Homework 2

Due: Tuesday Sep 19, at 11:59pm via Blackboard

A car dealership wants to understand their customers and their buying habbits. The data (`cardealership.csv`) represents a randsome sample of their sales.

VARIABLE	DESCRIPTION
Gender	gender for customer
marital status	is the customer 'Married' or 'Single'?
age	age of the customer
country	country make of the car
size	the size of the car they bought ('Small', 'Medium', 'Large')
type	the type of the car they bought ('Family', 'Sporty', 'work')

In [1]:
```
import pandas as pd
import numpy as np

df = pd.read_csv('cardealership.csv')
df.head()
```

Out[1]:

	Gender	marital status	age	country	size	type
0	Male	Married	34	American	Large	Family
1	Male	Single	36	Japanese	Small	Sporty
2	Male	Married	23	Japanese	Small	Family
3	Male	Single	29	American	Large	Family
4	Male	Married	39	American	Medium	Family

In [2]:
```
df.shape[1]
```

Out[2]:
```
6
```

1. (1 point) Select all the married customers in the given dataset, and save it in a variable (`married_customers`). What is the percentage of married customers in the sample?

Answer No 1

In [3]:
```
df['marital status'].unique()
```

Out[3]:
```
array(['Married', 'Single'], dtype=object)
```

In [4]:
```
married_customers = df[df['marital status'] == 'Married']
married_customers.head()

df['marital status'].value_counts(normalize=True)*100
```

Out[4]:
```
marital status
Married    64.686469
Single     35.313531
Name: proportion, dtype: float64
```

In [5]:
```
len(married_customers)/len(df)
```

Out[5]:
```
0.6468646864686468
```

1. (1 point) Use a list comprehension to create a list with two age categories. The category is `Below or equal to 30` if `age <= 30`, otherwise the category is `Above 30`. Use the result from this question to compute the number of customers in each category.

Answer No 2

In [6]:
```
df['Age Categories'] = ['Below 30' if x <= 30 else 'Above 30' for x in df['a
df.head(10)
```

Out[6]:

	Gender	marital status	age	country	size	type	Age Categories
0	Male	Married	34	American	Large	Family	Above 30
1	Male	Single	36	Japanese	Small	Sporty	Above 30
2	Male	Married	23	Japanese	Small	Family	Below 30
3	Male	Single	29	American	Large	Family	Below 30
4	Male	Married	39	American	Medium	Family	Above 30
5	Male	Single	34	Japanese	Medium	Family	Above 30
6	Female	Married	42	American	Large	Family	Above 30
7	Female	Married	40	European	Medium	Family	Above 30
8	Male	Married	28	American	Medium	Sporty	Below 30
9	Female	Married	26	American	Medium	Family	Below 30

In [7]:
```
df[['Age Categories']].value_counts()
```

Out[7]:
```
Age Categories
Below 30          159
Above 30          144
Name: count, dtype: int64
```

1. (2 points) The current version of `Pandas` has 142 methods including (`DataFrame()`, `Series()`, `value_counts()`, etc.). In this question, you are expected to learn about the `cut()` method which allows you to categorize a numerical vector into user-defined categories. Click here to learn more about the `cut` method.

    - Use the `cut()` method to categorize the `age` variable into three buckets: `(0,30]`, `(30, 34]`, and `(34,60]`. (For this exercise, you don't have to add the new column to the original dataframe. You can save it in a seperate variable instead)

- Rename the labels of the buckets to the ones shown in the table below.
- How many element are there in each category?

bucket	label
(0,30]	Below 30
(30, 34]	Between 30 and 34
(34,60]	Above 34

Answer No 3

In [8]:
```
df['bucket'] = pd.cut(x=df['age'], bins=[0, 30, 34, 60])
df['label'] = pd.cut(x=df['age'], bins=[0, 30, 34, 60], labels=['Below 30',
df.head()
```

Out[8]:

	Gender	marital status	age	country	size	type	Age Categories	bucket	label
0	Male	Married	34	American	Large	Family	Above 30	(30, 34]	Between 30 and 34
1	Male	Single	36	Japanese	Small	Sporty	Above 30	(34, 60]	Above 34
2	Male	Married	23	Japanese	Small	Family	Below 30	(0, 30]	Below 30
3	Male	Single	29	American	Large	Family	Below 30	(0, 30]	Below 30
4	Male	Married	39	American	Medium	Family	Above 30	(34, 60]	Above 34

In [9]:
```
df[['label']].value_counts()
```

Out[9]:
```
label
Below 30              159
Above 34               76
Between 30 and 34      68
Name: count, dtype: int64
```

There are 159 elements for the age below 30, 76 elements for the age above 34 and 68 elements for the age between 30 and 34

1. (1 point) `Pandas` has another method called `qcut`, which allows you to categorize a numerical variable into equal-sized buckets based on quantiles. Use the `qcut()` method to categorize `age` into quartiles (4 buckets). Click here to learn more about the `cut` method

