In [1]:

```
#"""Problem Statement: To analyse the customer purchase
#behavior on a Black Friday for Walmart"""
Recommendations:

1. Walmart can come up with more recommendations for the age group 26-35.
This is the age group when people have growth in career and personal life.
Generally,
the salaries of the age group between 26-35 increase at a faster rate and
they are willing to send that money for their needs and desires.
2. Recommend costlier products to male than female customers.
Generally, males look for more branded items.
The amount spent per transaction of all the 50 million female customers is less than the 50
3.  Singles tend to spend more money than married as unmarried have fewer commitments in li
4.  Recommend more products for city 'B' as the city maybe more developing
5. Recommend more products to people stay in current city 1 year as they have moved to new
will buy products for their new home
6. Age group 51-55 tend to buy products with average price [9321,9452].
Age []0-17] group tend to buy products with least average price.
Walmart can recommend the same set of products to these age groups 18-25,26-35,36-45, 46-50
7. Walmart needs to provide separate product recommendations for male and female as their c
8. Female tend to buy products with an average price [8506,8570]. Recommend the same to the
9. Walmart can recommend the same set of products to all irrespective of the  marital statu
10. Walmart can have a common recommendation for Stay_In_Current_City_Years 4+,3,2 and 1 fo
10. Walmart will need a separate recommendations for each city category as there is no over
```

In [2]:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
from  scipy import stats
```

In [3]:

```
df=pd.read_csv('Walmart_data.csv')
```

In [4]:

```
df.shape
#Shape of the data is 550068 rows and 10 columns
```

Out[4]:

```
(550068, 10)
```

In [5]:

```
df.isnull().sum()
# There is no null values in any of the columns
```

Out[5]:

```
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 [6]:

```
df.head()
```

Out[6]:

	User_ID	Product_ID	Gender	Age	Occupation	City_Category	Stay_In_Current_City_Years
0	1000001	P00069042	F	0-17	10	A	2
1	1000001	P00248942	F	0-17	10	A	2
2	1000001	P00087842	F	0-17	10	A	2
3	1000001	P00085442	F	0-17	10	A	2
4	1000002	P00285442	M	55+	16	C	4+

In [7]:

```
df.describe()
# Mean of purchase data is 9263.968
# STD of purchase data is 5023.065
# Median of purchase data is 8047.00
# Min and Max of purchase data are 12 and 23961.00
```

Out[7]:

	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 [8]:

```
len(df[df['Gender']=='M']),len(df[df['Gender']=='F'])
```

Out[8]:

```
(414259, 135809)
```

In [9]:

```
df.describe(include=object)
# Both male and female in age group between 26-35 buy the most products on a Black friday
```

Out[9]:

	Product_ID	Gender	Age	City_Category	Stay_In_Current_City_Years
count	550068	550068	550068	550068	550068
unique	3631	2	7	3	5
top	P00265242	M	26-35	B	1
freq	1880	414259	219587	231173	193821

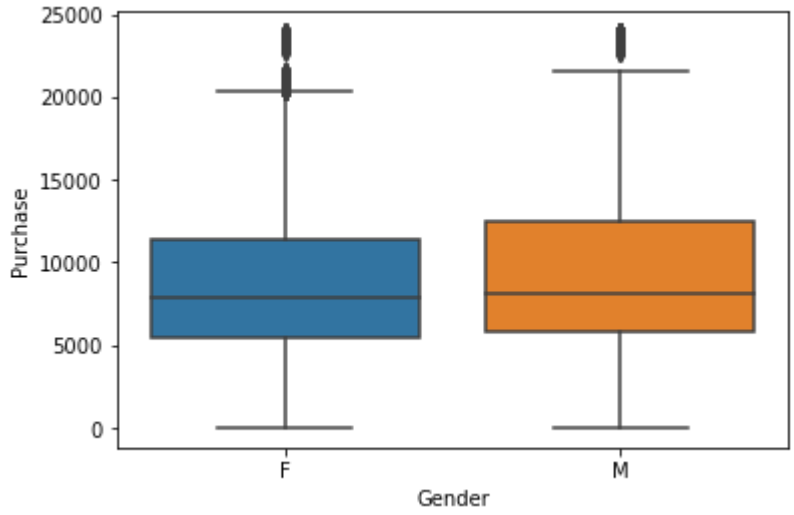
In [10]:

```
df.info()
# Columns 'User_ID','Occupation','Marital_Status','Product_Category' and 'Purchase' have it
```

```
<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
```

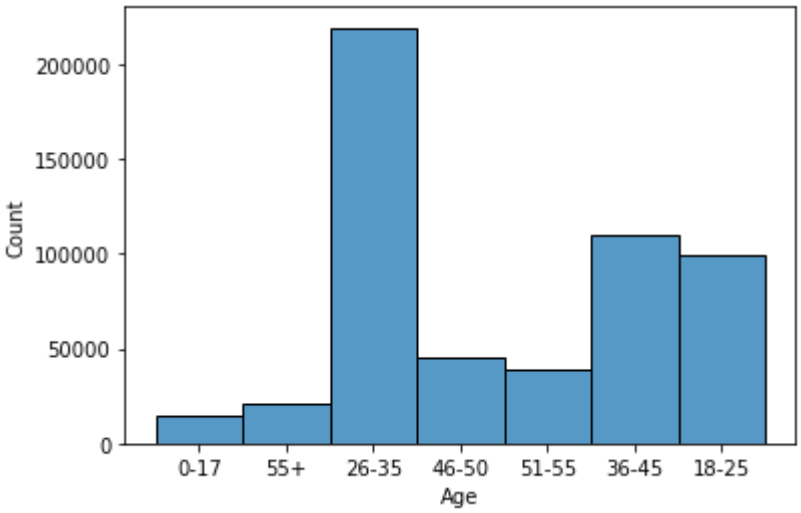
In [11]:

```
sns.boxplot(x=df['Gender'],y=df['Purchase'])
plt.show()
# Boxplot between gender and Purchase. Purchase column has some outliners that needs to be
#removed.
```



In [12]:

```
sns.histplot(x=df['Age'],multiple='dodge')
plt.show()
# Age group between 26-35, 36-45 and 18-25 buy more products on a Black Friday
```



In [13]:

```
def outliners(x,col):
    Q1 = np.percentile(x[col], 25)
    Q3 = np.percentile(x[col], 75)
    IQR = Q3 - Q1
    upper = Q3 +1.5*IQR
    lower = Q1 - 1.5*IQR
    #print(upper,lower)
    ls=list(x.iloc[((x[col]<lower) | (x[col]>upper)).values].index)
    return ls
```

In [14]:

```
df1=df[df['Gender']=='M']
df2=df[df['Gender']=='F']
df1.reset_index(inplace=True)
df1.drop(['index'],axis=1,inplace=True)
```

```
C:\Users\hp\anaconda3\lib\site-packages\pandas\core\frame.py:4906: SettingWi
thCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/
stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pand
as.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-v
ersus-a-copy)
  return super().drop(
```

In [15]:

```
ls1=outliners(df1,'Purchase')
ls2=outliners(df2,'Purchase')
```

In [16]:

```
(len(ls1)/len(df1))*100, (len(ls2)/len(df2))*100
# 0.4374 % of Male are outliers and they are removed
# 1.5205 % of Female are outliers and they are removed
```

Out[16]:

```
(0.4374075155880741, 1.5205177860082912)
```

In [17]:

```
df1.drop(ls1,axis=0,inplace=True)
df2.drop(ls2,axis=0,inplace=True)
```

In [18]:

```
dff=pd.concat([df1, df2], axis=0,ignore_index=True)
```

In [19]:

```
len(dff)
```

Out[19]:

```
546191
```

In [20]:

```
((len(df)-len(dff))/len(df))*100
```

Out[20]:

```
0.7048219492862701
```

In [21]:

```
dff.groupby('Marital_Status')[['Purchase']].aggregate({'Purchase':['min','max','mean','medi
# Median of purchase amount of partenered and single are 8038 and 8031 respectively for the
```

Out[21]:

	Purchase			
	min	max	mean	median
Marital_Status				
0	12	21568	9178.030493	8031.0
1	12	21569	9160.388650	8038.0

In [22]:

```
dff.groupby('Age')[['Purchase']].aggregate({'Purchase':['min','max','mean','median']})
# Median purchase amount for respective age group is shown below for the sample data.
#Lowest median of purchase is from the age group '0-17'
```

Out[22]:

	Purchase			
	min	max	mean	median
Age				
0-17	12	21567	8846.141296	7970.5
18-25	12	21567	9108.206666	8017.0
26-35	12	21568	9169.190790	8018.0
36-45	12	21569	9226.662898	8047.0
46-50	12	21561	9100.615103	8021.0
51-55	12	21563	9387.134312	8111.0
55+	12	21555	9184.171770	8086.0

In [23]:

```
dff.groupby('Gender')[['Purchase']].aggregate({'Purchase':['min','max','mean','median']})
# Median purchase amount for respective gender is shown below for the sample data.
#Lowest median of purchase is from Female
```

Out[23]:

	Purchase			
	min	max	mean	median
Gender				
F	12	20350	8538.524637	7886.0
M	12	21569	9375.837603	8090.0

In [24]:

```
for i in dff:
    print('Unique values in',i,'column are',dff[i].unique())
# Below are the unique values in each column
```

```
Unique values in User_ID column are [1000002 1000003 1000004 ... 1000703 100
4293 1004588]
Unique values in Product_ID column are ['P00285442' 'P00193542' 'P00184942'
... 'P00038842' 'P00350742'
 'P00060842']
Unique values in Gender column are ['M' 'F']
Unique values in Age column are ['55+' '26-35' '46-50' '36-45' '51-55' '0-1
7' '18-25']
Unique values in Occupation column are [16 15  7 20  1 12 17  0 10  4 11  3
8 19  2 18 14 13  9  6  5]
Unique values in City_Category column are ['C' 'A' 'B']
Unique values in Stay_In_Current_City_Years column are ['4+' '3' '2' '1'
'0']
Unique values in Marital_Status column are [0 1]
Unique values in Product_Category column are [ 8  1  5  6  3 11  2 13 15  4
7 16 18 12 10 14 17  9 20 19]
Unique values in Purchase column are [ 7969 15227 19215 ...  3365  6701 1838
5]
```

In [25]:

```
for i in dff:
    print('Value counts in',i,'column are')
    print(dff[i].value_counts())
# Below is the count of the unique values in each colum
```

```
Value counts in User_ID column are
1001680    1024
1004277     975
1001941     898
1001181     861
1000889     817
           ...
1004991       7
1002690       7
1002111       7
1005810       7
1000708       6
Name: User_ID, Length: 5891, dtype: int64
Value counts in Product_ID column are
P00265242    1880
P00025442    1615
P00110742    1612
P00112142    1562
P00057642    1470
             ...
P00348142       1
P00064542       1
P00069742       1
P00056542       1
P00060842       1
Name: Product_ID, Length: 3631, dtype: int64
Value counts in Gender column are
M    412447
F    133744
Name: Gender, dtype: int64
Value counts in Age column are
26-35    218193
36-45    109142
18-25     99194
46-50     45329
51-55     38068
55+       21261
0-17      15004
Name: Age, dtype: int64
Value counts in Occupation column are
4     71898
0     69117
7     58783
1     47013
17    39821
20    33291
12    30974
14    27132
2     26360
16    25220
6     20156
3     17478
10    12866
```

