In [141…

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import expon
from scipy.stats import norm
from scipy.stats import binom
from scipy.stats import poisson
import math
```

In [142…

```
df_walmart = pd.read_csv("https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/293/original/walmart
```

In [143…

```
df_walmart.head()
```

Out[143…

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category	Purchase
0	1000001	P00069042	F	0-17	10	A	2	0	3	8370
1	1000001	P00248942	F	0-17	10	A	2	0	1	15200
2	1000001	P00087842	F	0-17	10	A	2	0	12	1422
3	1000001	P00085442	F	0-17	10	A	2	0	12	1057
4	1000002	P00285442	M	55+	16	C	4+	0	8	7969

In [144…

```
df_walmart.nunique()
```

Out[144…

```
User_ID                        5891
Product_ID                     3631
Gender                            2
Age                               7
Occupation                       21
City_Category                     3
Stay_In_Current_City_Years        5
Marital_Status                    2
Product_Category                 20
Purchase                      18105
dtype: int64
```

In [145…

```
df_walmart.shape
```

Out[145…

```
(550068, 10)
```

In [146…

```
df_walmart.isna().sum()
```

Out[146…

```
User_ID                       0
Product_ID                    0
Gender                        0
Age                           0
Occupation                    0
City_Category                 0
Stay_In_Current_City_Years    0
Marital_Status                0
Product_Category              0
Purchase                      0
dtype: int64
```

In [147…

```
df_walmart.isnull().sum()
```

Out[147…

```
User_ID                       0
Product_ID                    0
Gender                        0
Age                           0
Occupation                    0
City_Category                 0
Stay_In_Current_City_Years    0
Marital_Status                0
Product_Category              0
Purchase                      0
dtype: int64
```

In [148…

```
df_walmart.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 550068 entries, 0 to 550067
Data columns (total 10 columns):
 #   Column                      Non-Null Count   Dtype 
---  ------                      --------------   ----- 
 0   User_ID                     550068 non-null  int64 
 1   Product_ID                  550068 non-null  object
 2   Gender                      550068 non-null  object
 3   Age                         550068 non-null  object
 4   Occupation                  550068 non-null  int64 
 5   City_Category               550068 non-null  object
 6   Stay_In_Current_City_Years  550068 non-null  object
 7   Marital_Status              550068 non-null  int64 
 8   Product_Category            550068 non-null  int64 
 9   Purchase                    550068 non-null  int64 
dtypes: int64(5), object(5)
memory usage: 42.0+ MB
```

In [149…

```
df_walmart.describe()
```

Out[149…

	User_ID	Occupation	Marital_Status	Product_Category	Purchase
count	5.500680e+05	550068.000000	550068.000000	550068.000000	550068.000000
mean	1.003029e+06	8.076707	0.409653	5.404270	9263.968713
std	1.727592e+03	6.522660	0.491770	3.936211	5023.065394
min	1.000001e+06	0.000000	0.000000	1.000000	12.000000
25%	1.001516e+06	2.000000	0.000000	1.000000	5823.000000
50%	1.003077e+06	7.000000	0.000000	5.000000	8047.000000
75%	1.004478e+06	14.000000	1.000000	8.000000	12054.000000
max	1.006040e+06	20.000000	1.000000	20.000000	23961.000000

In [150…

```
df_walmart.drop_duplicates(inplace=True)
```

In [151…

```
df_walmart.head()
```

Out[151…

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category	Purchase
0	1000001	P00069042	F	0-17	10	A	2	0	3	8370
1	1000001	P00248942	F	0-17	10	A	2	0	1	15200
2	1000001	P00087842	F	0-17	10	A	2	0	12	1422
3	1000001	P00085442	F	0-17	10	A	2	0	12	1057
4	1000002	P00285442	M	55+	16	C	4+	0	8	7969

In [152…

```
df_Male = df_walmart[df_walmart["Gender"] == "M"]
```

In [153…

```
df_Male
```

Out[153…

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category	Purchase
4	1000002	P00285442	M	55+	16	C	4+	0	8	7969
5	1000003	P00193542	M	26-35	15	A	3	0	1	15227
6	1000004	P00184942	M	46-50	7	B	2	1	1	19215
7	1000004	P00346142	M	46-50	7	B	2	1	1	15854
8	1000004	P0097242	M	46-50	7	B	2	1	1	15686
...	...	...	...	...	...	...	...	...	...	...
550057	1006023	P00370853	M	26-35	0	C	2	1	19	61

550058	1006024	P00372445	M	26-35	12	A	0	1	20	121
550060	1006026	P00371644	M	36-45	6	C	1	1	20	494
550062	1006032	P00372445	M	46-50	7	A	3	0	20	473
550063	1006033	P00372445	M	51-55	13	B	1	1	20	368

414259 rows × 10 columns

In [154…

```
df_Female = df_walmart[df_walmart["Gender"] == "F"]
```

In [155…

```
df_Female
```

Out[155…

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category	Purchase
0	1000001	P00069042	F	0-17	10	A	2	0	3	8370
1	1000001	P00248942	F	0-17	10	A	2	0	1	15200
2	1000001	P00087842	F	0-17	10	A	2	0	12	1422
3	1000001	P00085442	F	0-17	10	A	2	0	12	1057
14	1000006	P00231342	F	51-55	9	A	1	0	5	5378
...	...	...	...	...	...	...	...	...	...	...
550061	1006029	P00372445	F	26-35	1	C	1	1	20	599
550064	1006035	P00375436	F	26-35	1	C	3	0	20	371
550065	1006036	P00375436	F	26-35	15	B	4+	1	20	137
550066	1006038	P00375436	F	55+	1	C	2	0	20	365
550067	1006039	P00371644	F	46-50	0	B	4+	1	20	490

135809 rows × 10 columns

In [156…

