
origin             2.00
dtype: float64
```

(c)

(c) Mean and Standard Deviation

[35]
```
auto[quantitative].mean()
```

```
mpg               23.45
cylinders          5.47
displacement     194.41
horsepower       104.47
weight         2,977.58
acceleration      15.54
year              75.98
origin             1.58
dtype: float64
```

[36]
```
auto[quantitative].std()
```

```
mpg              7.81
cylinders        1.71
displacement   104.64
horsepower      38.49
weight         849.40
acceleration     2.76
year             3.68
origin           0.81
dtype: float64
```

(d)

(d) Range, Mean, Standard Deviation of data sample

[37]
```
auto_sample = auto.drop(axis=0, index=range(10, 86))
```

[38]
```
auto_sample[quantitative].apply(max) - auto_sample[quantitative].apply(min)
```

```
mpg               35.60
cylinders          5.00
displacement     387.00
horsepower       184.00
weight         3,348.00
acceleration      16.30
year              12.00
origin             2.00
dtype: float64
```

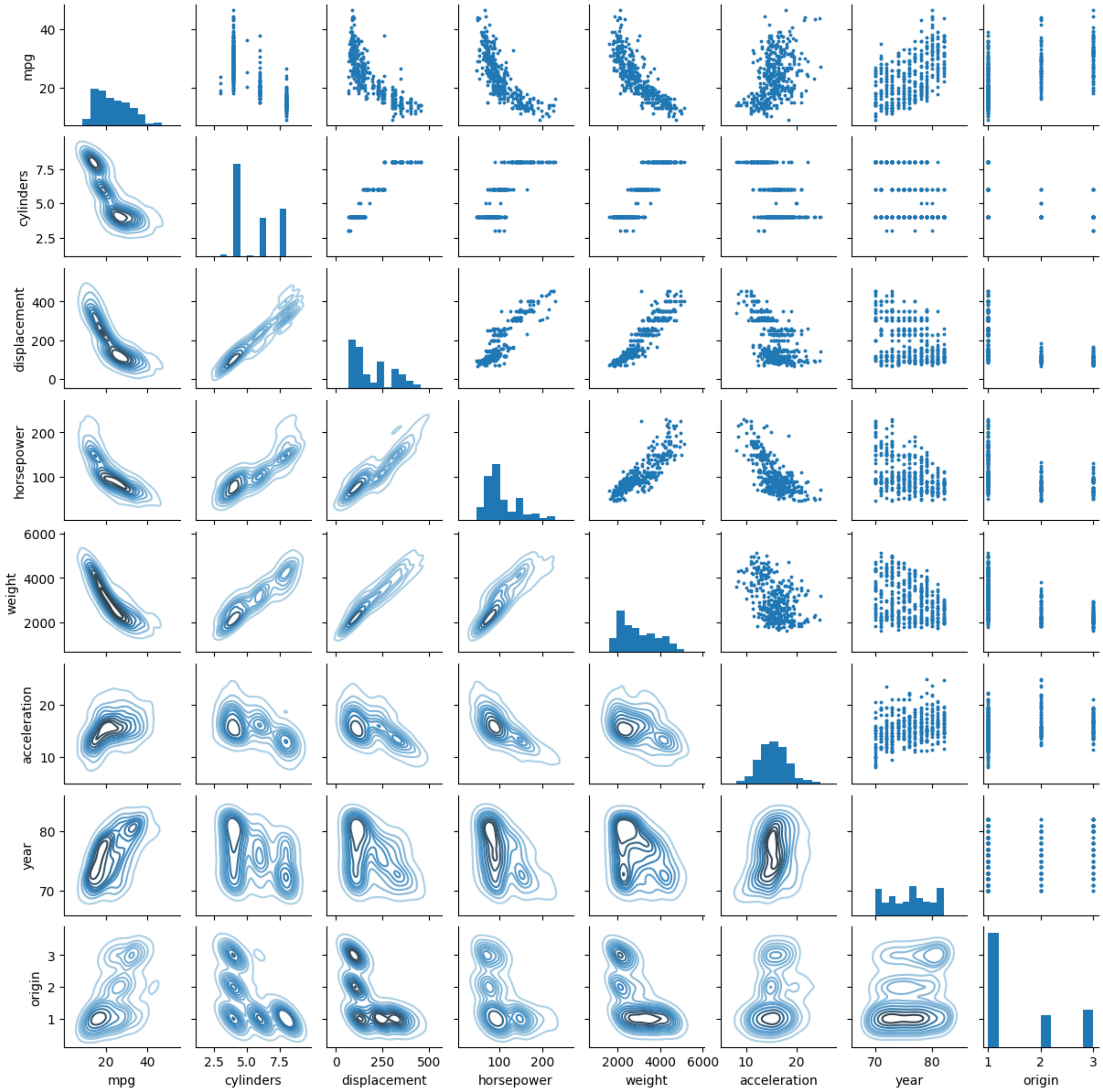
[39]
```
auto_sample[quantitative].mean()
```