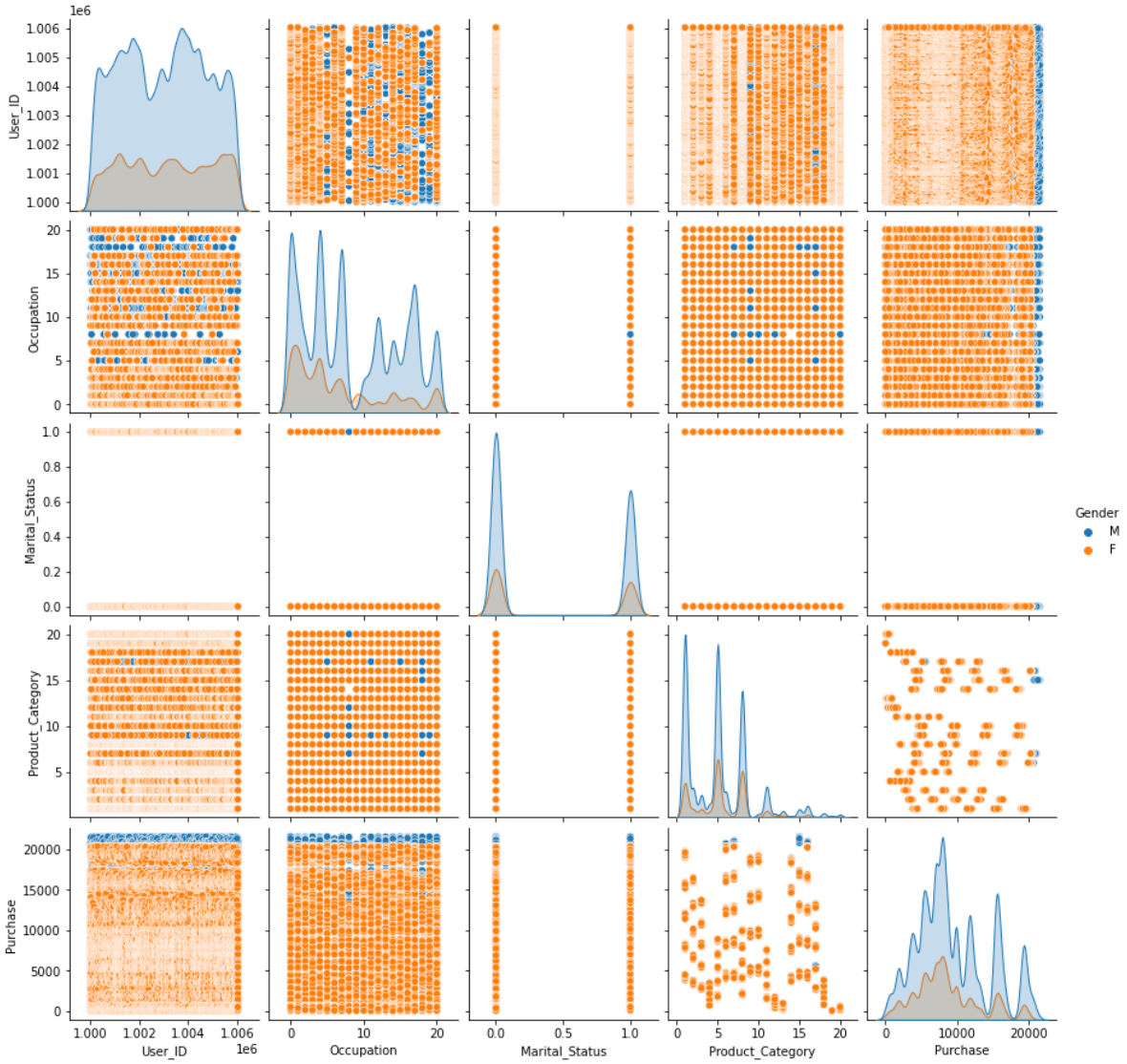
```
5     12117
15    12054
11    11488
19     8395
13     7658
18     6597
9      6233
8      1540
Name: Occupation, dtype: int64
Value counts in City_Category column are
B    229663
C    169812
A    146716
Name: City_Category, dtype: int64
Value counts in Stay_In_Current_City_Years column are
1     192360
2     101192
3      94583
4+     84162
0      73894
Name: Stay_In_Current_City_Years, dtype: int64
Value counts in Marital_Status column are
0    322568
1    223623
Name: Marital_Status, dtype: int64
Value counts in Product_Category column are
5     150933
1     140378
8     113925
11     24287
2      23864
3      20213
6      19893
4      11753
16      9447
15      6008
13      5549
12      3947
7       3430
18      3125
10      2850
20      2550
19      1603
14      1523
17       578
9        335
Name: Product_Category, dtype: int64
Value counts in Purchase column are
7011     191
7193     188
6855     187
6891     184
6960     183
        ...
855        1
3338       1
9392       1
18506      1
18385      1
Name: Purchase, Length: 17193, dtype: int64
```

In [26]:

```
sns.pairplot(data=dff,hue='Gender')
#Below is the pair plot having hue as gender
# We cannot conclude wit any relation as the count of male sample are more than female
```

Out[26]:

```
<seaborn.axisgrid.PairGrid at 0x1a6a690ba60>
```

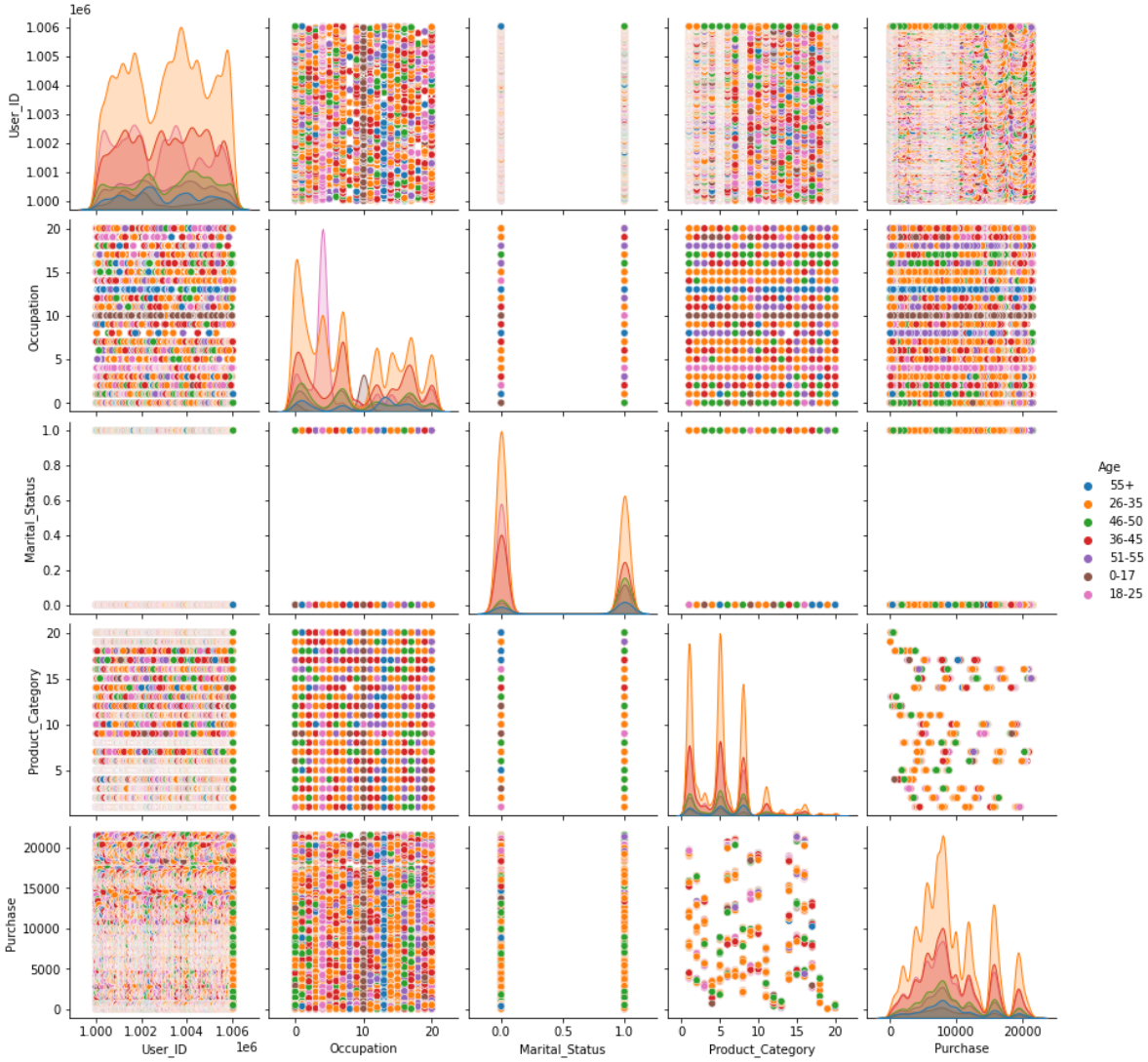


In [27]:

```
sns.pairplot(data=dff,hue='Age')
#Below is the pair plot having hue as Age
# Age group 26-35,36-45 have the highest purchase and other age groups purchase amount are
```

Out[27]:

```
<seaborn.axisgrid.PairGrid at 0x1a6a7e909a0>
```

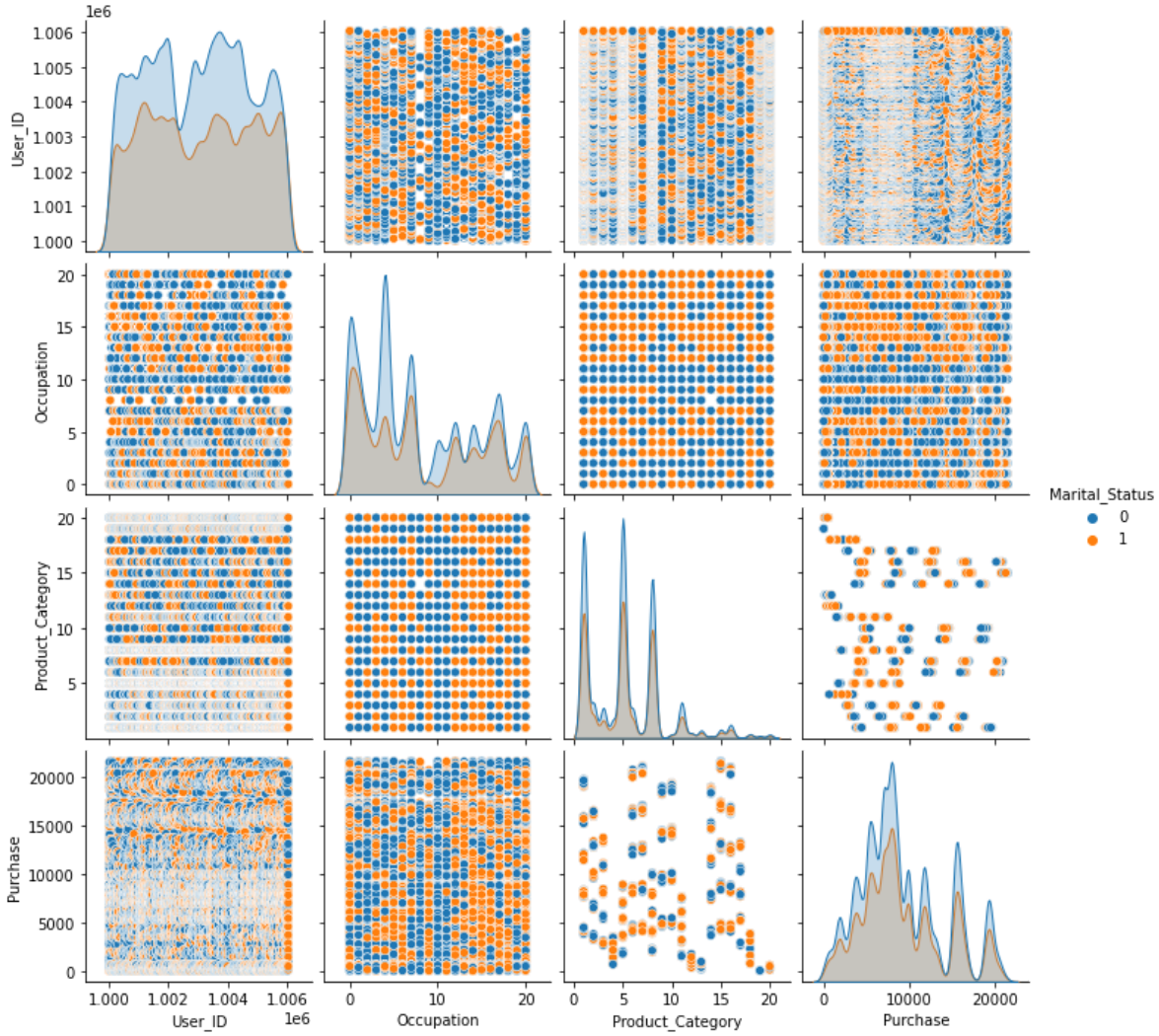


In [28]:

```
sns.pairplot(data=dff,hue='Marital_Status')
# Marital status 0 tend to purchase more on a black friday sale.
# Purchase amount of marital status 1 lie with in marital status 0 purchase
```