```
df_walmart.groupby(["Gender"])["Purchase"].mean()
```

Out[156…

```
Gender
F    8734.565765
M    9437.526040
Name: Purchase, dtype: float64
```

In [157…

```
df_Male["Purchase"].mean()
```

Out[157…

```
9437.526040472265
```

In [158…

```
df_Female["Purchase"].mean()
```

Out[158…

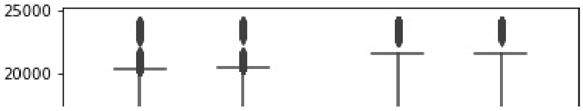
```
8734.565765155476
```

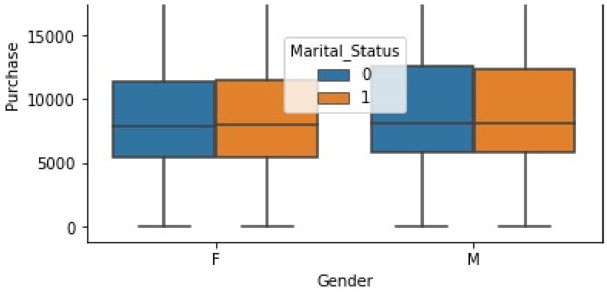
In [159…

```
sns.boxplot(x = "Gender",y = "Purchase",hue = "Marital_Status",data = df_walmart)
```

Out[159…

```
<AxesSubplot:xlabel='Gender', ylabel='Purchase'>
```





In [160…
```
P_25_Male = np.percentile(df_Male["Purchase"],25)
```

In [161…
```
P_25_Male
```

Out[161…
```
5863.0
```

In [162…
```
P_75_Male = np.percentile(df_Male["Purchase"],75)
```

In [163…
```
P_75_Male
```

Out[163…
```
12454.0
```

In [164…
```
Male_iqr = P_75_Male - P_25_Male
```

In [165…
```
Male_iqr
```

Out[165…
```
6591.0
```

In [166…
```
Male_lower = max(P_25_Male - 1.5*Male_iqr,0)
```

In [167…
```
Male_lower
```

Out[167…
```
0
```

In [168…
```
Male_upper = P_75_Male + 1.5*Male_iqr
```

In [169…
```
Male_upper
```

Out[169…
```
22340.5
```

In [170…
```
df_Male_new = df_Male[df_Male["Purchase"] <= Male_upper]
```

In [171…
```
df_Male_new
```

Out[171…

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category	Purchase
4	1000002	P00285442	M	55+	16	C	4+	0	8	7969
5	1000003	P00193542	M	26-35	15	A	3	0	1	15227
6	1000004	P00184942	M	46-50	7	B	2	1	1	19215
7	1000004	P00346142	M	46-50	7	B	2	1	1	15854
8	1000004	P0097242	M	46-50	7	B	2	1	1	15686

...	...	...	...	...	...	...	...	...	...	...
550057	1006023	P00370853	M	26-35	0	C	2	1	19	61
550058	1006024	P00372445	M	26-35	12	A	0	1	20	121
550060	1006026	P00371644	M	36-45	6	C	1	1	20	494
550062	1006032	P00372445	M	46-50	7	A	3	0	20	473
550063	1006033	P00372445	M	51-55	13	B	1	1	20	368

412447 rows × 10 columns

In [172…

```
P_25_Female = np.percentile(df_Female["Purchase"],25)
```

In [173…

```
P_25_Female
```

Out[173…

```
5433.0
```

In [174…

```
P_75_Female = np.percentile(df_Female["Purchase"],75)
```

In [175…

```
P_75_Female
```

Out[175…

```
11400.0
```

In [176…

```
Female_iqr = P_75_Female - P_25_Female
```

In [177…

```
Female_iqr
```

Out[177…

```
5967.0
```

In [178…

```
Female_lower = max(P_25_Female - 1.5*Female_iqr,0)
```

In [179…

```
Female_lower
```

Out[179…

```
0
```

In [180…

```
Female_upper = P_75_Female + 1.5*Female_iqr
```

In [181…

```
Female_upper
```

Out[181…

```
20350.5
```

In [182…

```
df_Female_new = df_Female[df_Female["Purchase"] <= 20350.5]
```

In [183…

```
df_Female_new
```

Out[183…

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category	Purchase
0	1000001	P00069042	F	0-17	10	A	2	0	3	8370
1	1000001	P00248942	F	0-17	10	A	2	0	1	15200
2	1000001	P00087842	F	0-17	10	A	2	0	12	1422

3	1000001	P00085442	F	0-17	10	A	2	0	12	1057
14	1000006	P00231342	F	51-55	9	A	1	0	5	5378
...	...	...	...	...	...	...	...	...	...	...
550061	1006029	P00372445	F	26-35	1	C	1	1	20	599
550064	1006035	P00375436	F	26-35	1	C	3	0	20	371
550065	1006036	P00375436	F	26-35	15	B	4+	1	20	137
550066	1006038	P00375436	F	55+	1	C	2	0	20	365
550067	1006039	P00371644	F	46-50	0	B	4+	1	20	490

133744 rows × 10 columns

In [194…

