Statistics Homework 01

B08705034 資管一 施芊羽

Chapter 1

1.3

a.

The population of interest is $25,000$ which are the number of the registered voters in the city.

b.

The sample is 200 which is the number of registered voters who are interviewed.

c.

48% is a statistic since it's a description of the sample instead of the population.

1.7

a.

I'll make the conclusion that the percentage of observed a head is 95% is far from fair($50-50$ in this case). However, I'll state that my conclusion and opinion will be a bit wrong since the sample is not large enough.

b.

I'll conclude that the owner is quite true since after my flips I got a 55% chance to observe head which may be closed to 50%, and it might be closer when the sample is larger.

c.

No, I don't believe that if I flip a perfectly fair coin 100 times, I will always observe exactly 50 heads. I do think that the number between $40 - 60$ may be likely got, however, the numbers from 0 to 100 are all possible to happen.

Chapter 2

2.5

a.

Interval.

b.

Interval.

c.

Nominal.

d.

Interval.

e.

Nominal.

2.11

a.

Nominal.

b.

Interval.

c.

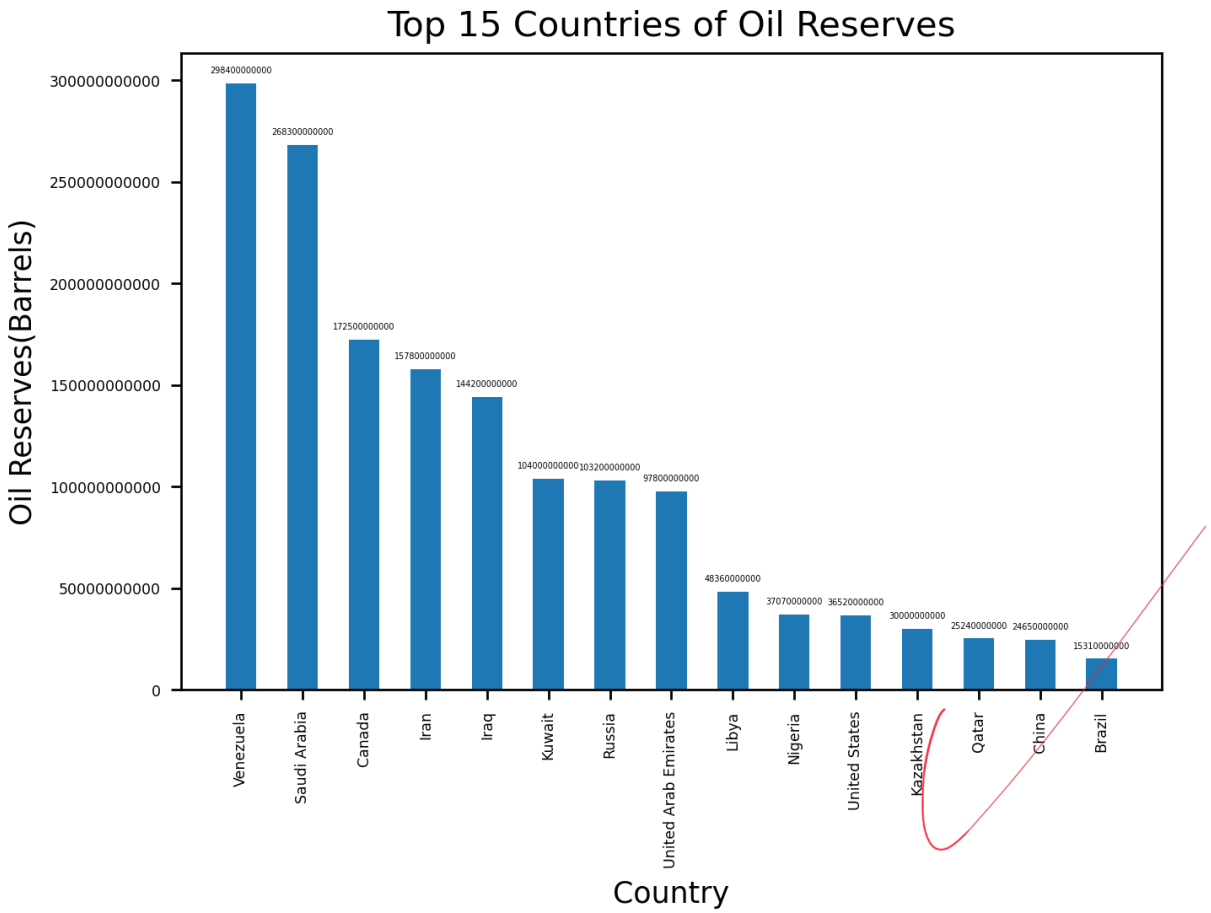
Ordinal.