```
mpg               24.41
cylinders          5.37
displacement     187.51
horsepower       100.85
weight         2,936.53
acceleration      15.72
year              77.14
origin             1.60
dtype: float64
```

[40]
```
auto_sample[quantitative].std()
```

```
mpg              7.86
cylinders        1.65
displacement   100.11
horsepower      35.95
weight         811.87
acceleration     2.71
year             3.12
origin           0.82
dtype: float64
```

(e)



According to the scatterplot, displacement and weight seems to have strong linear relationship.

Also, mpg seems to have non-linear relationship with weight, horsepower, and displacement.

(f)

Based on the previous answer, weight, horsepower, and displacement can be used to predict mpg.

Chapter 3.

1. The null hypothesis for 'TV' is that in the presence of radio ads and newspaper ads, TV ads have no effect on sales. Similarly, the null hypothesis for "radio" is that in the presence of TV and newspaper ads, radio ads have no effect on sales. (And there is a similar null hypothesis for 'newspaper'.) From Table 3.4, one can conclude that there is no evidence to reject null hypothesis for 'newspaper', because p-value (0.86) is very high, even much higher than the typical confidence level (0.05, …). This suggests that there is no relationship between newspaper ads and sales, in the presence of TV and Radio.

4.

(A) Cubic regression will have lower training RSS, since it is more flexible model than a linear model. Therefore, it will fit better to the training set, and lower RSS will clearly support this.

(B) However, for RSS for test data, the linear model will have a lower RSS than the cubic model. This is because X and Y are proposed to have a linear relationship. If the two variables have a linear relationship, less flexible is more suitable for the test set because the more flexible model increases the variance compared to the effect of reducing the bias.

(C) Whether the relationship between X and Y is linear or not, training RSS of the cubic regression is less than linear regression. This is because more flexible model (cubic regression in this case) can fit better to the given training data then the other one (a linear regression model).

(D) There is not enough information to tell. Generally, more flexible model tends to fit better in the training data. However, when its flexibility exceeds certain threshold, overfitting occurs. Then, test RSS increases as its flexibility increases further. Therefore, without information about how much non-linear relationship two variables has, one cannot strongly conclude that certain model has lower RSS than the other.

7.

WTS: $R^2 = r^2 = \text{cor}^2(X,Y)$ for simple linear regression of Y onto X.

WLOG, let $\bar{x} = \bar{y} = 0$.

$$\text{cor}(X,Y) = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i (x_i - \bar{x})^2}\sqrt{\sum_i (y_i - \bar{y})^2}} = \frac{\sum_i x_i y_i}{\sqrt{\sum_i x_i^2}\sqrt{\sum_i y_i^2}}$$

$$\begin{cases} \hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x} = 0 \\ \hat{\beta}_1 = \dfrac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sum_i (x_i - \bar{x})^2} = \dfrac{\sum_i x_i y_i}{\sum x_i^2}, \end{cases}$$

$$R^2 = 1 - \frac{RSS}{TSS} = 1 - \frac{\sum (y_i - \hat{y})^2}{\sum (y_i - \bar{y})^2} = 1 - \frac{\sum_i (y_i - \hat{\beta}_1 x_i)^2}{\sum y_i^2}$$

$$= 1 - \frac{\sum_i (y_i^2 - 2\hat{\beta}_1 x_i y_i + \hat{\beta}_1^2 x_i^2)}{\sum y_i^2} = \frac{2\hat{\beta}_1 \sum x_i y_i - \hat{\beta}_1^2 \sum_i x_i^2}{\sum y_i^2}$$

$$= \frac{2\left(\sum_i x_i y_i\right)\left(\sum_i x_i y_i\right) / \left(\sum_i x_i^2\right) - \left(\sum_i x_i y_i\right)^2 \left(\sum x_i^2\right) / \left(\sum x_i^2\right)^2}{\sum y_i^2}$$

$$= \frac{\left(\sum_i x_i y_i\right)^2}{\sum x_i^2 \sum y_i^2} = \text{cor}^2(X,Y)$$

$\boxed{}$

13.

(a)