```
Mu_male = df_Male_new["Purchase"].mean()
```

In [197…

```
Mu_male
```

Out[197…

```
9375.837603376918
```

In [198…

```
Sigma_male = df_Male_new["Purchase"].std()
```

In [199…

```
Sigma_male
```

Out[199…

```
5017.363604662484
```

In [213…

```
Mu_female = df_Female_new["Purchase"].mean()
Mu_female
```

Out[213…

```
8538.524636619213
```

In [214…

```
Sigma_female = df_Female_new["Purchase"].std()
Sigma_female
```

Out[214…

```
4530.515859709995
```

In [201…

```
norm.ppf(0.95)
```

Out[201…

```
1.6448536269514722
```

In [202…

```
norm.ppf(0.05)
```

Out[202…

```
-1.6448536269514729
```

In [203…

```
norm.ppf(0.975)
```

Out[203…

```
1.959963984540054
```

In [204…

```
norm.ppf(0.025)
```

Out[204…

```
-1.9599639845400545
```

In [205…

```
norm.ppf(0.995)
```

Out[205…

```
2.5758293035489004
```

In [206…

```
norm.ppf(0.005)
```

Out[206…

```
-2.575829303548901
```

In [236…

```
left_male_90 = Mu_male - 1.6448536269514722 * Sigma_male/math.sqrt(len(df_Male_new))
left_male_90
```

Out[236…

```
9362.987140911348
```

In [237…

```
right_male_90 = Mu_male + 1.6448536269514722 * Sigma_male/math.sqrt(len(df_Male_new))
right_male_90
```

Out[237…

```
9388.688065842489
```

In [238…

```
print(f"90% confidence that the purchase mean for male is in {left_male_90}, {right_male_90}")
```

```
90% confidence that the purchase mean for male is in 9362.987140911348, 9388.688065842489
```

In [239…

```
left_female_90 = Mu_female - 1.6448536269514722 * Sigma_female/math.sqrt(len(df_Female_new))
left_female_90
```

Out[239…

```
8518.147753334202
```

In [240…

```
right_female_90 = Mu_female + 1.6448536269514722 * Sigma_female/math.sqrt(len(df_Female_new))
right_female_90
```

Out[240…

```
8558.901519904224
```

In [241…

```
print(f"90% confidence that the purchase mean for female is in {left_female_90}, {right_female_90}")
```

```
90% confidence that the purchase mean for female is in 8518.147753334202, 8558.901519904224
```

In [242…

```
left_male_95 = Mu_male - 1.959963984540054 * Sigma_male/math.sqrt(len(df_Male_new))
left_male_95
```

Out[242…

```
9360.525332908279
```

In [243…

```
right_male_95 = Mu_male + 1.959963984540054 * Sigma_male/math.sqrt(len(df_Male_new))
right_male_95
```

Out[243…

```
9391.149873845558
```

In [247…

```
print(f"95% confidence that the purchase mean for male is in {left_male_95}, {right_male_95}")
```

```
95% confidence that the purchase mean for male is in 9360.525332908279, 9391.149873845558
```

In [248…

```
left_female_95 = Mu_female - 1.959963984540054 * Sigma_female/math.sqrt(len(df_Female_new))
```

```
left_female_95 = Mu_female - 1.959963984540054 * Sigma_female/math.sqrt(len(df_Female_new))
left_female_95
```

Out[248…

```
8514.244082592155
```

In [249…

```
right_female_95 = Mu_female + 1.959963984540054 * Sigma_female/math.sqrt(len(df_Female_new))
right_female_95
```

Out[249…

```
8562.805190646272
```

In [250…

```
print(f"95% confidence that the purchase mean for female is in {left_female_95}, {right_female_95}")
```

```
95% confidence that the purchase mean for female is in 8514.244082592155, 8562.805190646272
```

In [251…

```
left_male_99 = Mu_male - 2.5758293035489004 * Sigma_male/math.sqrt(len(df_Male_new))
left_male_99
```

Out[251…

```
9355.713868814397
```

In [252…

```
right_male_99 = Mu_male + 2.5758293035489004 * Sigma_male/math.sqrt(len(df_Male_new))
right_male_99
```

Out[252…

```
9395.96133793944
```

In [253…

```
print(f"99% confidence that the purchase mean for male is in {left_male_99}, {right_male_99}")
```

```
99% confidence that the purchase mean for male is in 9355.713868814397, 9395.96133793944
```

In [254…

```
left_female_99 = Mu_female - 2.5758293035489004 * Sigma_female/math.sqrt(len(df_Female_new))
left_female_99
```

Out[254…

```
8506.614579565805
```

In [257…

```
right_female_99 = Mu_female + 2.5758293035489004 * Sigma_female/math.sqrt(len(df_Female_new))
right_female_99
```

Out[257…

```
8570.434693672622
```

In [258…

```
print(f"99% confidence that the purchase mean for female is in {left_female_99}, {right_female_99}")
```

```
99% confidence that the purchase mean for female is in 8506.614579565805, 8570.434693672622
```

In [265…

```
Male_samples = np.random.choice(df_Male_new["Purchase"])
print(Male_samples)
np.mean(Male_samples)
```

```
11611
```

Out[265…

```
11611.0
```

In [283…

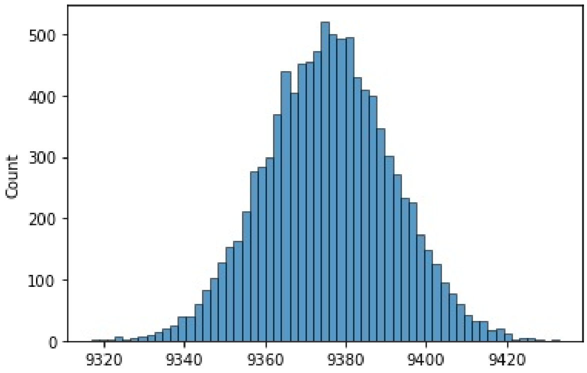
```
Male_means_survey = []
for rep in range(10000):
    Male_samples = np.random.choice(df_Male_new["Purchase"],size = 100000)
    Male_mean = np.mean(Male_samples)
    Male_means_survey.append(Male_mean)
```

In [284…