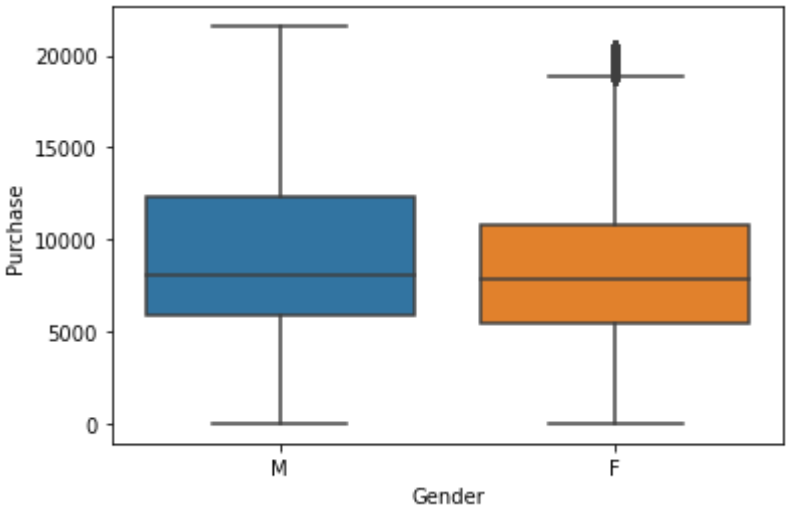
Out[28]:

```
<seaborn.axisgrid.PairGrid at 0x1a6a6cc3d30>
```



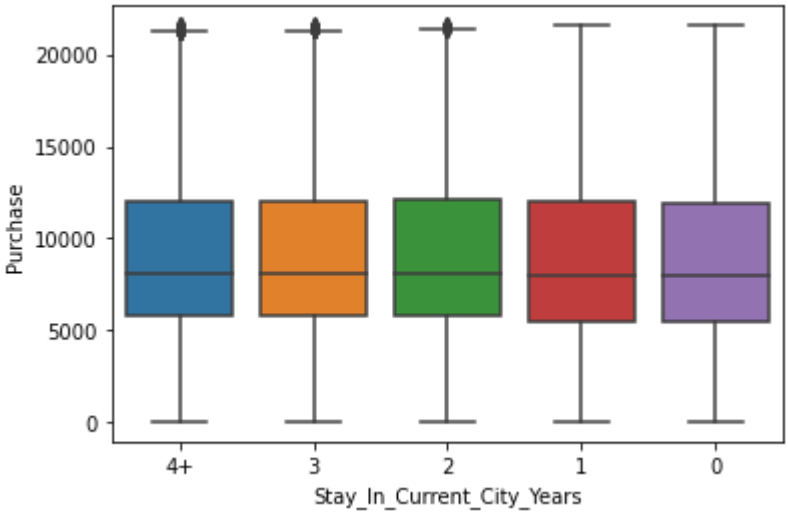
In [29]:

```
sns.boxplot(x=dff['Gender'],y=dff['Purchase'])
plt.show()
# Below barplot is between Gender and purchase.
# Median of M and F are same for the sample data
```



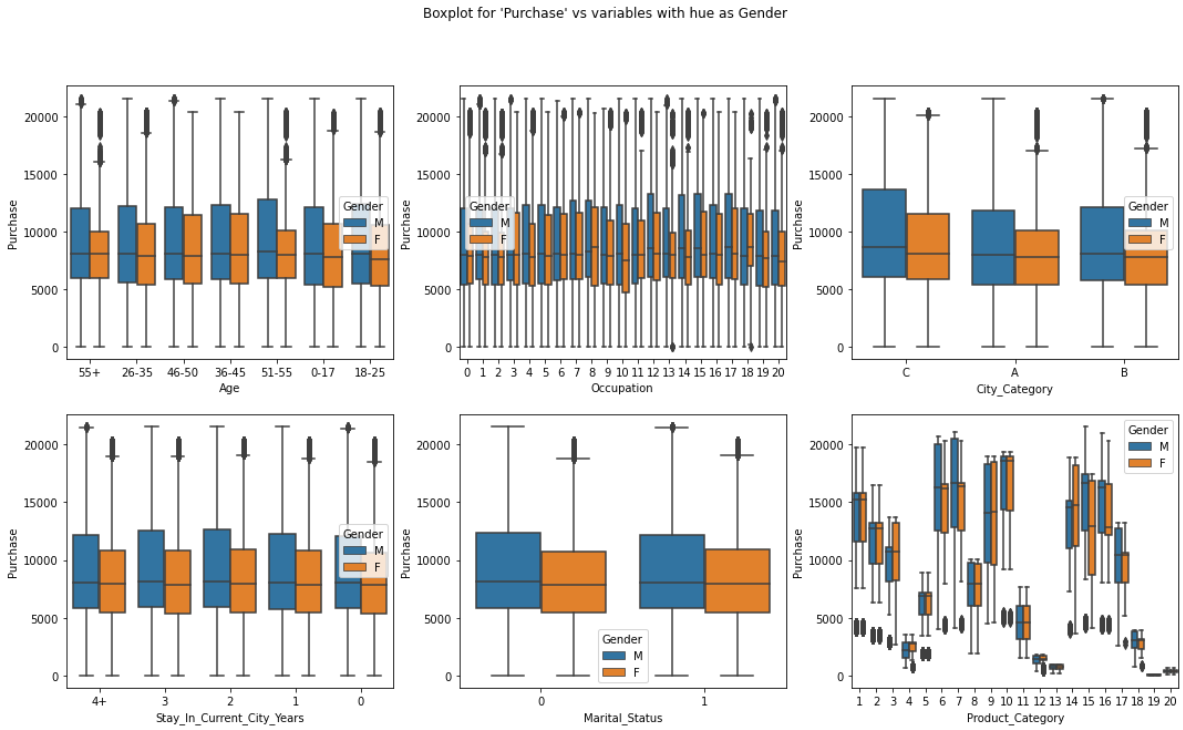
In [30]:

```
sns.boxplot(x=dff['Stay_In_Current_City_Years'],y=dff['Purchase'])
plt.show()
#Box plot for Stay_in_currect_city_years vs Purchase
# Median remains same for all the years
```



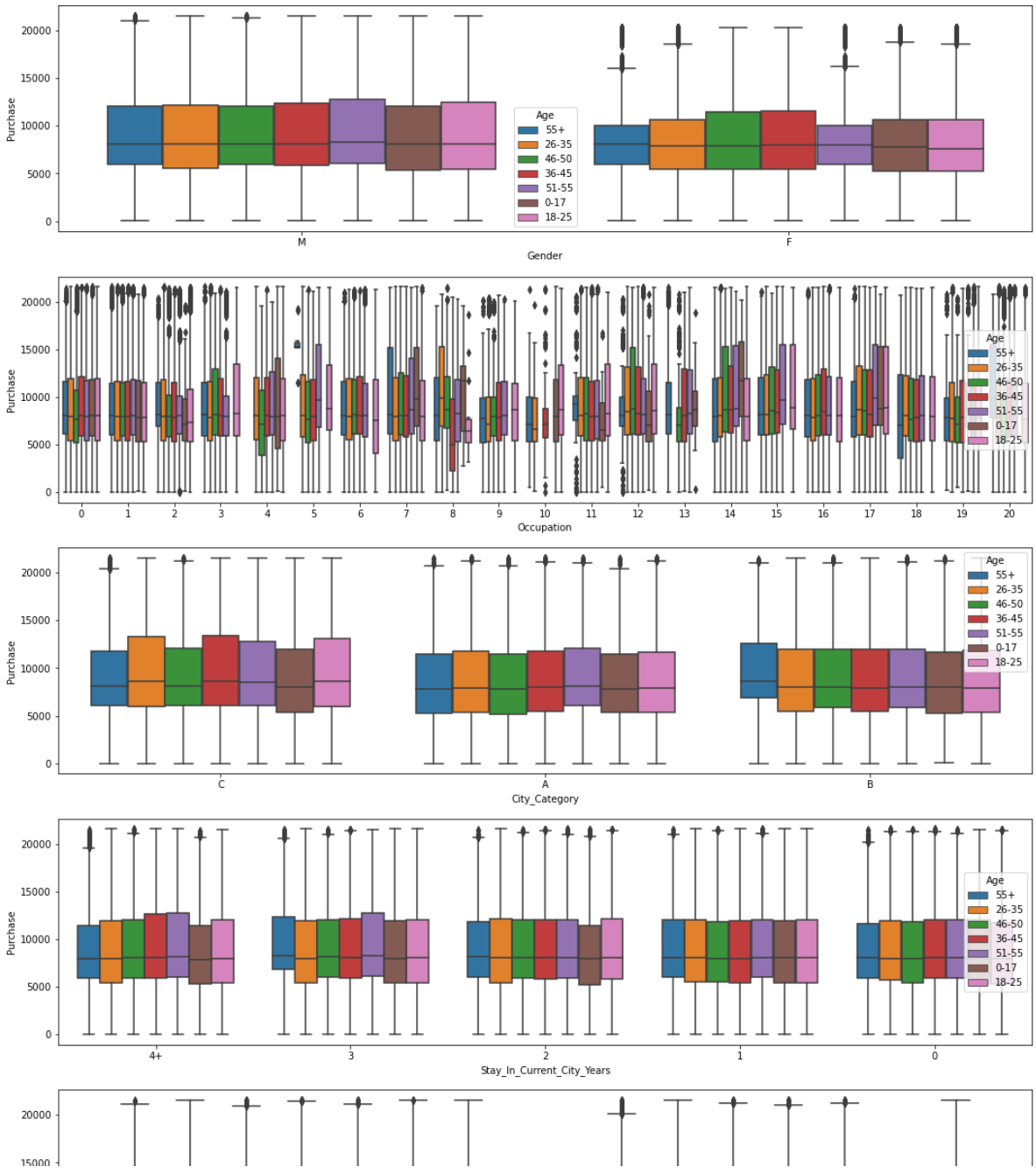
In [31]:

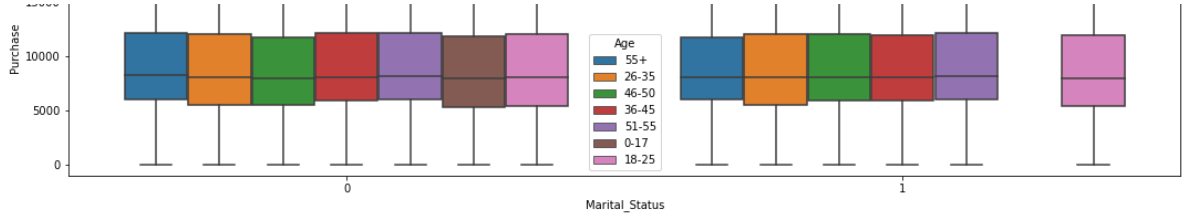
```
fig, axes = plt.subplots(2, 3, figsize=(18, 10))

fig.suptitle('Boxplot for \'Purchase\' vs variables with hue as Gender')
sns.boxplot(ax=axes[0, 0], data=dff, x='Age', y='Purchase',hue='Gender' )
sns.boxplot(ax=axes[0, 1], data=dff, x='Occupation', y='Purchase',hue='Gender')
sns.boxplot(ax=axes[0, 2], data=dff, x='City_Category', y='Purchase',hue='Gender')
sns.boxplot(ax=axes[1, 0], data=dff, x='Stay_In_Current_City_Years', y='Purchase',hue='Gend
sns.boxplot(ax=axes[1, 1], data=dff, x='Marital_Status', y='Purchase',hue='Gender')
sns.boxplot(ax=axes[1, 2], data=dff, x='Product_Category', y='Purchase',hue='Gender')
plt.show()
# Box plot shows all parameters vs purchase amount with hue as gender
```

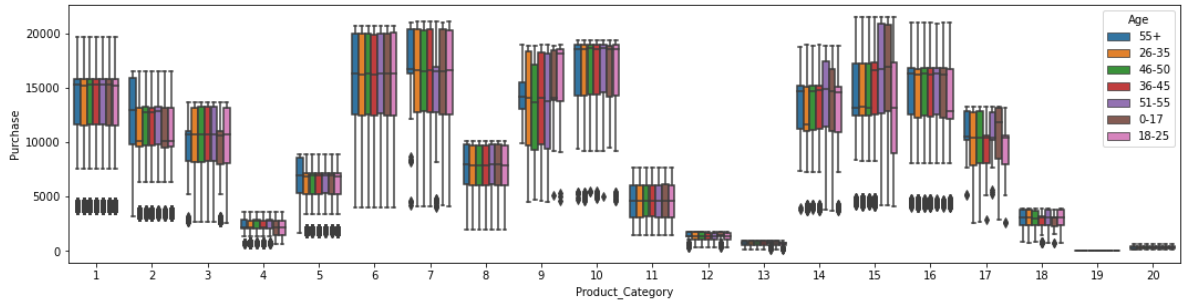


In [32]:

```
fig, axes = plt.subplots(6, 1, figsize=(18, 30))

fig.suptitle('Boxplot for \'Purchase\' vs variables with hue as Age')
sns.boxplot(ax=axes[0], data=dff, x='Gender', y='Purchase',hue='Age' )
sns.boxplot(ax=axes[1], data=dff, x='Occupation', y='Purchase',hue='Age')
sns.boxplot(ax=axes[2], data=dff, x='City_Category', y='Purchase',hue='Age')
sns.boxplot(ax=axes[3], data=dff, x='Stay_In_Current_City_Years', y='Purchase',hue='Age')
sns.boxplot(ax=axes[4], data=dff, x='Marital_Status', y='Purchase',hue='Age')
sns.boxplot(ax=axes[5], data=dff, x='Product_Category', y='Purchase',hue='Age')
plt.show()
# Box plot shows all parameters vs purchase amount with hue as age
```

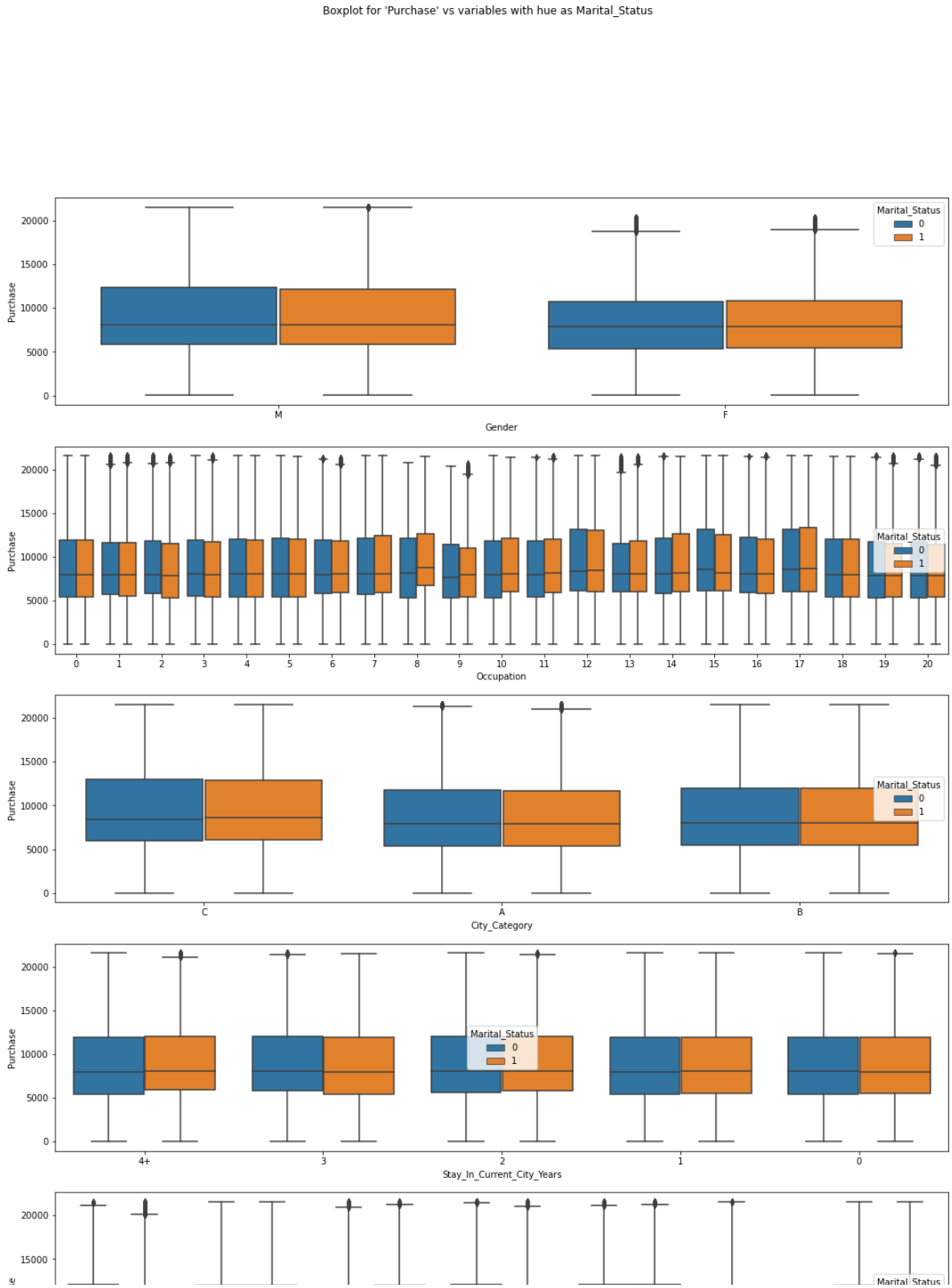


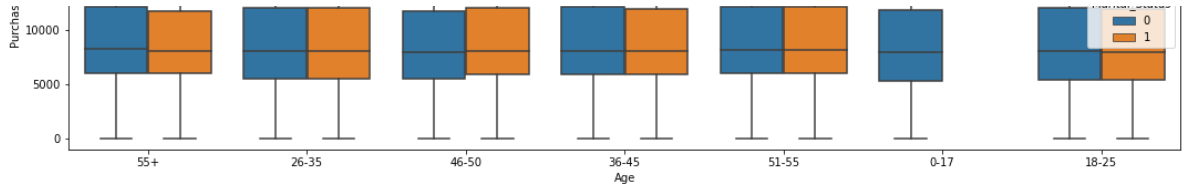

Purchase
10000
5000
0
Age
55+
26-35
46-50
36-45
51-55
0-17
18-25
0
1
Marital_Status

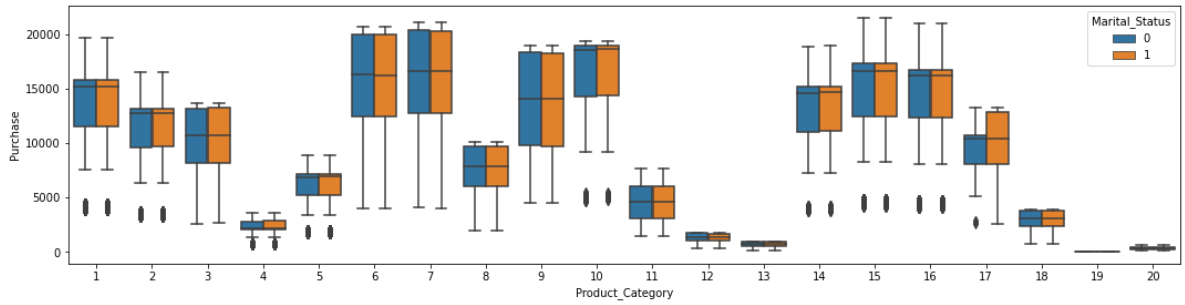

Purchase
20000
15000
10000
5000
0
Age
55+
26-35
46-50
36-45
51-55
0-17
18-25
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
Product_Category

In [33]:

```
fig, axes = plt.subplots(6, 1, figsize=(18, 30))

fig.suptitle('Boxplot for \'Purchase\' vs variables with hue as Marital_Status')
sns.boxplot(ax=axes[0], data=dff, x='Gender', y='Purchase',hue='Marital_Status' )
sns.boxplot(ax=axes[1], data=dff, x='Occupation', y='Purchase',hue='Marital_Status')
sns.boxplot(ax=axes[2], data=dff, x='City_Category', y='Purchase',hue='Marital_Status')
sns.boxplot(ax=axes[3], data=dff, x='Stay_In_Current_City_Years', y='Purchase',hue='Marital
sns.boxplot(ax=axes[4], data=dff, x='Age', y='Purchase',hue='Marital_Status')
sns.boxplot(ax=axes[5], data=dff, x='Product_Category', y='Purchase',hue='Marital_Status')
plt.show()
# Box plot shows all parameters vs purchase amount with hue as marital status
```