```
x = np.random.normal(loc=0, scale=1, size=100)
```

(b)

```
[43] eps = np.random.normal(loc=0, scale=0.25, size=100)
```

(c)

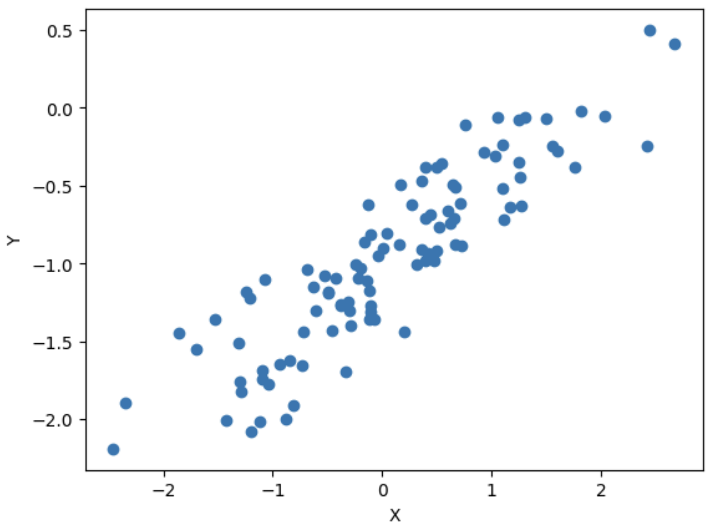
```
[44] y = -1 + 0.5*x + eps
     len(y)
```

```
100
```

$\beta_0 = -1, \beta_1 = 0.5$

(d)

```
plt.scatter(x=x, y=y)
plt.xlabel('X')
plt.ylabel('Y')
plt.show()
```



It seems that X, Y linear relation, with a variance as is to be expected.

(e)

```
[49] # Calculate Using ols package

     df = pd.DataFrame({'x': x, 'y': y})
     reg = smf.ols('y ~ x', data=df).fit()
     reg.summary()
```

OLS Regression Results

Dep. Variable:	y	**R-squared:**	0.797
Model:	OLS	**Adj. R-squared:**	0.795
Method:	Least Squares	**F-statistic:**	384.2
Date:	Sat, 16 Sep 2023	**Prob (F-statistic):**	1.11e-35
Time:	06:13:08	**Log-Likelihood:**	-7.0113
No. Observations:	100	**AIC:**	18.02
Df Residuals:	98	**BIC:**	23.23
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	-0.9993	0.026	-38.032	0.000	-1.051	-0.947
x	0.5035	0.026	19.601	0.000	0.453	0.555

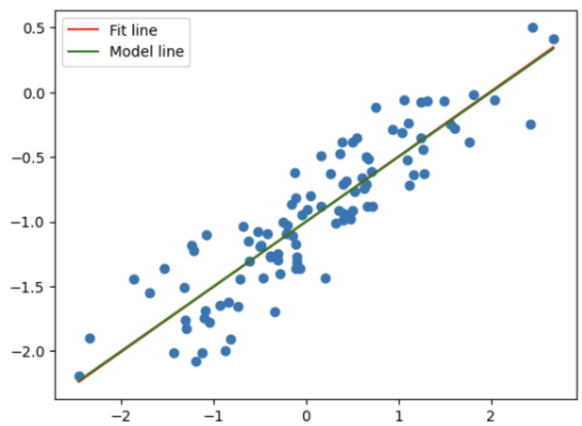
Omnibus:	3.285	**Durbin-Watson:**	2.024
Prob(Omnibus):	0.194	**Jarque-Bera (JB):**	1.926
Skew:	-0.016	**Prob(JB):**	0.382
Kurtosis:	2.321	**Cond. No.**	1.07

From the regression, we obtained $\widehat{\beta_0} = -0.9993$, $\widehat{\beta_1} = 0.5035$, both of which are quite close to real β_0, and β_1 respectively. As p-value of each predictor is pretty low, so that we can reject the null hypothesis.

(f)

```
model = LinearRegression()
model.fit(x.reshape(-1, 1), y)