```
sns.histplot(Male_means_survey)
```

Out[284…

```
<AxesSubplot:ylabel='Count'>
```



In [285…

```
Maleupper = np.percentile(Male_means_survey, 97.5)
Maleupper
```

Out[285…

```
9406.80583275
```

In [310…

```
Maleupper = np.percentile(Male_means_survey, 95)
Maleupper
```

Out[310…

```
9401.914526499999
```

In [311…

```
Maleupper = np.percentile(Male_means_survey, 99.5)
Maleupper
```

Out[311…

```
9416.93187575
```

In [286…

```
Malelower = np.percentile(Male_means_survey, 2.5)
Malelower
```

Out[286…

```
9344.739201
```

In [312…

```
Malelower = np.percentile(Male_means_survey, 5)
Malelower
```

Out[312…

```
9349.796991
```

In [313…

```
Malelower = np.percentile(Male_means_survey, 0.5)
Malelower
```

Out[313…

```
9334.6265842
```

In [287…

```
Female_samples = np.random.choice(df_Male_new["Purchase"])
print(Female_samples)
np.mean(Female_samples)
```

```
11728
```

Out[287…

```
11728.0
```

In [288…

```
Female_means_survey = []
```

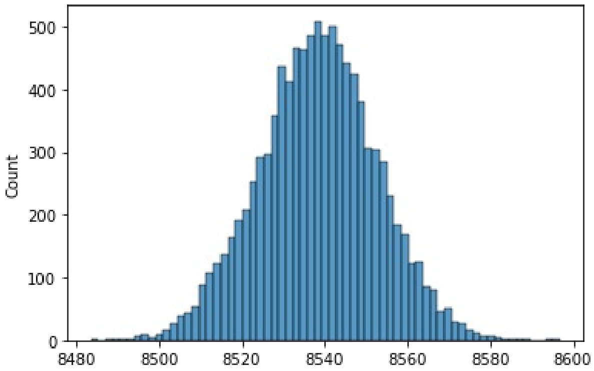
```
for rep in range(10000):
    Female_samples = np.random.choice(df_Female_new["Purchase"],size = 100000)
    Female_mean = np.mean(Female_samples)
    Female_means_survey.append(Female_mean)
```

In [289…

```
sns.histplot(Female_means_survey)
```

Out[289…

```
<AxesSubplot:ylabel='Count'>
```



In [290…

```
Femaleupper = np.percentile(Female_means_survey, 97.5)
Femaleupper
```

Out[290…

```
8566.24343575
```

In [314…

```
Femaleupper = np.percentile(Female_means_survey, 95)
Femaleupper
```

Out[314…

```
8562.0411785
```

In [315…

```
Femaleupper = np.percentile(Female_means_survey, 99.5)
Femaleupper
```

Out[315…

```
8575.00517585
```

In [291…

```
Femalelower = np.percentile(Female_means_survey, 2.5)
Femalelower
```

Out[291…

```
8510.52377375
```

In [316…

```
Femalelower = np.percentile(Female_means_survey, 5)
Femalelower
```

Out[316…

```
8514.4598705
```

In [317…

```
Femalelower = np.percentile(Female_means_survey, 0.5)
Femalelower
```

Out[317…

```
8502.33007675
```

In [292…

```
df_walmart.head()
```

Out[292…

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category	Purchase
0	1000001	P00069042	F	0-17	10	A	2	0	3	8370
1	1000001	P00248942	F	0-17	10	A	2	0	1	15200
2	1000001	P00087842	F	0-17	10	A	2	0	12	1422

3	1000001	P00085442	F	0-17	10	A	2	0	12	1057
4	1000002	P00285442	M	55+	16	C	4+	0	8	7969

In [293…

```
df_walmart["Marital_Status"].value_counts()
```

Out[293…

```
0    324731
1    225337
Name: Marital_Status, dtype: int64
```

In [295…

```
df_Married = df_walmart[df_walmart["Marital_Status"] == 0]
```

In [298…

```
df_Married.shape
```

Out[298…

```
(324731, 10)
```

In [299…

```
df_Unmarried = df_walmart[df_walmart["Marital_Status"] == 1]
```

In [300…

```
df_Unmarried.shape
```

Out[300…

```
(225337, 10)
```

In [301…

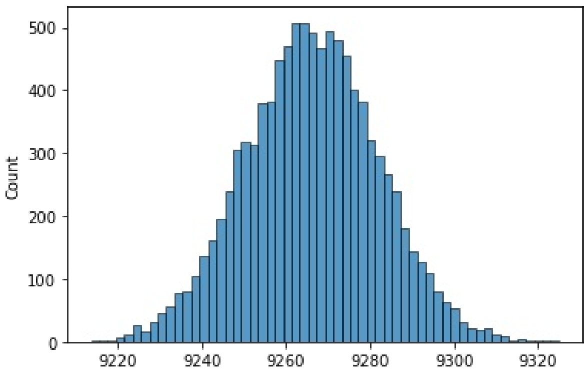
```
Married_means_survey = []
for rep in range(10000):
    Married_samples = np.random.choice(df_Married["Purchase"],size = 100000)
    Married_mean = np.mean(Married_samples)
    Married_means_survey.append(Married_mean)
```

In [302…

```
sns.histplot(Married_means_survey)
```

Out[302…

```
<AxesSubplot:ylabel='Count'>
```



In [303…

```
Marriedupper = np.percentile(Married_means_survey, 97.5)
Marriedupper
```

Out[303…