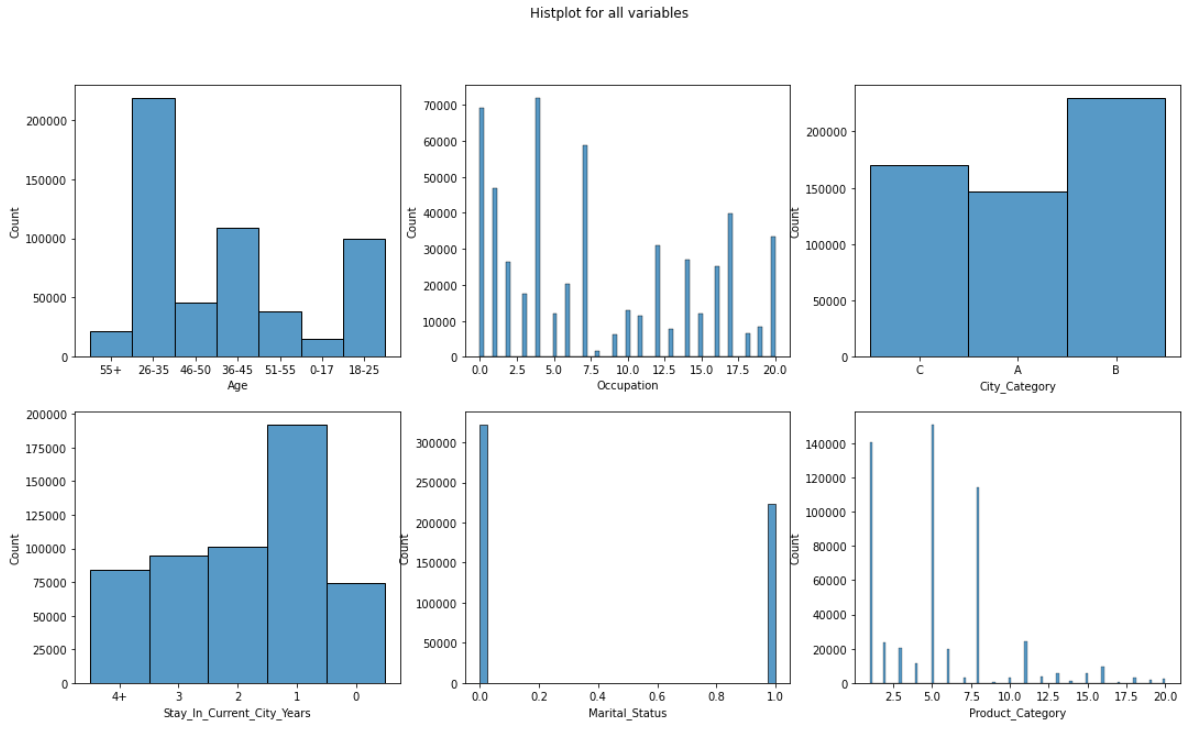






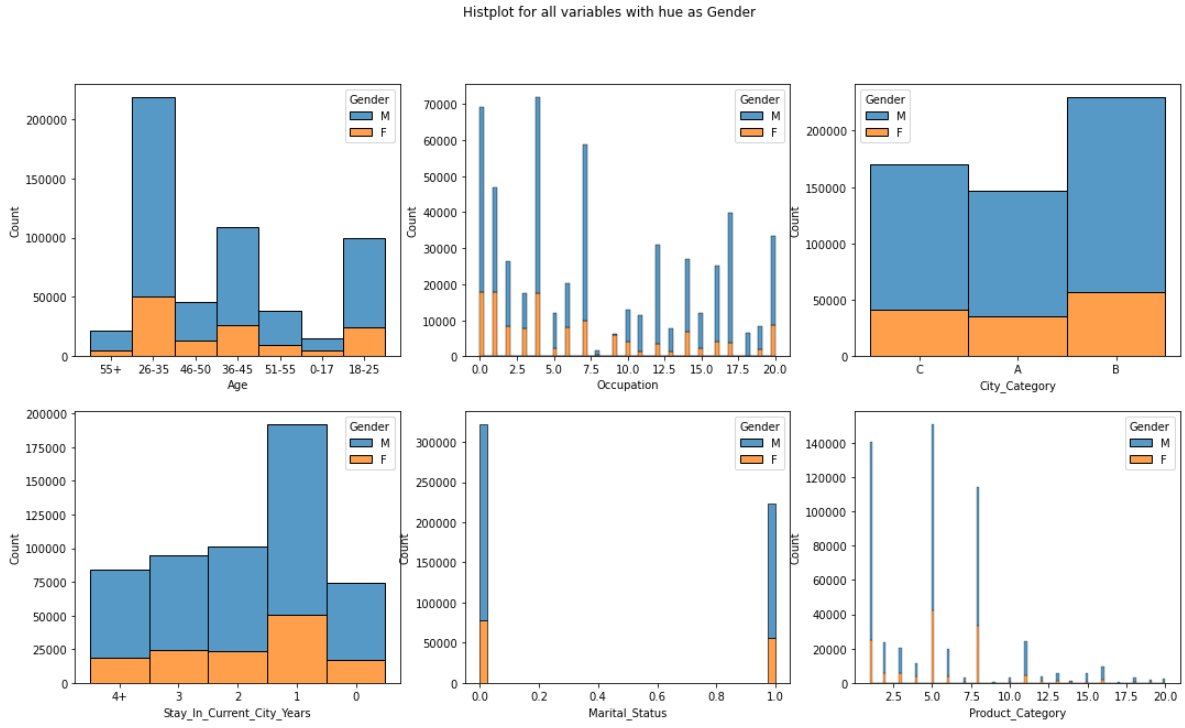
In [34]:

```
fig, axes = plt.subplots(2, 3, figsize=(18, 10))

fig.suptitle('Histplot for all variables')
sns.histplot(ax=axes[0, 0], data=dff, x='Age')
sns.histplot(ax=axes[0, 1], data=dff, x='Occupation')
sns.histplot(ax=axes[0, 2], data=dff, x='City_Category')
sns.histplot(ax=axes[1, 0], data=dff, x='Stay_In_Current_City_Years')
sns.histplot(ax=axes[1, 1], data=dff, x='Marital_Status')
sns.histplot(ax=axes[1, 2], data=dff, x='Product_Category')
plt.show()
# Hist plot shows all parameters vs count
```



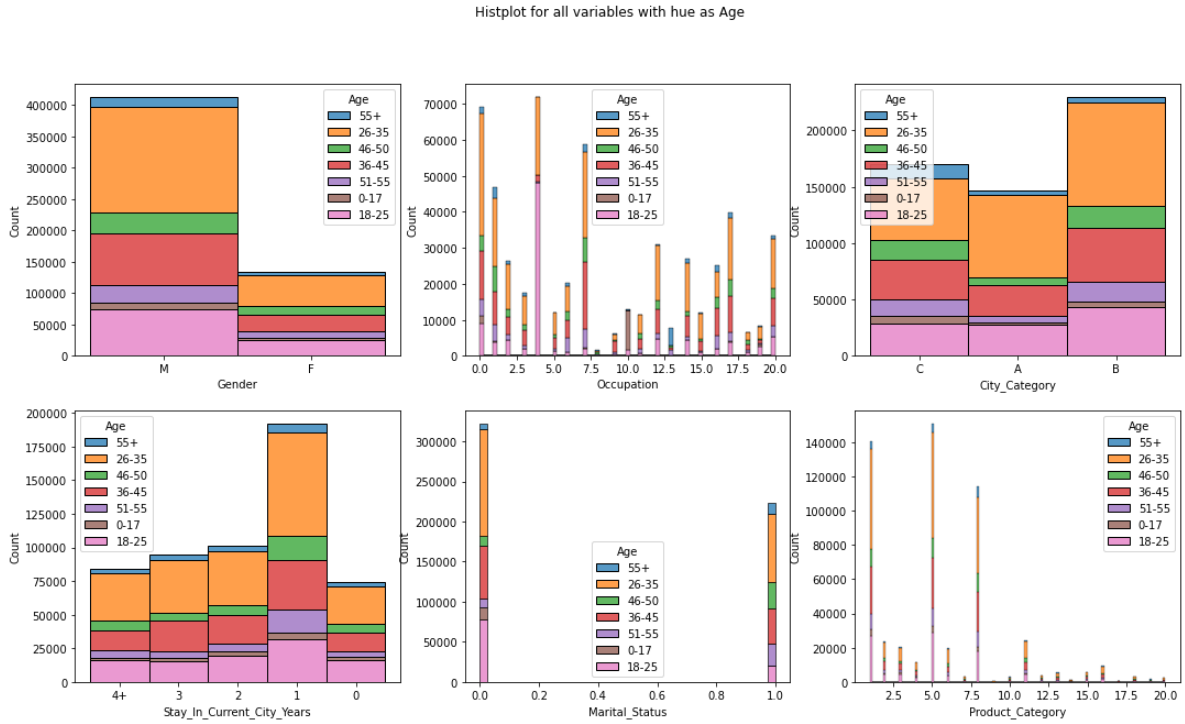
In [35]:

```
fig, axes = plt.subplots(2, 3, figsize=(18, 10))

fig.suptitle('Histplot for all variables with hue as Gender')
sns.histplot(ax=axes[0, 0], data=dff, x='Age',hue='Gender',multiple='stack')
plt.grid()
sns.histplot(ax=axes[0, 1], data=dff, x='Occupation', hue='Gender',multiple='stack')
plt.grid()
sns.histplot(ax=axes[0, 2], data=dff, x='City_Category',hue='Gender',multiple='stack')
plt.grid()
sns.histplot(ax=axes[1, 0], data=dff, x='Stay_In_Current_City_Years',hue='Gender',multiple=
plt.grid()
sns.histplot(ax=axes[1, 1], data=dff, x='Marital_Status',hue='Gender',multiple='stack')
plt.grid()
sns.histplot(ax=axes[1, 2], data=dff, x='Product_Category',hue='Gender',multiple='stack')
plt.grid()
plt.show()
# Hist plot shows all parameters vs count with hue as gender
```



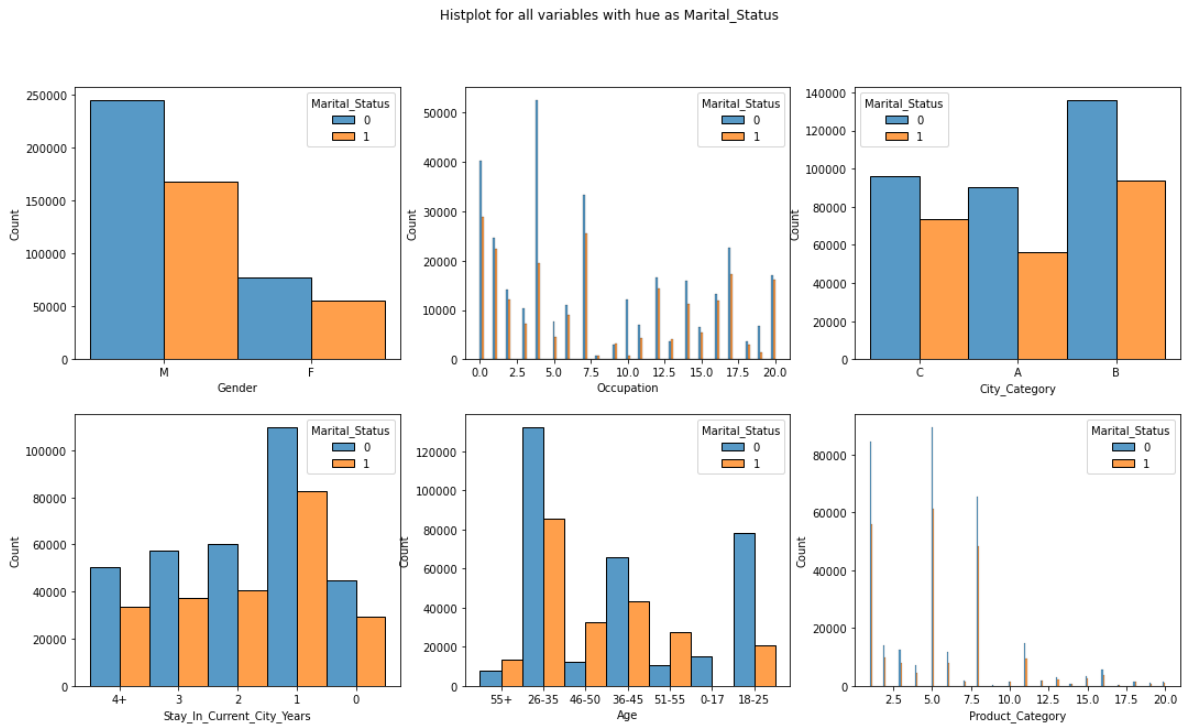
In [36]:

```
fig, axes = plt.subplots(2, 3, figsize=(18, 10))

fig.suptitle('Histplot for all variables with hue as Age')
sns.histplot(ax=axes[0, 0], data=dff, x='Gender',hue='Age',multiple='stack')
plt.grid()
sns.histplot(ax=axes[0, 1], data=dff, x='Occupation', hue='Age',multiple='stack')
plt.grid()
sns.histplot(ax=axes[0, 2], data=dff, x='City_Category',hue='Age',multiple='stack')
plt.grid()
sns.histplot(ax=axes[1, 0], data=dff, x='Stay_In_Current_City_Years',hue='Age',multiple='st
plt.grid()
sns.histplot(ax=axes[1, 1], data=dff, x='Marital_Status',hue='Age',multiple='stack')
plt.grid()
sns.histplot(ax=axes[1, 2], data=dff, x='Product_Category',hue='Age',multiple='stack')
plt.grid()
plt.show()
# Hist plot shows all parameters vs count with hue as age
```