# Scatter plot
plt.scatter(df.x, df.y)

# Fitted line with LSE
xfit = np.linspace(df.x.min(), df.x.max(), 100)
yfit = model.predict(xfit.reshape(-1, 1))
fit, = plt.plot(xfit, yfit, color='r')

# Population line
xpop = np.linspace(df.x.min(), df.x.max(), 100)
ypop = -1 + 0.5 * xpop
pop, = plt.plot(xpop, ypop, color='g')

# create legend
plt.legend([fit, pop],['Fit line','Model line'])
```

```
<matplotlib.legend.Legend at 0x7953730e4a30>
```



(g)

```
[53] reg = smf.ols('y ~ x + I(x**2)', data=df).fit()
     reg.summary()
```

OLS Regression Results

Dep. Variable:	y	**R-squared:**	0.798
Model:	OLS	**Adj. R-squared:**	0.794
Method:	Least Squares	**F-statistic:**	191.6
Date:	Sat, 16 Sep 2023	**Prob (F-statistic):**	2.02e-34
Time:	06:13:09	**Log-Likelihood:**	-6.6976
No. Observations:	100	**AIC:**	19.40
Df Residuals:	97	**BIC:**	27.21
Df Model:	2		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
Intercept	-1.0139	0.032	-31.351	0.000	-1.078	-0.950
x	0.5000	0.026	19.138	0.000	0.448	0.552
I(x ** 2)	0.0143	0.018	0.781	0.437	-0.022	0.050

Omnibus:	2.755	**Durbin-Watson:**	2.071
Prob(Omnibus):	0.252	**Jarque-Bera (JB):**	1.749
Skew:	-0.038	**Prob(JB):**	0.417
Kurtosis:	2.357	**Cond. No.**	2.52

Let H_0: coefficient of 2nd order term=0, H_1: coefficient of 2nd order term≠0.

However, as p-value of X^2 term is 0.437, there is no evidence to reject H_0. Therefore, it is reasonable to conclude that X^2 and Y are statistically unrelated. In other words, it means that the quadratic term did not improve the model fit.

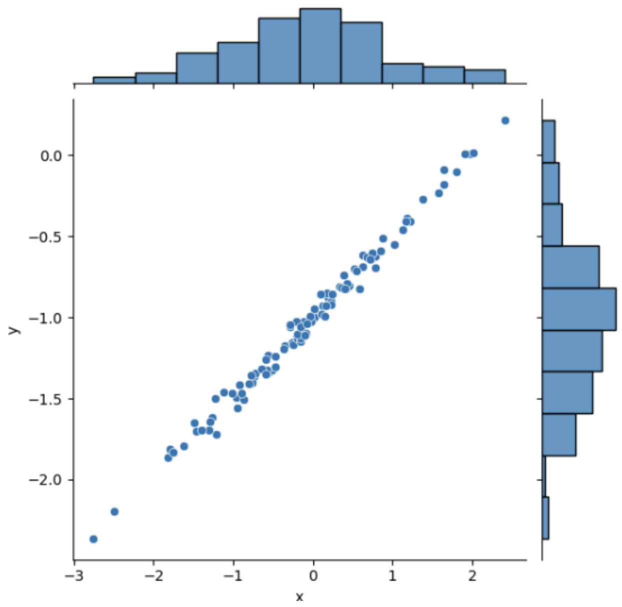
(h) $X \sim N(0,1)$, $\varepsilon \sim N(0, 0.05^2)$

```
[54] x = np.random.normal(loc = 0, size = 100)
     eps = np.random.normal(loc = 0, scale = 0.05, size = 100)
     y = -1 + 0.5*x + eps
     len(y)
```

100

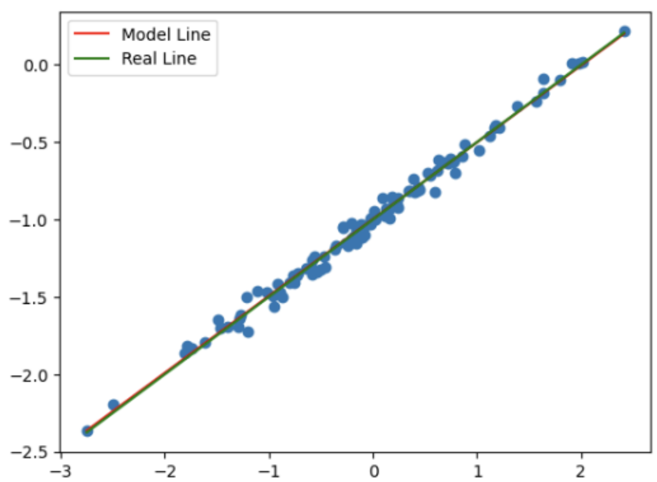
```
df = pd.DataFrame({'x': x, 'y': y})
sns.jointplot(x='x', y='y', data=df)
```

<seaborn.axisgrid.JointGrid at 0x795372ff2f80>