```
9296.978973000001
```

In [318…

```
Marriedupper = np.percentile(Married_means_survey, 95)
Marriedupper
```

Out[318…

```
9291.986185000002
```

In [319…

```
Marriedupper = np.percentile(Married_means_survey, 99.5)
Marriedupper
```

Out[319…

```
9307.93135165
```

In [305…

```
Marriedlower = np.percentile(Married_means_survey, 2.5)
Marriedlower
```

Out[305…

```
9234.877355
```

In [320…

```
Marriedlower = np.percentile(Married_means_survey, 5)
Marriedlower
```

Out[320…

```
9239.923171
```

In [321…

```
Marriedlower = np.percentile(Married_means_survey, 0.5)
Marriedlower
```

Out[321…

```
9225.4731507
```

In [341…

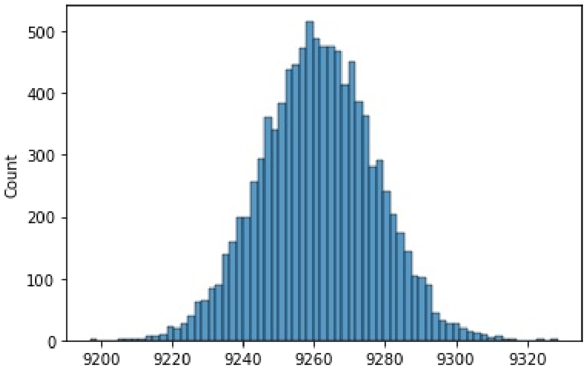
```
Unmarried_means_survey = []
for rep in range(10000):
    Unmarried_samples = np.random.choice(df_Unmarried["Purchase"],size = 100000)
    Unmarried_mean = np.mean(Unmarried_samples)
    Unmarried_means_survey.append(Unmarried_mean)
```

In [307…

```
sns.histplot(Unmarried_means_survey)
```

Out[307…

```
<AxesSubplot:ylabel='Count'>
```



In [308…

```
Unmarriedupper = np.percentile(Unmarried_means_survey, 97.5)
Unmarriedupper
```

Out[308…

```
9292.28162125
```

In [322…

```
Unmarriedupper = np.percentile(Unmarried_means_survey, 95)
Unmarriedupper
```

Out[322…

```
9287.419857500001
```

In [323…

```
Unmarriedupper = np.percentile(Unmarried_means_survey, 99.5)
Unmarriedupper
```

Out[323…

```
9303.51742685
```

In [309…

```
Unmarriedlower = np.percentile(Unmarried_means_survey, 2.5)
Unmarriedlower
```

Out[309…

```
9229.90211175
```

In [324…

```
Unmarriedlower = np.percentile(Unmarried_means_survey, 5)
Unmarriedlower
```

Out[324…

```
9235.1515575
```

In [325…

```
Unmarriedlower = np.percentile(Unmarried_means_survey, 0.5)
Unmarriedlower
```

Out[325…

```
9220.26553705
```

In [327…

```
df_walmart["Age"].value_counts()
```

Out[327…

```
26-35    219587
36-45    110013
18-25     99660
46-50     45701
51-55     38501
55+       21504
0-17      15102
Name: Age, dtype: int64
```

In [331…

```
df_walmart.groupby(["Age"])["Purchase"].mean()
```

Out[331…

```
Age
0-17     8933.464640
18-25    9169.663606
26-35    9252.690633
36-45    9331.350695
46-50    9208.625697
51-55    9534.808031
55+      9336.280459
Name: Purchase, dtype: float64
```

In [333…

```
df_walmart.groupby(["Age"])["Purchase"].std()
```

Out[333…

```
Age
0-17     5111.114046
18-25    5034.321997
26-35    5010.527303
36-45    5022.923879
46-50    4967.216367
51-55    5087.368080
55+      5011.493996
Name: Purchase, dtype: float64
```

In [343…

```
df_newage = df_walmart["Purchase"][df_walmart["Age"] == "0-17"]
df_newage
```

Out[343…

```
0          8370
1         15200
2          1422
3          1057
85         7746
          ...  
549904      256
550012       26
550024       12
550035       61
550046      236
Name: Purchase, Length: 15102, dtype: int64
```

In [344…

```
np.random.choice(df_newage,size = 10)
```

Out[344…

```
array([ 6882,  7124, 10872, 19201,  7931, 15863, 16369, 21215, 21245,
```

Out[344…

```
       11890])
```

In [345…

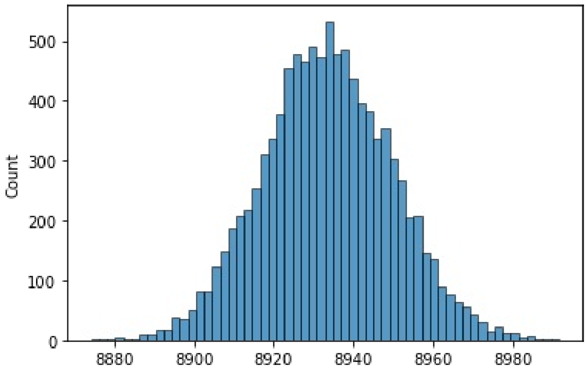
```
Age_means_survey = []
for rep in range(10000):
    Age_samples = np.random.choice(df_newage,size = 100000)
    Age_mean = np.mean(Age_samples)
    Age_means_survey.append(Age_mean)
```

In [346…