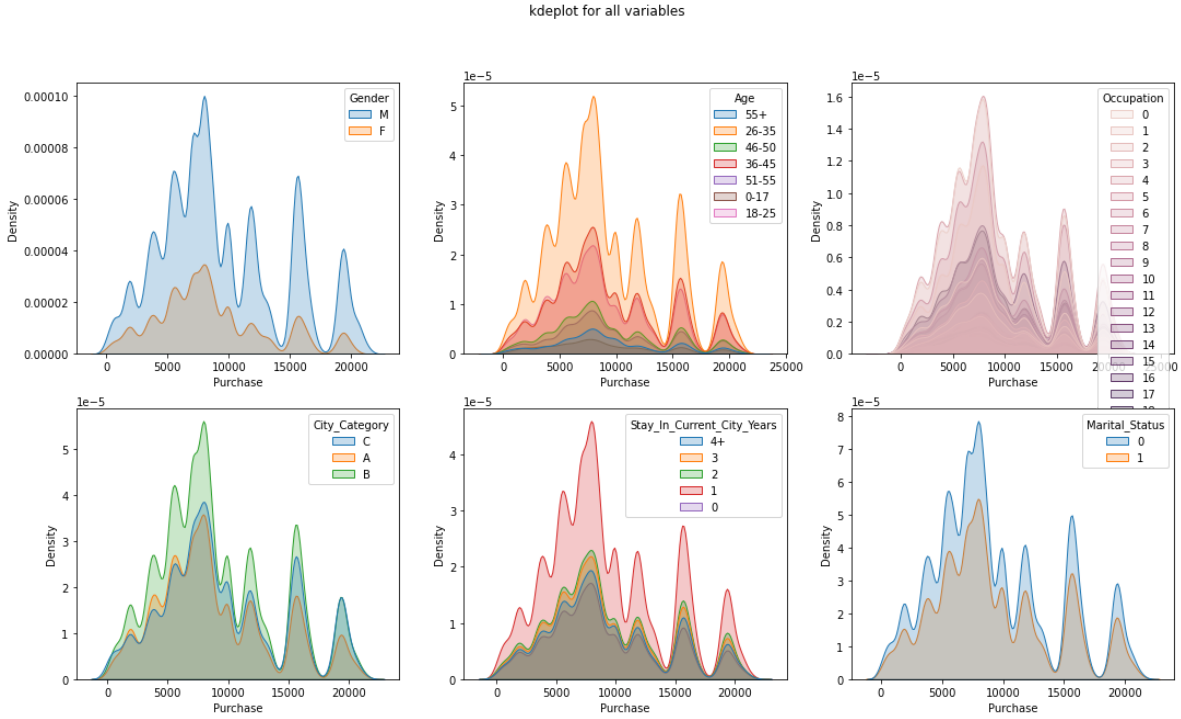
In [37]:

```
fig, axes = plt.subplots(2, 3, figsize=(18, 10))

fig.suptitle('Histplot for all variables with hue as Marital_Status')
sns.histplot(ax=axes[0, 0], data=dff, x='Gender',hue='Marital_Status',multiple='dodge')
plt.grid()
sns.histplot(ax=axes[0, 1], data=dff, x='Occupation', hue='Marital_Status',multiple='dodge'
plt.grid()
sns.histplot(ax=axes[0, 2], data=dff, x='City_Category',hue='Marital_Status',multiple='dodg
plt.grid()
sns.histplot(ax=axes[1, 0], data=dff, x='Stay_In_Current_City_Years',hue='Marital_Status',m
plt.grid()
sns.histplot(ax=axes[1, 1], data=dff, x='Age',hue='Marital_Status',multiple='dodge')
plt.grid()
sns.histplot(ax=axes[1, 2], data=dff, x='Product_Category',hue='Marital_Status',multiple='d
plt.grid()
plt.show()
# Hist plot shows all parameters vs count with hue as marital status
```



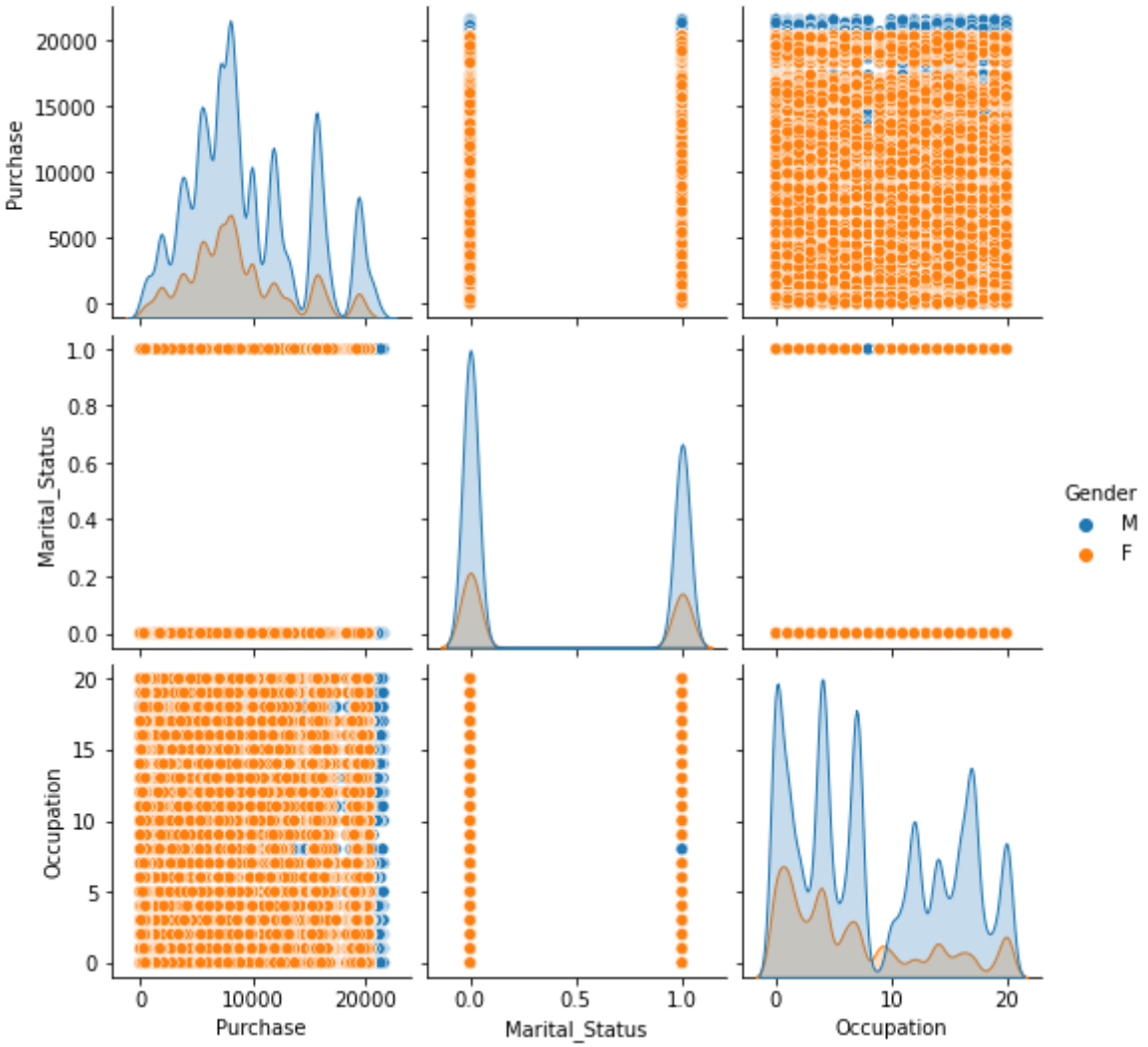
In [38]:

```
fig, axes = plt.subplots(2, 3, figsize=(18, 10))

fig.suptitle('kdeplot for all variables')
sns.kdeplot(ax=axes[0, 0], data=dff, x='Purchase',hue='Gender',shade=True)
sns.kdeplot(ax=axes[0, 1], data=dff, x='Purchase',hue='Age',shade=True)
sns.kdeplot(ax=axes[0, 2], data=dff, x='Purchase',hue='Occupation',shade=True)
sns.kdeplot(ax=axes[1, 0], data=dff, x='Purchase',hue='City_Category',shade=True)
sns.kdeplot(ax=axes[1, 1], data=dff, x='Purchase',hue='Stay_In_Current_City_Years',shade=Tr
sns.kdeplot(ax=axes[1, 2], data=dff, x='Purchase',hue='Marital_Status',shade=True)
plt.show()
# kde plot for all variables
```



In [39]:

```
sns.pairplot(dff, vars=['Purchase', 'Marital_Status','Occupation'], hue = "Gender")
plt.show()
# Below is the pair plot between 'Purchase','Marital_Status' and 'Occupation'
```



In [40]:

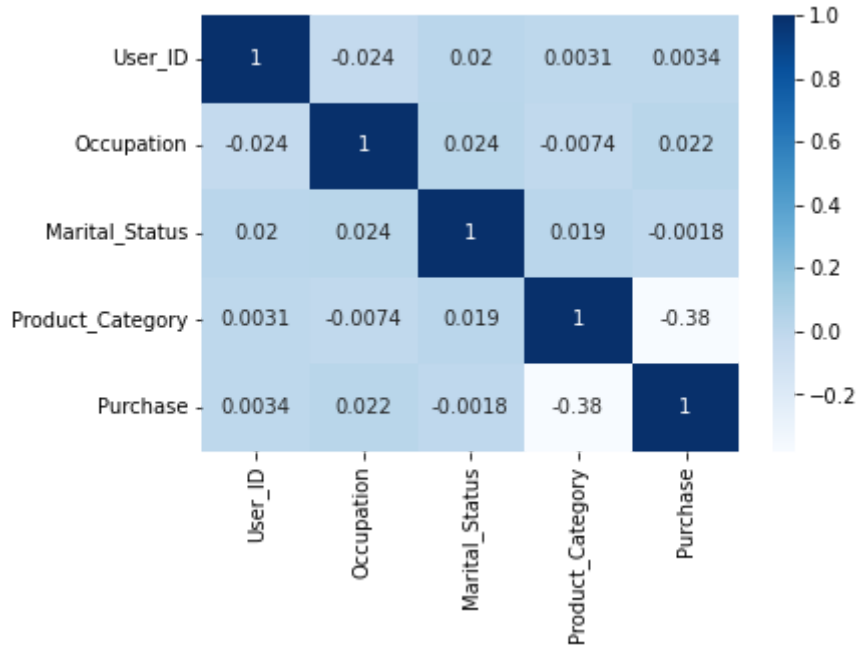
```
dff.corr()
# There is no correlation between integer vaariables
```

Out[40]:

	User_ID	Occupation	Marital_Status	Product_Category	Purchase
User_ID	1.000000	-0.024086	0.020097	0.003097	0.003360
Occupation	-0.024086	1.000000	0.024070	-0.007401	0.022131
Marital_Status	0.020097	0.024070	1.000000	0.019487	-0.001765
Product_Category	0.003097	-0.007401	0.019487	1.000000	-0.377867
Purchase	0.003360	0.022131	-0.001765	-0.377867	1.000000