2.13

In [2]:

```
from matplotlib import pyplot as plt
%matplotlib inline
# 設定圖形大小; DPI越大圖越大
plt.rcParams["figure.dpi"] = 250
import seaborn as sns
import pandas as pd
import numpy as np
import scipy.stats as stats
import statsmodels.api as sm
import statsmodels.stats.api as sms
import statsmodels.formula.api as smf

df_c02_13 = pd.read_excel('Xr02-13.xlsx') # Read the given data
df_c02_13 = df_c02_13.sort_values(by=['Oil Reserves (Barrels)'], as
cending = False) # Sorting the data with a descending order
display(df_c02_13)
sizes = df_c02_13['Oil Reserves (Barrels)'] # Collect the data of e
ach country
labels =  df_c02_13['Country']

fig, ax = plt.subplots()
rects1 = ax.bar(labels, sizes, width=0.5, bottom=None, align='cente
r')
plt.ylabel('Oil Reserves(Barrels)')
plt.yticks(np.arange(0, 350000000000, 50000000000),np.arange(0, 350
000000000, 50000000000), fontsize=5) # set the y-axis display
plt.title('Top 15 Countries of Oil Reserves')
plt.xlabel('Country')
plt.xticks(labels, rotation = 90, fontsize = 5) # Rotate the labels
display to make it more clear

def autolabel(rects):
    """Attach a text label above each bar in *rects*, displaying it
s height."""
    for rect in rects:
        height = rect.get_height()
        plt.annotate('{}'.format(height),
                     xy=(rect.get_x() + rect.get_width() / 2, height
),
                     xytext=(0, 3),  # 3 points vertical offset
                     textcoords="offset points",
                     ha='center', va='bottom', fontsize = 2.8)

autolabel(rects1)

plt.show()
```

	Country	Oil Reserves (Barrels)
14	Venezuela	298400000000
11	Saudi Arabia	268300000000
1	Canada	172500000000
3	Iran	157800000000
4	Iraq	144200000000
6	Kuwait	104000000000
10	Russia	103200000000
12	United Arab Emirates	97800000000
7	Libya	48360000000
8	Nigeria	37070000000
13	United States	36520000000
5	Kazakhstan	30000000000
9	Qatar	25240000000
2	China	24650000000
0	Brazil	15310000000



Results:

I applied the code provided in Professor Chen's slides and made a few change to make the display clear and readible. From the graph, we can see the rank of the top 15 countries of oil reserves, but we still need further data to predict about when the world will run out of oil.

2.29

a.

In [139]:
```
df_c02_29 = pd.read_excel('Xr02-29.xlsx')

counts = df_c02_29['Newspaper'].value_counts()
display(counts, counts/ counts.sum())
```