```
sns.histplot(Age_means_survey)
```

Out[346…

```
<AxesSubplot:ylabel='Count'>
```



In [348…

```
Ageupper = np.percentile(Age_means_survey, 97.5)
Ageupper
```

Out[348…

```
8965.79441975
```

In [347…

```
Agelower = np.percentile(Age_means_survey, 2.5)
Agelower
```

Out[347…

```
8901.941057249998
```

In [350…

```
df_newage = df_walmart["Purchase"][df_walmart["Age"] == "18-25"]
```

In [351…

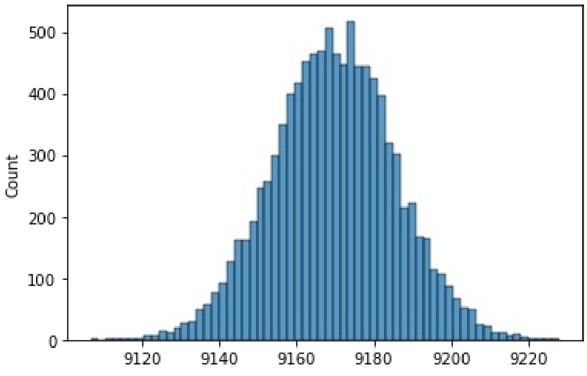
```
Age_means_survey = []
for rep in range(10000):
    Age_samples = np.random.choice(df_newage,size = 100000)
    Age_mean = np.mean(Age_samples)
    Age_means_survey.append(Age_mean)
```

In [352…

```
sns.histplot(Age_means_survey)
```

Out[352…

```
<AxesSubplot:ylabel='Count'>
```



In [356…

```
Ageupper = np.percentile(Age_means_survey, 97.5)
```

```
print(Ageupper)
Agelower = np.percentile(Age_means_survey, 2.5)
print(Agelower)
```

```
9201.208897
9138.38198825
```

In [ ]:

In [357…

```
df_newage = df_walmart["Purchase"][df_walmart["Age"] == "26-35"]
```

In [358…

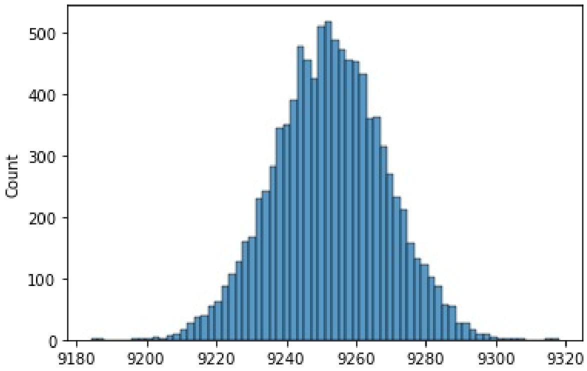
```
Age_means_survey = []
for rep in range(10000):
    Age_samples = np.random.choice(df_newage,size = 100000)
    Age_mean = np.mean(Age_samples)
    Age_means_survey.append(Age_mean)
```

In [359…

```
sns.histplot(Age_means_survey)
```

Out[359…

```
<AxesSubplot:ylabel='Count'>
```



In [360…

```
Ageupper = np.percentile(Age_means_survey, 97.5)
print(Ageupper)
Agelower = np.percentile(Age_means_survey, 2.5)
print(Agelower)
```

```
9283.960361
9221.11563325
```

In [ ]:

In [361…

```
df_newage = df_walmart["Purchase"][df_walmart["Age"] == "36-45"]
```

In [362…

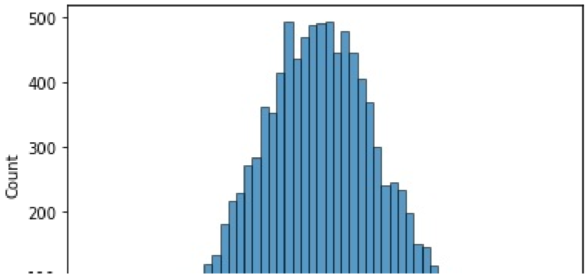
```
Age_means_survey = []
for rep in range(10000):
    Age_samples = np.random.choice(df_newage,size = 100000)
    Age_mean = np.mean(Age_samples)
    Age_means_survey.append(Age_mean)
```

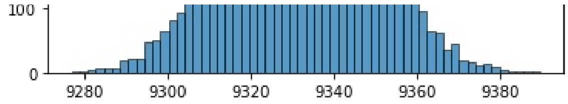
In [363…

```
sns.histplot(Age_means_survey)
```

Out[363…

```
<AxesSubplot:ylabel='Count'>
```





In [364…

```
Ageupper = np.percentile(Age_means_survey, 97.5)
print(Ageupper)
Agelower = np.percentile(Age_means_survey, 2.5)
print(Agelower)
```

```
9363.372332
9300.493819
```

In [ ]:

```

```

In [367…

```
df_newage = df_walmart["Purchase"][df_walmart["Age"] == "46-50"]
```

In [368…

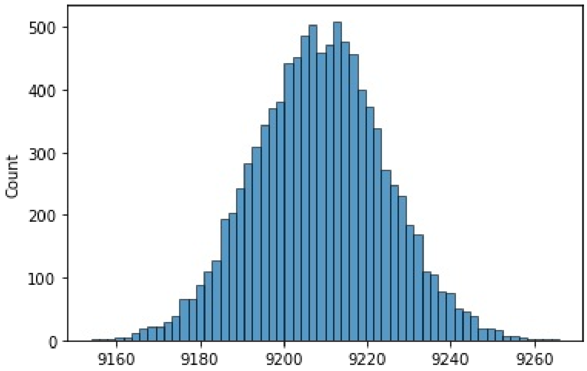
```
Age_means_survey = []
for rep in range(10000):
    Age_samples = np.random.choice(df_newage,size = 100000)
    Age_mean = np.mean(Age_samples)
    Age_means_survey.append(Age_mean)
```

In [369…