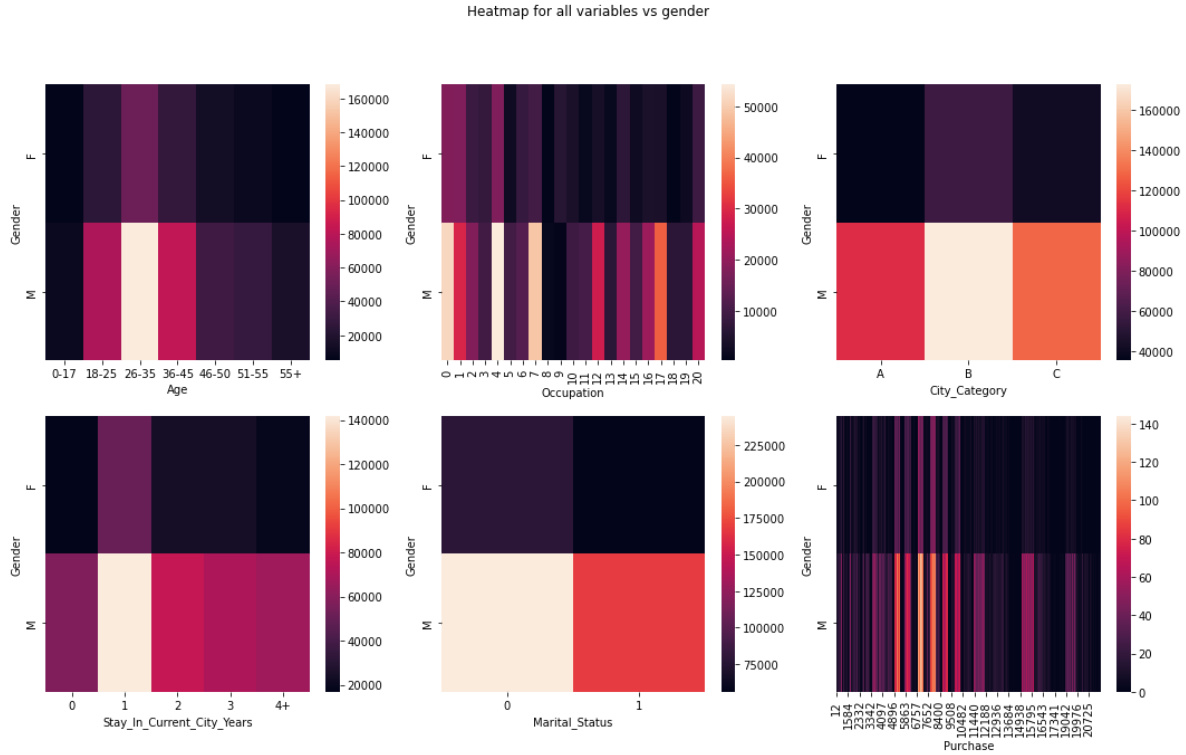
In [41]:

```
sns.heatmap(dff.corr(),cmap='Blues',annot=True)
plt.show()
# Heat map for the correlation for all integer variables
```



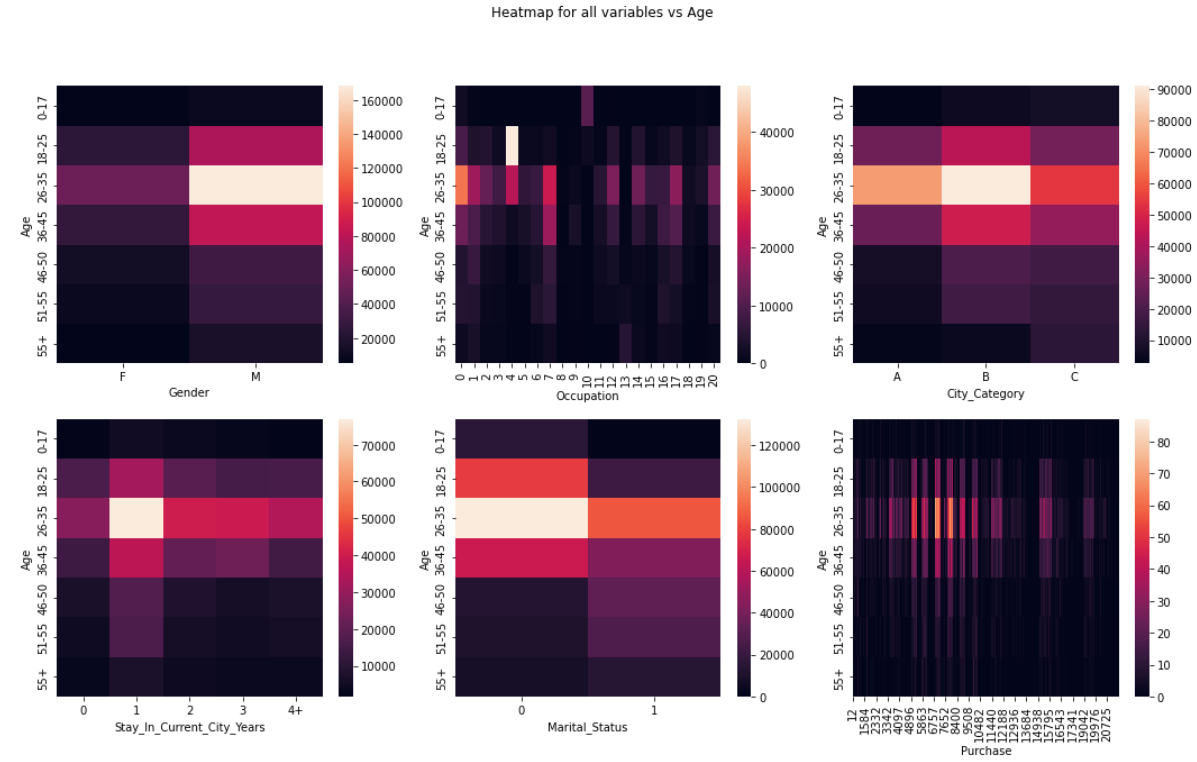
In [42]:

```
fig, axes = plt.subplots(2, 3, figsize=(18, 10))

fig.suptitle('Heatmap for all variables vs gender')
sns.heatmap(pd.crosstab(dff['Gender'],dff['Age']),ax=axes[0, 0])
sns.heatmap(pd.crosstab(dff['Gender'],dff['Occupation']),ax=axes[0, 1])
sns.heatmap(pd.crosstab(dff['Gender'],dff['City_Category']),ax=axes[0, 2])
sns.heatmap(pd.crosstab(dff['Gender'],dff['Stay_In_Current_City_Years']),ax=axes[1, 0])
sns.heatmap(pd.crosstab(dff['Gender'],dff['Marital_Status']),ax=axes[1, 1])
sns.heatmap(pd.crosstab(dff['Gender'],dff['Purchase']),ax=axes[1, 2])
plt.show()
# Heatmap for all variables vs gender
```



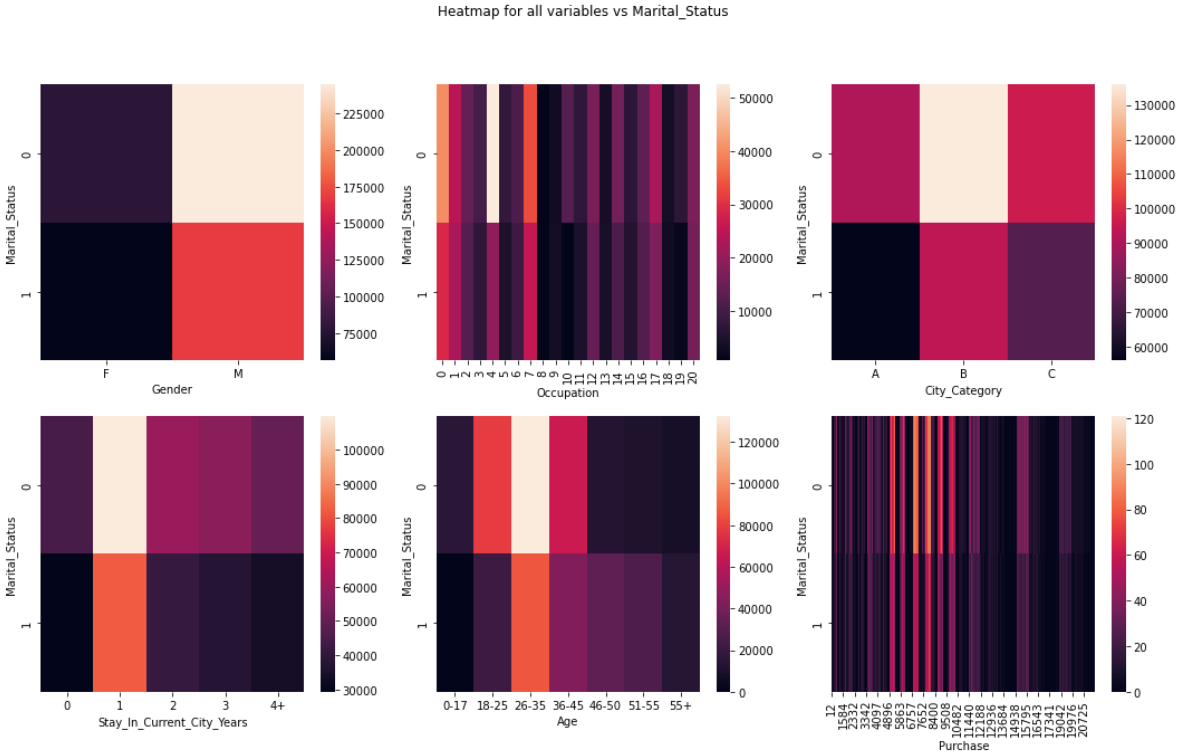
In [43]:

```
fig, axes = plt.subplots(2, 3, figsize=(18, 10))

fig.suptitle('Heatmap for all variables vs Age')
sns.heatmap(pd.crosstab(dff['Age'],dff['Gender']),ax=axes[0, 0])
sns.heatmap(pd.crosstab(dff['Age'],dff['Occupation']),ax=axes[0, 1])
sns.heatmap(pd.crosstab(dff['Age'],dff['City_Category']),ax=axes[0, 2])
sns.heatmap(pd.crosstab(dff['Age'],dff['Stay_In_Current_City_Years']),ax=axes[1, 0])
sns.heatmap(pd.crosstab(dff['Age'],dff['Marital_Status']),ax=axes[1, 1])
sns.heatmap(pd.crosstab(dff['Age'],dff['Purchase']),ax=axes[1, 2])
plt.show()
# Heatmap for all variables vs Age
```



In [44]:

```
fig, axes = plt.subplots(2, 3, figsize=(18, 10))

fig.suptitle('Heatmap for all variables vs Marital_Status')
sns.heatmap(pd.crosstab(dff['Marital_Status'],dff['Gender']),ax=axes[0, 0])
sns.heatmap(pd.crosstab(dff['Marital_Status'],dff['Occupation']),ax=axes[0, 1])
sns.heatmap(pd.crosstab(dff['Marital_Status'],dff['City_Category']),ax=axes[0, 2])
sns.heatmap(pd.crosstab(dff['Marital_Status'],dff['Stay_In_Current_City_Years']),ax=axes[1,
sns.heatmap(pd.crosstab(dff['Marital_Status'],dff['Age']),ax=axes[1, 1])
sns.heatmap(pd.crosstab(dff['Marital_Status'],dff['Purchase']),ax=axes[1, 2])
plt.show()
# Heatmap for all variables vs Marital_Status
```