Answer No 4

In [10]:
```
df['qcut_bucket'] = pd.qcut(x=df['age'], q = 4)
df.head()
```

Out[10]:

	Gender	marital status	age	country	size	type	Age Categories	bucket	label	qcut_bucl
0	Male	Married	34	American	Large	Family	Above 30	(30, 34]	Between 30 and 34	(30.0, 34
1	Male	Single	36	Japanese	Small	Sporty	Above 30	(34, 60]	Above 34	(34.5, 60
2	Male	Married	23	Japanese	Small	Family	Below 30	(0, 30]	Below 30	(17.9 26
3	Male	Single	29	American	Large	Family	Below 30	(0, 30]	Below 30	(26.0, 30
4	Male	Married	39	American	Medium	Family	Above 30	(34, 60]	Above 34	(34.5, 60

In [11]:
```
df.qcut_bucket.unique()
```

Out[11]:
```
[(30.0, 34.5], (34.5, 60.0], (17.999, 26.0], (26.0, 30.0]]
Categories (4, interval[float64, right]): [(17.999, 26.0] < (26.0, 30.0] <
(30.0, 34.5] < (34.5, 60.0]]
```

In [12]:
```
df[['qcut_bucket']].value_counts()
```

Out[12]:
```
qcut_bucket
(17.999, 26.0]    85
(34.5, 60.0]      76
(26.0, 30.0]      74
(30.0, 34.5]      68
Name: count, dtype: int64
```

1. (1 point) Using `pandas`, summarize the customer characteristics: `Gender`, `marital status` (using relative frequency tables) and `age` (using the `describe()` method).

Answer No 5

In [13]:
```
df['marital status'].value_counts(normalize=True)*100
```

Out[13]:
```
marital status
Married    64.686469
Single     35.313531
Name: proportion, dtype: float64
```

In [14]:
```
df['Gender'].value_counts(normalize=True)*100
```

Out[14]:
```
Gender
Male      54.455446
Female    45.544554
Name: proportion, dtype: float64
```

In [15]:
```
df['age'].describe()
```

Out[15]:
```
count    303.000000
mean      30.719472
std        5.984294
min       18.000000
25%       26.000000
50%       30.000000
75%       34.500000
max       60.000000
Name: age, dtype: float64
```

1. (1 point) Using `pandas`, summarize the data on the cars sold: `country`, `size`, and `type` (using relative frequency tables).

Answer No 6

In [16]:
```
df['country'].value_counts(normalize=True)*100
```

Out[16]:
```
country
Japanese    48.844884
American    37.953795
European    13.201320
Name: proportion, dtype: float64
```

In [17]:
```
df['size'].value_counts(normalize=True)*100
```

Out[17]:
```
size
Small     45.214521
Medium    40.924092
Large     13.861386
Name: proportion, dtype: float64
```

In [18]:
```
df['type'].value_counts(normalize=True)*100
```

Out[18]:
```
type
Family    51.155116
Sporty    33.003300
Work      15.841584
Name: proportion, dtype: float64
```

1. (1 point) Write a summary paragraph describing the customers and cars sold data. Round all numbers in this paragraph to nearest integers.

Answer No 7

In [19]:
```
df.head()
```

Out[19]:

	Gender	marital status	age	country	size	type	Age Categories	bucket	label	qcut_bucl
0	Male	Married	34	American	Large	Family	Above 30	(30, 34]	Between 30 and 34	(30.0, 34
1	Male	Single	36	Japanese	Small	Sporty	Above 30	(34, 60]	Above 34	(34.5, 60
2	Male	Married	23	Japanese	Small	Family	Below 30	(0, 30]	Below 30	(17.9 26
3	Male	Single	29	American	Large	Family	Below 30	(0, 30]	Below 30	(26.0, 30
4	Male	Married	39	American	Medium	Family	Above 30	(34, 60]	Above 34	(34.5, 60

In [20]:
```
df[['Gender', 'marital status']].value_counts(normalize=True).unstack()
```

Out[20]:

marital status	Married	Single
Gender		
Female	0.313531	0.141914
Male	0.333333	0.211221

In [21]:
```
df['marital status'].value_counts(normalize=True)
```

Out[21]:
```
marital status
Married    0.646865
Single     0.353135
Name: proportion, dtype: float64
```

In [22]:
```
df['Gender'].value_counts(normalize=True)
```

Out[22]:
```
Gender
Male      0.544554
Female    0.455446
Name: proportion, dtype: float64
```

In [23]:
```
df['label'].value_counts(normalize=True)
```

Out[23]:
```
label
Below 30             0.524752
Above 34             0.250825
Between 30 and 34    0.224422
Name: proportion, dtype: float64
```

In [24]:
```
df['type'].value_counts(normalize=True)
```

Out[24]:
```
type
Family    0.511551
Sporty    0.330033
Work      0.158416
Name: proportion, dtype: float64
```

In [25]:
```
df['size'].value_counts(normalize=True)
```

Out[25]:
```
size
Small     0.452145
Medium    0.409241
Large     0.138614
Name: proportion, dtype: float64
```

In [26]:
```
df['country'].value_counts(normalize=True)
```

Out[26]:
```
country
Japanese    0.488449
American    0.379538
European    0.132013
Name: proportion, dtype: float64
```

Summary

The car buyers in this case is dominated by married people with almost 50:50 on the gender distribution between male and female. On the age perspective, the majority of buyers are coming from people with age below 30 covering 52% of the total buyers. Seeing the car type category, family type is dominant compared to the other types with 51% proportion and it has connection to the insight that the cars are mostly bought by the married people. In addition, the buyers mostly purchase small and medium car with 45% and 41% in proportion. Then, seeing the car origin, the car sold are mostly produced in Japan, covering 49% of total car sales.