```
[56] model = LinearRegression()
     model.fit(x.reshape(-1, 1), y)

     plt.subplots()
     plt.scatter(df.x, df.y)

     xfit = np.linspace(x.min(), x.max(), 100)
     yfit = model.predict(xfit.reshape(-1, 1))
     fit, = plt.plot(xfit, yfit, color='r')

     xpop = np.linspace(x.min(), x.max(), 100)
     ypop = -1 + 0.5*xpop
     pop, = plt.plot(xpop, ypop, color='g')

     plt.legend([fit, pop], ['Model Line', 'Real Line'])
```

<matplotlib.legend.Legend at 0x795372ed3610>



```
[57] reg = smf.ols('y ~ x', data=df).fit()
     reg.summary()
```

OLS Regression Results

Dep. Variable:	y	**R-squared:**	0.990
Model:	OLS	**Adj. R-squared:**	0.990
Method:	Least Squares	**F-statistic:**	1.020e+04
Date:	Sat, 16 Sep 2023	**Prob (F-statistic):**	7.10e-101
Time:	06:13:10	**Log-Likelihood:**	161.57
No. Observations:	100	**AIC:**	-319.1
Df Residuals:	98	**BIC:**	-313.9
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	-0.9978	0.005	-204.990	0.000	-1.007	-0.988
x	0.4965	0.005	100.998	0.000	0.487	0.506

Omnibus:	0.187	**Durbin-Watson:**	1.937
Prob(Omnibus):	0.911	**Jarque-Bera (JB):**	0.176
Skew:	-0.095	**Prob(JB):**	0.916
Kurtosis:	2.919	**Cond. No.**	1.07

As we reduce the noise, we could obtain a better fit, which can be inferred from narrowed confidence intervals, and a higher R-squared.

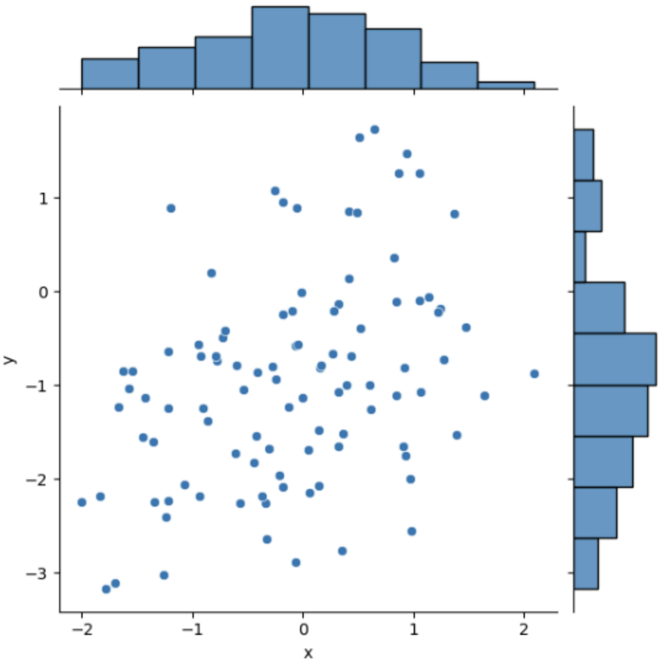
(i) $X \sim N(0, 1^2), \varepsilon \sim N(0, 1^2)$