```
sns.histplot(Age_means_survey)
```

Out[369…

```
<AxesSubplot:ylabel='Count'>
```



In [370…

```
Ageupper = np.percentile(Age_means_survey, 97.5)
print(Ageupper)
Agelower = np.percentile(Age_means_survey, 2.5)
print(Agelower)
```

```
9239.71497275
9177.7491965
```

In [371…

```
df_newage = df_walmart["Purchase"][df_walmart["Age"] == "51-55"]
```

In [372…

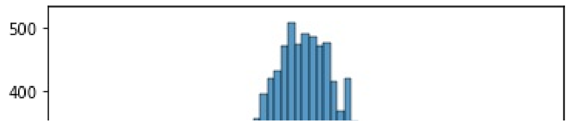
```
Age_means_survey = []
for rep in range(10000):
    Age_samples = np.random.choice(df_newage,size = 100000)
    Age_mean = np.mean(Age_samples)
    Age_means_survey.append(Age_mean)
```

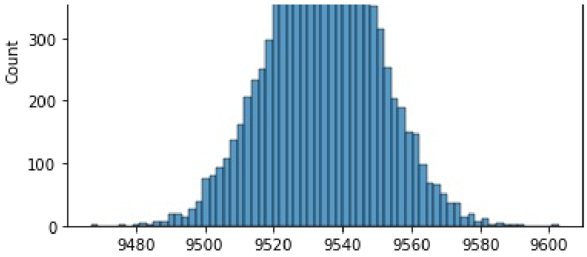
In [373…

```
sns.histplot(Age_means_survey)
```

Out[373…

```
<AxesSubplot:ylabel='Count'>
```





In [374…

```
Ageupper = np.percentile(Age_means_survey, 97.5)
print(Ageupper)
Agelower = np.percentile(Age_means_survey, 2.5)
print(Agelower)
```

```
9566.3904985
9502.24301575
```

In [375…

```
df_newage = df_walmart["Purchase"][df_walmart["Age"] == "55+"]
```

In [376…

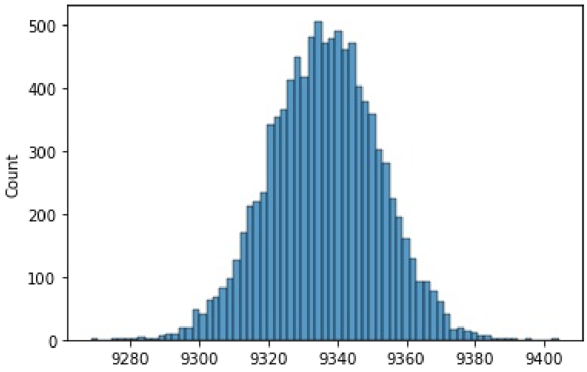
```
Age_means_survey = []
for rep in range(10000):
    Age_samples = np.random.choice(df_newage,size = 100000)
    Age_mean = np.mean(Age_samples)
    Age_means_survey.append(Age_mean)
```

In [377…

```
sns.histplot(Age_means_survey)
```

Out[377…

```
<AxesSubplot:ylabel='Count'>
```



In [378…

```
Ageupper = np.percentile(Age_means_survey, 97.5)
print(Ageupper)
Agelower = np.percentile(Age_means_survey, 2.5)
print(Agelower)
```

```
9367.094005750001
9304.559771749999
```

In [379…

```
df_walmart.head()
```

Out[379…

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category	Purchase
0	1000001	P00069042	F	0-17	10	A	2	0	3	8370
1	1000001	P00248942	F	0-17	10	A	2	0	1	15200
2	1000001	P00087842	F	0-17	10	A	2	0	12	1422
3	1000001	P00085442	F	0-17	10	A	2	0	12	1057
4	1000002	P00285442	M	55+	16	C	4+	0	8	7969

In [386…

```
df_walmart.corr()
```

Out[386…

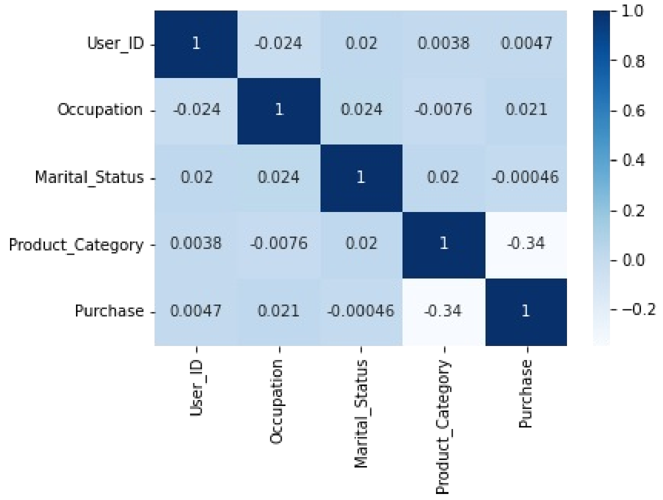
	User_ID	Occupation	Marital_Status	Product_Category	Purchase
User_ID	1.000000	-0.023971	0.020443	0.003825	0.004716
Occupation	-0.023971	1.000000	0.024280	-0.007618	0.020833
Marital_Status	0.020443	0.024280	1.000000	0.019888	-0.000463
Product_Category	0.003825	-0.007618	0.019888	1.000000	-0.343703
Purchase	0.004716	0.020833	-0.000463	-0.343703	1.000000

In [387…

```
sns.heatmap(data=df_walmart.corr(),annot=True,cmap="Blues")
plt.plot()
```

Out[387…