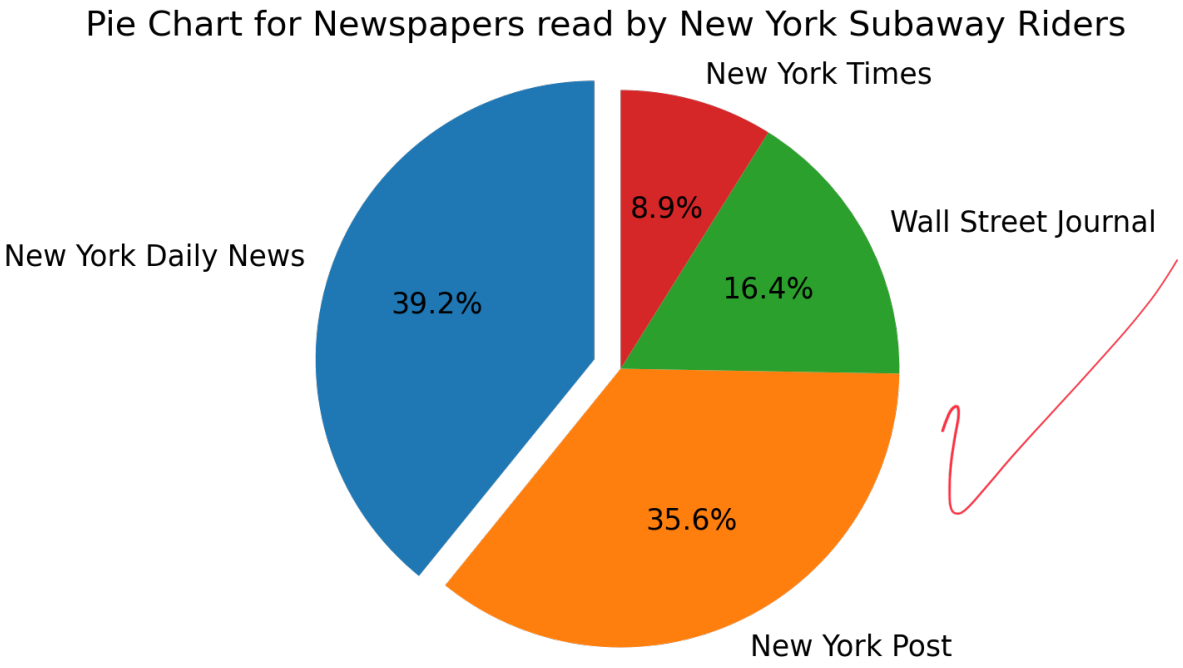
```
1    141
2    128
4     59
3     32
Name: Newspaper, dtype: int64

1    0.391667
2    0.355556
4    0.163889
3    0.088889
Name: Newspaper, dtype: float64
```

Result:

No.	Newspapers	Frequency Distribution	Proportion
1	New York Daily News	141	39.17%
2	New York Post	128	35.56%
3	New York Times	32	8.89%
4	Wall Street Journal	59	16.39%

b.

In [140]:
```
Newspapers = ("New York Daily News", "New York Post", "Wall Street
Journal", "New York Times")
explode = (0.1, 0, 0, 0)  # only "explode" the 1st slice (i.e. 'Bud
weiser Light')
fig2, ax2 = plt.subplots()
ax2.pie(counts, explode=explode, labels=Newspapers, autopct='%1.1f%
%',
        shadow=False, startangle=90)
ax2.axis('equal')  # Equal aspect ratio ensures that pie is drawn a
s a circle.
plt.title('Pie Chart for Newspapers read by New York Subaway Riders
')
plt.show()
```



Result:

From the graph we can know that New York Daily takes the largest part read by the subway riders in New York, with a 39.2% of all riders that took the survey. The second is New York Post with 35.6% and the third is Wall Street Journal with 16.4%. The least one is New York Times

Try to give more "information", instead of just repeating the numbers on the graph :)