```
x = np.random.normal(loc = 0, size = 100)
eps = np.random.normal(loc = 0, scale = 1, size = 100)
y = -1 + 0.5*x + eps
len(y)
```

100

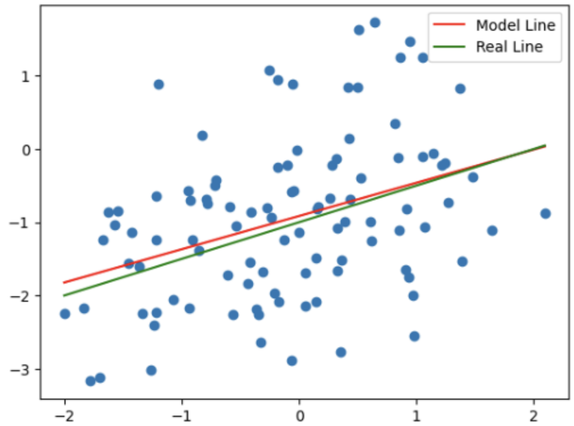
```
[59] df = pd.DataFrame({'x': x, 'y': y})
     sns.jointplot(x='x', y='y', data=df)
```

<seaborn.axisgrid.JointGrid at 0x795372fda2f0>



```
model = LinearRegression()
model.fit(x.reshape(-1, 1), y)

plt.subplots()
plt.scatter(df.x, df.y)

xfit = np.linspace(x.min(), x.max(), 100)
yfit = model.predict(xfit.reshape(-1, 1))
fit, = plt.plot(xfit, yfit, color='r')

xpop = np.linspace(x.min(), x.max(), 100)
ypop = -1 + 0.5*xpop
pop, = plt.plot(xpop, ypop, color='g')

plt.legend([fit, pop], ['Model Line', 'Real Line'])
```

```
<matplotlib.legend.Legend at 0x795372cf5c60>
```




```
[60] reg = smf.ols('y ~ x', data=df).fit()
     reg.summary()
```

OLS Regression Results

Dep. Variable:	y	**R-squared:**	0.142
Model:	OLS	**Adj. R-squared:**	0.134
Method:	Least Squares	**F-statistic:**	16.28
Date:	Sat, 16 Sep 2023	**Prob (F-statistic):**	0.000108
Time:	06:13:11	**Log-Likelihood:**	-144.21
No. Observations:	100	**AIC:**	292.4
Df Residuals:	98	**BIC:**	297.6
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	-0.9148	0.104	-8.798	0.000	-1.121	-0.708
x	0.4539	0.112	4.035	0.000	0.231	0.677

Omnibus:	2.107	**Durbin-Watson:**	2.053
Prob(Omnibus):	0.349	**Jarque-Bera (JB):**	2.131
Skew:	0.314	**Prob(JB):**	0.345
Kurtosis:	2.659	**Cond. No.**	1.14

On the contrary to the previous one, we obtained a worse fit. This is because the R-squared is 0.142 and the confidence intervals for the coefficients are much wider.

(j)

Original data set: $[-1.078, -0.950]$ for β_0, $[0.448, 0.552]$ for β_1

Noiser data set: $[-1.121, -7.08]$ for β_0, $[0.231, 0.677]$ for β_1

Less noisy data set: $[-1.007, -9.988]$ for β_0, $[0.487, 0.506]$ for β_1

14.

(a)

```
[ ]  rng = np.random.default_rng(10)
     x1 = rng.uniform(0, 1, size=100)
     x2 = 0.5 * x1 + rng.normal(size=100) / 10
     y = 2 + 2 * x1 + 0.3 * x2 + rng.normal(size=100)
```

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \epsilon = 2 + 2X_1 + 0.3X_2 + \epsilon$$

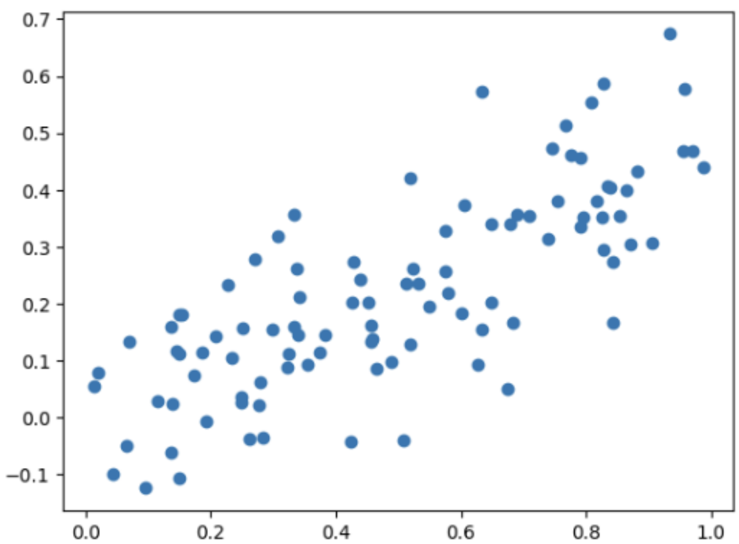
$$\beta_0 = 2,\ \beta_1 = 2,\ \beta_2 = 0.3$$

(b)