1. (2 points) Create a bargraph that shows the distribution of car `type`. Your bargraph should be similar to the attached bargraph picture on blackboard ('CarsTypeDistribution.png'). In particular, make sure to:

- Use default matplotlib plot style
- Use % for the labels of the y-axis ticks
- Use `lightgrey` for the bars color
- Overlay a horizontal line (y=25). The line's style is "dashed", and the color is "blue"

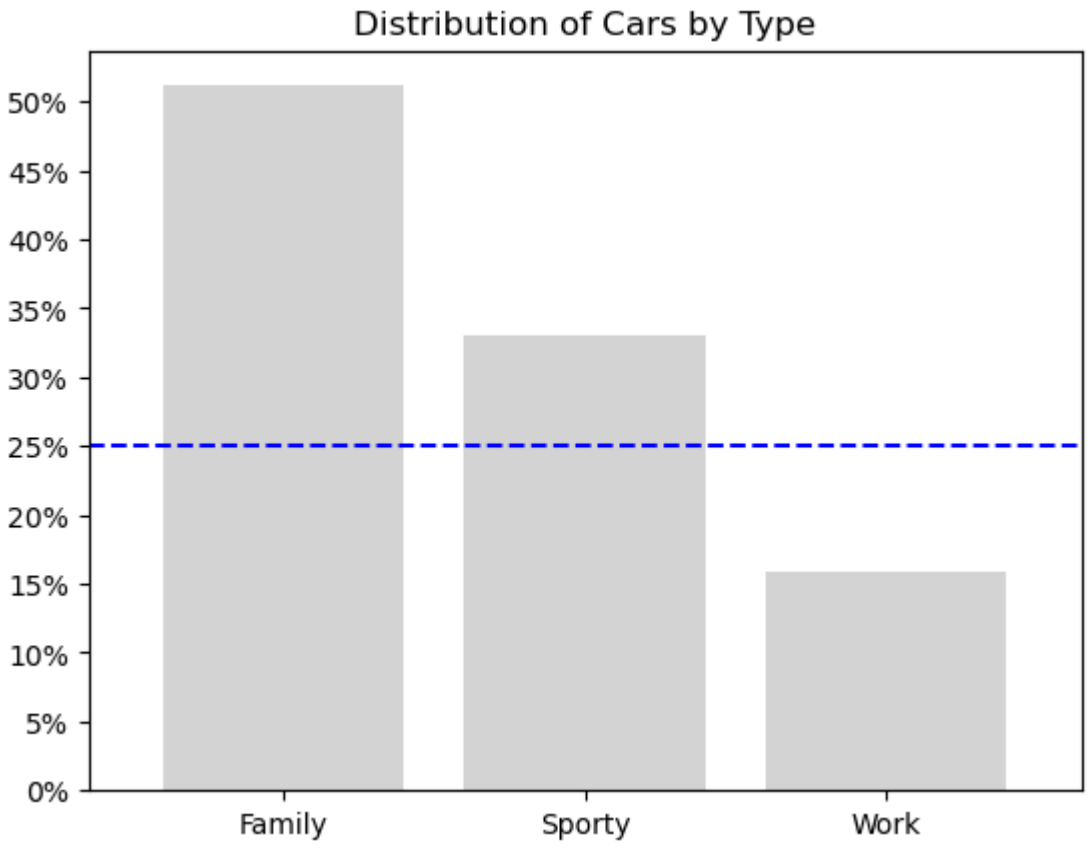
Answer No 8

In [27]:
```
import matplotlib.pyplot as plt
import matplotlib.ticker as mtick

df_chart = pd.DataFrame(df['type'].value_counts(normalize = True))
df_chart['proportion'] = df_chart['proportion']*100
labels = list(df_chart.index)

ax = df_chart.plot(kind='bar', color = 'lightgrey', width = 0.8)
ax.set_yticks(range(0, 55, 5))
ax.yaxis.set_major_formatter(mtick.PercentFormatter())
ax.set_xlabel('')
ax.get_legend().remove()
plt.axhline(y = 25, color = 'b', linestyle = '--')
plt.title('Distribution of Cars by Type')
ax.set_xticklabels(labels=labels, rotation = 360)

plt.show()
```


Distribution of Cars by Type
50%
45%
40%
35%
30%
25%
20%
15%
10%
5%
0%
Family
Sporty
Work