```
[]
```



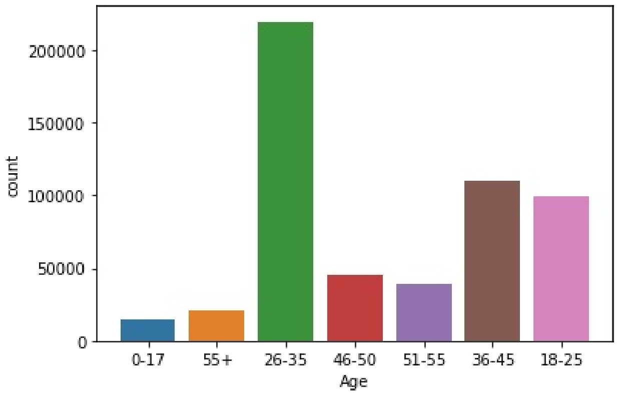
In [389…

```
sns.countplot(df_walmart["Age"])
plt.plot()
```

```
/Users/mrunmay/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[389…

```
[]
```



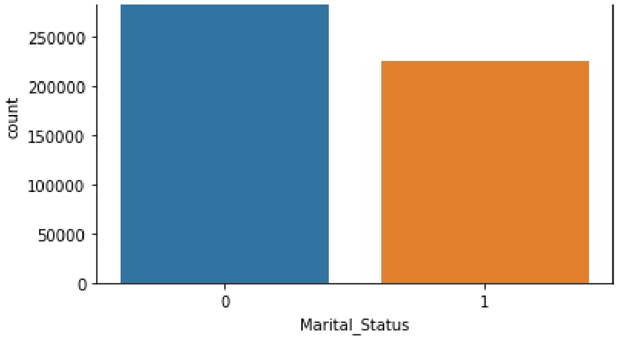
In [390…

```
sns.countplot(df_walmart["Marital_Status"])
plt.plot()
```

```
/Users/mrunmay/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[390…

```
[]
```

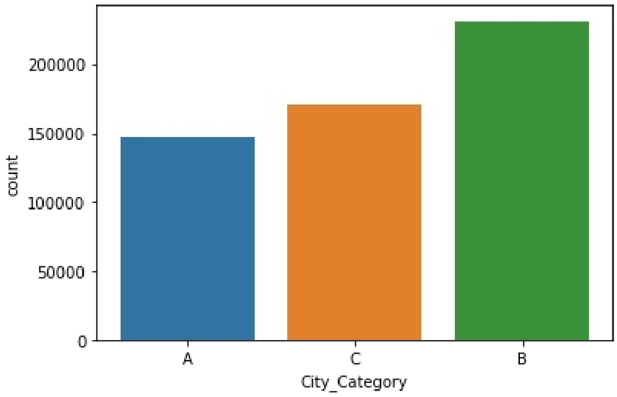


In [391…

```
sns.countplot(df_walmart["City_Category"])
plt.plot()
```

```
/Users/mrunmay/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the follo
wing variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and pass
ing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[391… []



In [392…

```
df_walmart.head()
```

Out[392…

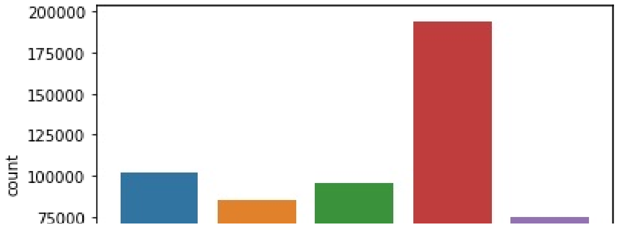
	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years	Marital_Status	Product_Category	Purchase
0	1000001	P00069042	F	0-17	10	A	2	0	3	8370
1	1000001	P00248942	F	0-17	10	A	2	0	1	15200
2	1000001	P00087842	F	0-17	10	A	2	0	12	1422
3	1000001	P00085442	F	0-17	10	A	2	0	12	1057
4	1000002	P00285442	M	55+	16	C	4+	0	8	7969

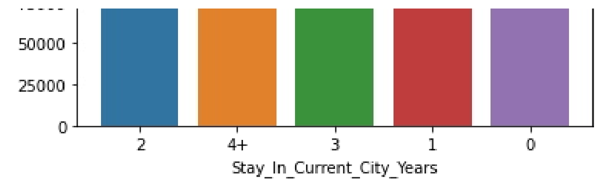
In [393…

```
sns.countplot(df_walmart["Stay_In_Current_City_Years"])
plt.plot()
```

```
/Users/mrunmay/opt/anaconda3/lib/python3.9/site-packages/seaborn/_decorators.py:36: FutureWarning: Pass the follo
wing variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and pass
ing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[393… []



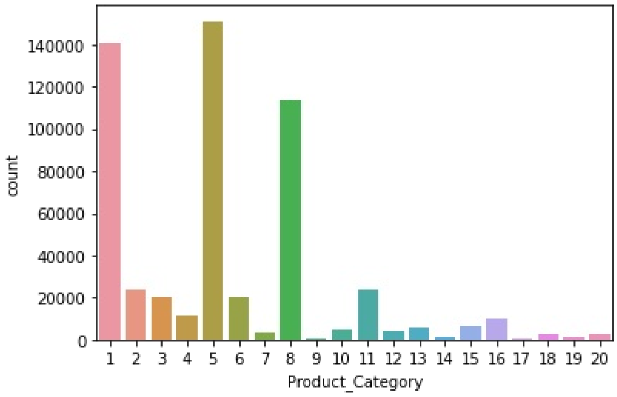


In [397…:

```
sns.countplot(data=df_walmart,x="Product_Category")
```

Out[397…:

```
<AxesSubplot:xlabel='Product_Category', ylabel='count'>
```



In [ ]:

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js