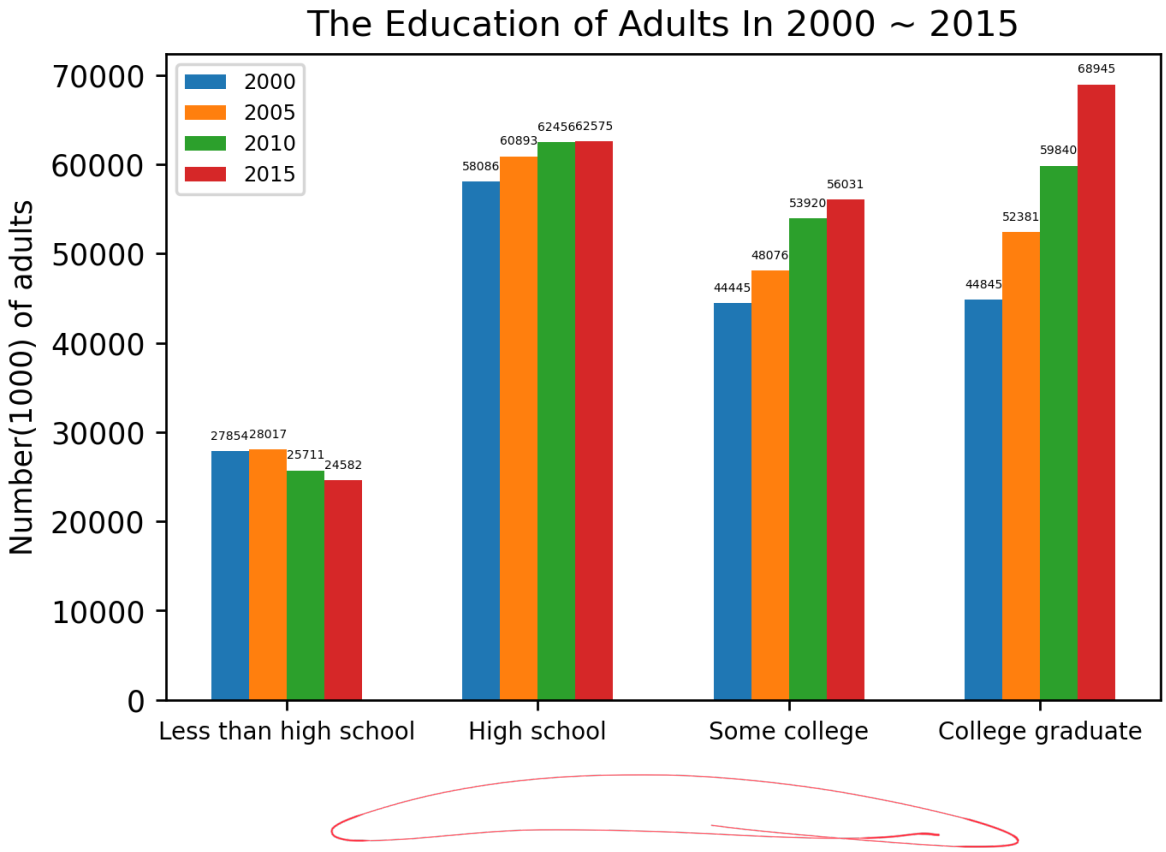
2.41

-5

In [8]:

```
#讀取資料集
df_c02_41 = pd.read_excel('Xr02-41.xlsx')

display(df_c02_41)

def autolabel(rects):
    """Attach a text label above each bar in *rects*, displaying it
s height."""
    for rect in rects:
        height = rect.get_height()
        ax.annotate('{}'.format(height),
                    xy=(rect.get_x() + rect.get_width() / 2, height
),
                    xytext=(0, 3),  # 3 points vertical offset
                    textcoords="offset points",
                    ha='center', va='bottom', fontsize=4)

Education = ("Less than high school", "High school", "Some college"
, "College graduate")
Year = ("2000", "2005", "2010", "2015")
x = np.arange(len(Education))  # the label locations
width = 0.15  # the width of the bars
fig, ax = plt.subplots()
rects1 = ax.bar(x - 3*width/2, df_c02_41[2000], width, label = "200
0")
rects2 = ax.bar(x - width/2, df_c02_41[2005], width, label = "2005"
)
rects3 = ax.bar(x + width/2, df_c02_41[2010], width, label = "2010"
)
rects4 = ax.bar(x + 3*width/2, df_c02_41[2015], width, label = "201
5")
# Add some text for labels, title and custom x-axis tick labels, et
c.
ax.set_ylabel('Number(1000) of adults', fontsize=10)
ax.set_title('The Education of Adults In 2000 ~ 2015')
ax.set_xticks(x)
ax.set_xticklabels(Education, fontsize=8)
ax.legend(fontsize=7)
autolabel(rects1)
autolabel(rects2)
autolabel(rects3)
autolabel(rects4)
plt.show()
```

	Education	2000	2005	2010	2015
0	Less than high school	27854	28017	25711	24582
1	High school	58086	60893	62456	62575
2	Some college	44445	48076	53920	56031
3	College graduate	44845	52381	59840	68945



Result:

From the graph, we can know that adults that have an education higher than high school kept growing since 2000. And those with college graduate had a dramatic growth from 2010 to 2015. On the other hand, the number of adults with education less than high school had been decreasing since 2005. Last but not least, before 2015, the number of adults with high school education was higher than that with college graduate, but in 2015 the number of adults with college graduate became greater than the number of adults with high school education.