```
[ ]  np.corrcoef(x1, x2)

     array([[1.        , 0.7723245],
            [0.7723245, 1.        ]])
```

$$Cor(X_1, X_2) = 0.772345$$

```
[ ]  plt.scatter(x1, x2)

     <matplotlib.collections.PathCollection at 0x795372bc03a0>
```



(c)

```
[ ]  df = pd.DataFrame({'x1': x1, 'x2': x2, 'y': y})

     reg = smf.ols('y ~ x1 + x2', data=df).fit()
     reg.summary()
```

OLS Regression Results

Dep. Variable:	y	**R-squared:**	0.291
Model:	OLS	**Adj. R-squared:**	0.276
Method:	Least Squares	**F-statistic:**	19.89
Date:	Sat, 16 Sep 2023	**Prob (F-statistic):**	5.76e-08
Time:	06:13:12	**Log-Likelihood:**	-130.62
No. Observations:	100	**AIC:**	267.2
Df Residuals:	97	**BIC:**	275.1
Df Model:	2		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	1.9579	0.190	10.319	0.000	1.581	2.334
x1	1.6154	0.527	3.065	0.003	0.569	2.661
x2	0.9428	0.831	1.134	0.259	-0.707	2.592

Omnibus:	0.051	**Durbin-Watson:**	1.964
Prob(Omnibus):	0.975	**Jarque-Bera (JB):**	0.041
Skew:	-0.036	**Prob(JB):**	0.979
Kurtosis:	2.931	**Cond. No.**	11.9

$\widehat{\beta_0} = 1.9579$, $\widehat{\beta_1} = 1.6154$, $\widehat{\beta_2} = 0.9428$ are the estimated values of the true coefficients, which are the followings. $\beta_0 = 2$, $\beta_1 = 2$, $\beta_2 = 0.3$. From the summary, we can conclude that we can reject the null hypothesis for β_1 because its p-value is below 5%. However, we cannot reject the null hypothesis for β_2 because its p-value is above the 5%.

(d)

```
reg = smf.ols('y ~ x1', data=df[['x1', 'y']]).fit()
reg.summary()
```

OLS Regression Results

Dep. Variable:	y	**R-squared:**	0.281
Model:	OLS	**Adj. R-squared:**	0.274
Method:	Least Squares	**F-statistic:**	38.39
Date:	Sat, 16 Sep 2023	**Prob (F-statistic):**	1.37e-08
Time:	06:13:12	**Log-Likelihood:**	-131.28
No. Observations:	100	**AIC:**	266.6
Df Residuals:	98	**BIC:**	271.8
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	1.9371	0.189	10.242	0.000	1.562	2.312
x1	2.0771	0.335	6.196	0.000	1.412	2.742

Omnibus:	0.204	**Durbin-Watson:**	1.931
Prob(Omnibus):	0.903	**Jarque-Bera (JB):**	0.042
Skew:	-0.046	**Prob(JB):**	0.979
Kurtosis:	3.038	**Cond. No.**	4.65

,

The null hypothesis can be rejected and the alternative hypothesis accepted because p-value is zero.

(e)

```
reg = smf.ols('y ~ x2', data=df[['x2', 'y']]).fit()
reg.summary()
```

OLS Regression Results

Dep. Variable:	y	**R-squared:**	0.222
Model:	OLS	**Adj. R-squared:**	0.214
Method:	Least Squares	**F-statistic:**	27.99
Date:	Sat, 16 Sep 2023	**Prob (F-statistic):**	7.43e-07
Time:	06:13:12	**Log-Likelihood:**	-135.24
No. Observations:	100	**AIC:**	274.5
Df Residuals:	98	**BIC:**	279.7
Df Model:	1		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
Intercept	2.3239	0.154	15.124	0.000	2.019	2.629
x2	2.9103	0.550	5.291	0.000	1.819	4.002

Omnibus:	0.191	**Durbin-Watson:**	1.943
Prob(Omnibus):	0.909	**Jarque-Bera (JB):**	0.373
Skew:	-0.034	**Prob(JB):**	0.830
Kurtosis:	2.709	**Cond. No.**	6.11

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

The null hypothesis can be rejected and the alternative hypothesis accepted because p-value is 0.

(f)

No, since X_1, and X_2 are highly correlated. Therefore, when two variables are both used in the prediction model, one of them is likely to lose its ability to explain because the other one can explain expected output enough. This can be clearly demonstrated by the fact that each variable has a clear linear relationship with Y, when used respectively.

(g)