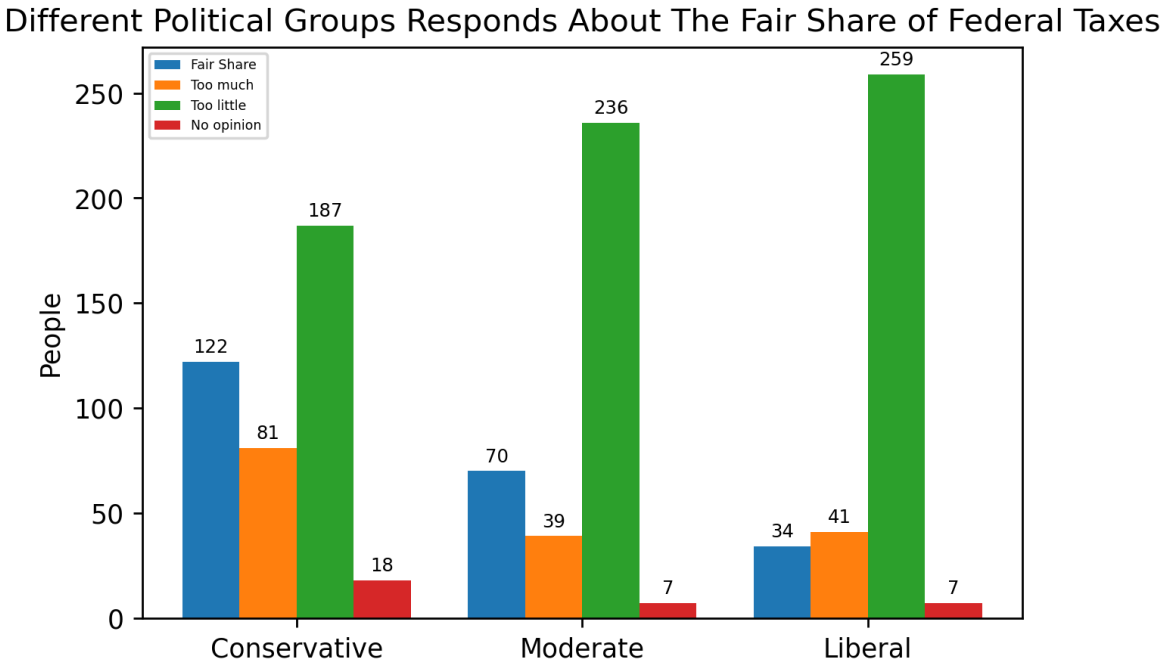
2.51

In [111]:

```
#讀取資料集
df_c02_51 = pd.read_excel('Xr02-35.xlsx')

# 列出Contingency Table
contb = pd.crosstab(df_c02_51["Political View"], df_c02_51["Share"]
)
print("Contingency Table:")
display(contb)

def autolabel(rects):
    """Attach a text label above each bar in *rects*, displaying it
s height."""
    for rect in rects:
        height = rect.get_height()
        ax.annotate('{}'.format(height),
                    xy=(rect.get_x() + rect.get_width() / 2, height
),
                    xytext=(0, 2),  # 3 points vertical offset
                    textcoords="offset points",
                    ha='center', va='bottom', fontsize = 7)

Share = ("Fair Share", "Too much", "Too little", "No opinion")
PV = ("Conservative", "Moderate", "Liberal")
x = np.arange(len(PV))  # the label locations
width = 0.2  # the width of the bars
fig, ax = plt.subplots()
rects1 = ax.bar(x - 3*width/2, contb[1], width, label = "Fair Share
")
rects2 = ax.bar(x - width/2, contb[2], width, label = "Too much")
rects3 = ax.bar(x + width/2, contb[3], width, label =  "Too little"
)
rects4 = ax.bar(x + 3*width/2, contb[4], width, label = "No opinion
")
# Add some text for labels, title and custom x-axis tick labels, et
c.
ax.set_ylabel('People')
ax.set_title('Different Political Groups Responds About The Fair Sh
are of Federal Taxes')
ax.set_xticks(x)
ax.set_xticklabels(PV)
ax.legend(fontsize = 5, loc = 0)
autolabel(rects1)
autolabel(rects2)
autolabel(rects3)
autolabel(rects4)
plt.show()
```

Contingency Table:

Share	1	2	3	4
Political View				
1	122	81	187	18
2	70	39	236	7
3	34	41	259	7



Result:

From the graph, we can know that in all political groups, most people think that the higher-income class pay too little on the fair share of federal taxes. However, in the conservative group there are also a high percentage of people think the share is fair or too much; in liberal group, a huge number of people do think it's too little; in moderate group, more people think the share is fair than those in liberal but still less than the conservative group.