```
x1 = np.concatenate([x1, [0.1]])
x2 = np.concatenate([x2, [0.8]])
y = np.concatenate([y, [6]])
```

```
[ ] # model (c)

df = pd.DataFrame({'x1': x1, 'x2': x2, 'y': y})
reg = smf.ols('y ~ x1 + x2', data=df).fit()
reg.summary()
```

OLS Regression Results

Dep. Variable:	y	**R-squared:**	0.292
Model:	OLS	**Adj. R-squared:**	0.277
Method:	Least Squares	**F-statistic:**	20.17
Date:	Sat, 16 Sep 2023	**Prob (F-statistic):**	4.60e-08
Time:	06:13:12	**Log-Likelihood:**	-135.30
No. Observations:	101	**AIC:**	276.6
Df Residuals:	98	**BIC:**	284.5
Df Model:	2		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	2.0618	0.192	10.720	0.000	1.680	2.443
x1	0.8575	0.466	1.838	0.069	-0.068	1.783
x2	2.2663	0.705	3.216	0.002	0.868	3.665

Omnibus:	0.139	**Durbin-Watson:**	1.894
Prob(Omnibus):	0.933	**Jarque-Bera (JB):**	0.320
Skew:	0.013	**Prob(JB):**	0.852
Kurtosis:	2.725	**Cond. No.**	9.68

```
[ ] # model (d)

reg = smf.ols('y ~ x1', data=df[['x1', 'y']]).fit()
reg.summary()
```

OLS Regression Results

Dep. Variable:	y	**R-squared:**	0.217
Model:	OLS	**Adj. R-squared:**	0.209
Method:	Least Squares	**F-statistic:**	27.42
Date:	Sat, 16 Sep 2023	**Prob (F-statistic):**	9.23e-07
Time:	06:13:12	**Log-Likelihood:**	-140.37
No. Observations:	101	**AIC:**	284.7
Df Residuals:	99	**BIC:**	290.0
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	2.0739	0.201	10.310	0.000	1.675	2.473
x1	1.8760	0.358	5.236	0.000	1.165	2.587

Omnibus:	8.232	**Durbin-Watson:**	1.636
Prob(Omnibus):	0.016	**Jarque-Bera (JB):**	10.781
Skew:	0.396	**Prob(JB):**	0.00456
Kurtosis:	4.391	**Cond. No.**	4.61

```
[ ] # model (e)

reg = smf.ols('y ~ x2', data=df[['x2', 'y']]).fit()
reg.summary()
```

OLS Regression Results

Dep. Variable:	y	**R-squared:**	0.267
Model:	OLS	**Adj. R-squared:**	0.260
Method:	Least Squares	**F-statistic:**	36.10
Date:	Sat, 16 Sep 2023	**Prob (F-statistic):**	3.13e-08
Time:	06:13:12	**Log-Likelihood:**	-137.01
No. Observations:	101	**AIC:**	278.0
Df Residuals:	99	**BIC:**	283.3
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	2.2840	0.151	15.088	0.000	1.984	2.584
x2	3.1458	0.524	6.008	0.000	2.107	4.185

Omnibus:	0.495	**Durbin-Watson:**	1.939
Prob(Omnibus):	0.781	**Jarque-Bera (JB):**	0.631
Skew:	-0.041	**Prob(JB):**	0.729
Kurtosis:	2.621	**Cond. No.**	5.84

```
[ ] np.corrcoef(x1, x2)

array([[1.        , 0.67891508],
       [0.67891508, 1.        ]])
```

By adding outlier observation to each variable X_1 and X_2, we can observe that the coefficient of X_1 decreased, and the one of X_2 increased. Also, now the null hypothesis for X_2 is rejected and accepted for X_1, which is the opposite compared to the result in (c). This is because the newly added data improved the credibility of X_2 to predict Y against X_1. Nevertheless, prediction using each variable tells that each one still has its relationship with Y.