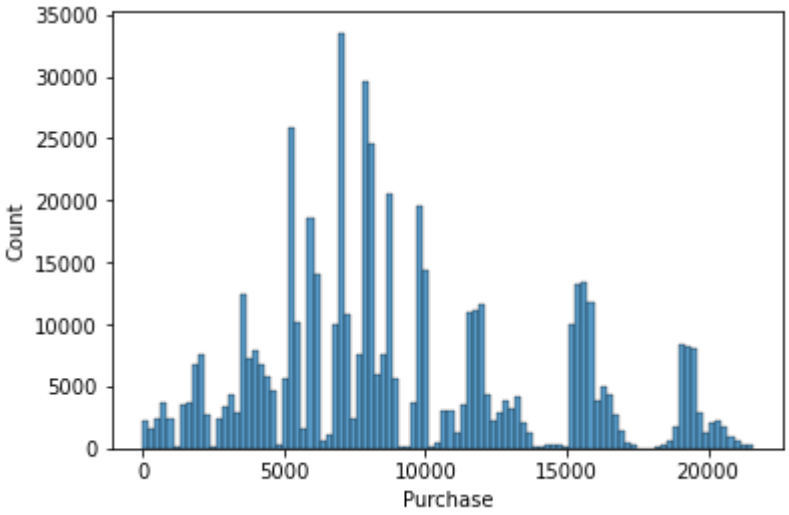


In [45]:

```
sns.histplot(dff['Purchase'],bins=100)
# Hist plot of purchase and count
```

Out[45]:

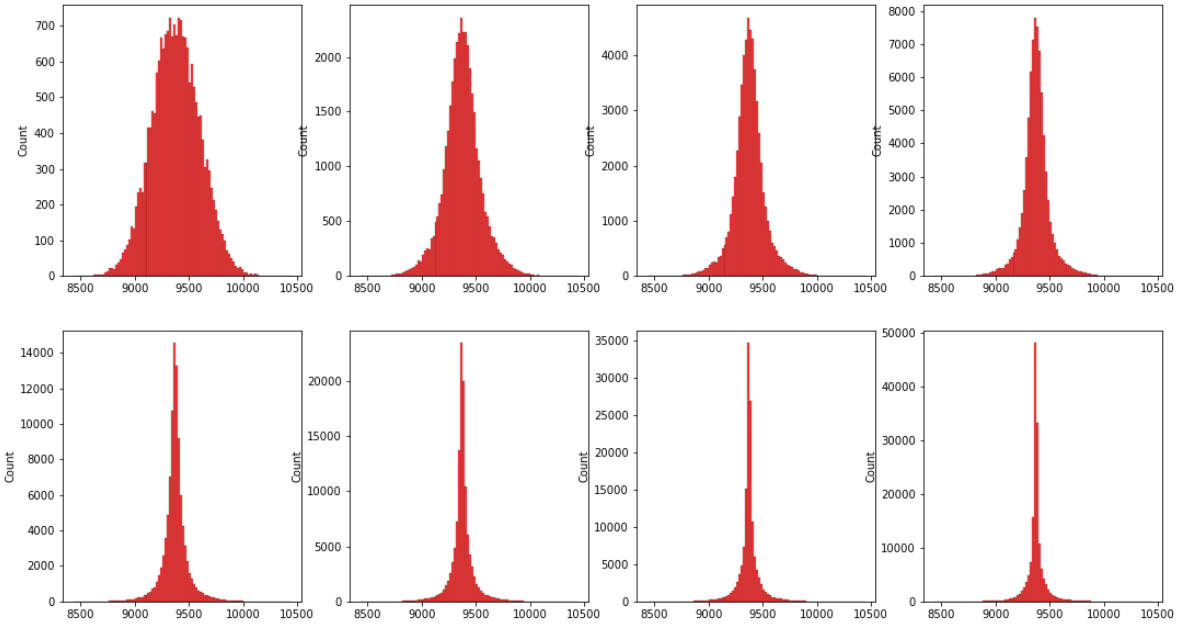
```
<AxesSubplot:xlabel='Purchase', ylabel='Count'>
```



In [46]:

```
data=dff[dff['Gender']=='M']
num=[500,2500,5000,10000,50000,100000,200000,400000]
len(data)
data=dff[dff['Gender']=='M']
fig, axes = plt.subplots(2, 4, figsize=(18, 10))
figs=[(0,0),(0,1),(0,2),(0,3),(1,0),(1,1),(1,2),(1,3)]
ls=[]
j=0
for _ in num:
    for i in range(20000):
        ans=np.random.choice(data['Purchase'],size=num[j],replace=True)
        l1=np.mean(ans)
        ls.append(l1)
    print('Confidence interval [{},{}] 90% confidence'.format(np.percentile(ls,5),np.percen
    sns.histplot(ls,color='r',ax=axes[figs[j]],bins=100)
    j=j+1
# Hist plot for all bootstraped sample size [500,2500,5000,10000,50000,100000,200000,400000
# are plotted for 90% confidence interval. More the number of sample size lesser is the sta
# spread of the hist plot
```

```
Confidence interval [9016.446100000001,9746.1367] 90% confidence
Confidence interval [9092.369700000001,9670.777100000001] 90% confidence
Confidence interval [9139.7447,9620.498] 90% confidence
Confidence interval [9172.5852,9584.8585] 90% confidence
Confidence interval [9197.81836,9559.818099999999] 90% confidence
Confidence interval [9216.05435,9539.844679999998] 90% confidence
Confidence interval [9230.51986,9523.72734] 90% confidence
Confidence interval [9243.323345,9510.0505] 90% confidence
```



In [47]:

```
def conf(para):
    per=[0.05,0.025,0.005]
    for a in dff[para].unique():
        data=dff[dff[para]==a]
        ls=[]
        for i in range(80000):
            ans=np.random.choice(data['Purchase'],len(data),replace=True)
            l1=np.mean(ans)
            ls.append(l1)
        print('Confidence interval using Bootstrap : ')
        print('Confidence interval for {} {} group is [{},{}] with {}% confidence'.format(a
        print('Confidence interval for {} {} group is [{},{}] with {}% confidence'.format(a
        print('Confidence interval for {} {} group is [{},{}] with {}% confidence'.format(a
        sample_mean=np.mean(ls)
        ans=0
        ls1=[]
        sample_std=np.std(ls,ddof=1)
        print()
        print('Confidence interval using formula : ')
        for i in [0.05,0.025,0.005]:
            CI1=sample_mean-(-stats.norm.ppf(i)*sample_std)
            CI2=sample_mean+(-stats.norm.ppf(i)*sample_std)
            print('Confidence interval for {} {} group is [{},{}] with {}% confidence'.form
        print('Standard Error of {} {} group is {}'.format(a,para,sample_std))
        print('-------------------------')
```

In [48]:

```
conf('Age')
# Confidence interval for all the age groups with pure bootstraping and bootstraping+CLT
```

```
Confidence interval using Bootstrap :
Confidence interval for 55+ Age group is [9129.976607403227,9238.560512205
448] with 90.0% confidence
Confidence interval for 55+ Age group is [9119.413230798174,9249.024219227
693] with 95.0% confidence
Confidence interval for 55+ Age group is [9098.816172804665,9269.166199379
146] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 55+ Age group is [9129.87593593939,9238.5521971093
86] with 90.0% confidence
Confidence interval for 55+ Age group is [9119.46619036764,9248.9619426811
35] with 95.0% confidence
Confidence interval for 55+ Age group is [9099.120933149163,9269.307199899
613] with 99.0% confidence
Standard Error of 55+ Age group is 33.03523772246333
-------------------------
Confidence interval using Bootstrap :
Confidence interval for 26-35 Age group is [9151.884593456252,9186.4157908
36552] with 90.0% confidence
Confidence interval for 26-35 Age group is [9148.476036009404,9189.7145893
77294] with 95.0% confidence
Confidence interval for 26-35 Age group is [9141.810843083877,9196.1661317
73247] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 26-35 Age group is [9151.914117644557,9186.4058551
29664] with 90.0% confidence
Confidence interval for 26-35 Age group is [9148.610266300373,9189.7097064
73848] with 95.0% confidence
Confidence interval for 26-35 Age group is [9142.153076261888,9196.1668965
12334] with 99.0% confidence
Standard Error of 26-35 Age group is 10.484743724288423
-------------------------
Confidence interval using Bootstrap :
Confidence interval for 46-50 Age group is [9063.258334620221,9137.7070385
40448] with 90.0% confidence
Confidence interval for 46-50 Age group is [9056.161245008714,9144.8100862
58245] with 95.0% confidence
Confidence interval for 46-50 Age group is [9042.503528535815,9158.9174645
37053] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 46-50 Age group is [9063.358917022812,9137.8917320
46213] with 90.0% confidence
Confidence interval for 46-50 Age group is [9056.2196611008,9145.030987968
225] with 95.0% confidence
Confidence interval for 46-50 Age group is [9042.266390381483,9158.9842586
87542] with 99.0% confidence
Standard Error of 46-50 Age group is 22.656367047546503
-------------------------
Confidence interval using Bootstrap :
Confidence interval for 36-45 Age group is [9202.348386047535,9250.9513752
72582] with 90.0% confidence
Confidence interval for 36-45 Age group is [9197.626508356087,9255.6555732
```

```
4403] with 95.0% confidence
Confidence interval for 36-45 Age group is [9188.515542183577,9264.7760658
13345] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 36-45 Age group is [9202.29521410442,9251.00850099
7925] with 90.0% confidence
Confidence interval for 36-45 Age group is [9197.629126800111,9255.6745883
02234] with 95.0% confidence
Confidence interval for 36-45 Age group is [9188.509523851539,9264.7941912
50807] with 99.0% confidence
Standard Error of 36-45 Age group is 14.807787785892254
-------------------------
Confidence interval using Bootstrap :
Confidence interval for 51-55 Age group is [9345.69378086582,9429.02811153
725] with 90.0% confidence
Confidence interval for 51-55 Age group is [9337.811700772303,9436.9919276
03238] with 95.0% confidence
Confidence interval for 51-55 Age group is [9321.95962488179,9452.71635363
0347] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 51-55 Age group is [9345.523919934361,9428.7958323
08335] with 90.0% confidence
Confidence interval for 51-55 Age group is [9337.547574302356,9436.7721779
4034] with 95.0% confidence
Confidence interval for 51-55 Age group is [9321.958259229465,9452.3614930
13232] with 99.0% confidence
Standard Error of 51-55 Age group is 25.312864017057166
-------------------------
Confidence interval using Bootstrap :
Confidence interval for 0-17 Age group is [8778.824773393762,8913.75674486
8035] with 90.0% confidence
Confidence interval for 0-17 Age group is [8765.94414156225,8926.773960277
258] with 95.0% confidence
Confidence interval for 0-17 Age group is [8741.21645294588,8953.398775993
068] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 0-17 Age group is [8778.701433356042,8913.54283666
3624] with 90.0% confidence
Confidence interval for 0-17 Age group is [8765.785413924956,8926.45885609
471] with 95.0% confidence
Confidence interval for 0-17 Age group is [8740.541786618878,8951.70248340
0788] with 99.0% confidence
Standard Error of 0-17 Age group is 40.98887618270691
-------------------------
Confidence interval using Bootstrap :
Confidence interval for 18-25 Age group is [9082.30902675565,9134.17278363
6108] with 90.0% confidence
Confidence interval for 18-25 Age group is [9077.318325705184,9139.0350346
2911] with 95.0% confidence
Confidence interval for 18-25 Age group is [9067.210364386958,9149.0014873
37944] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 18-25 Age group is [9082.245042294298,9134.1134297
40048] with 90.0% confidence
Confidence interval for 18-25 Age group is [9077.276738170525,9139.0817338
6382] with 95.0% confidence
Confidence interval for 18-25 Age group is [9067.566469591808,9148.7920024
```

```
42538] with 99.0% confidence
Standard Error of 18-25 Age group is 15.766870253945232
-------------------------
```

In [49]:

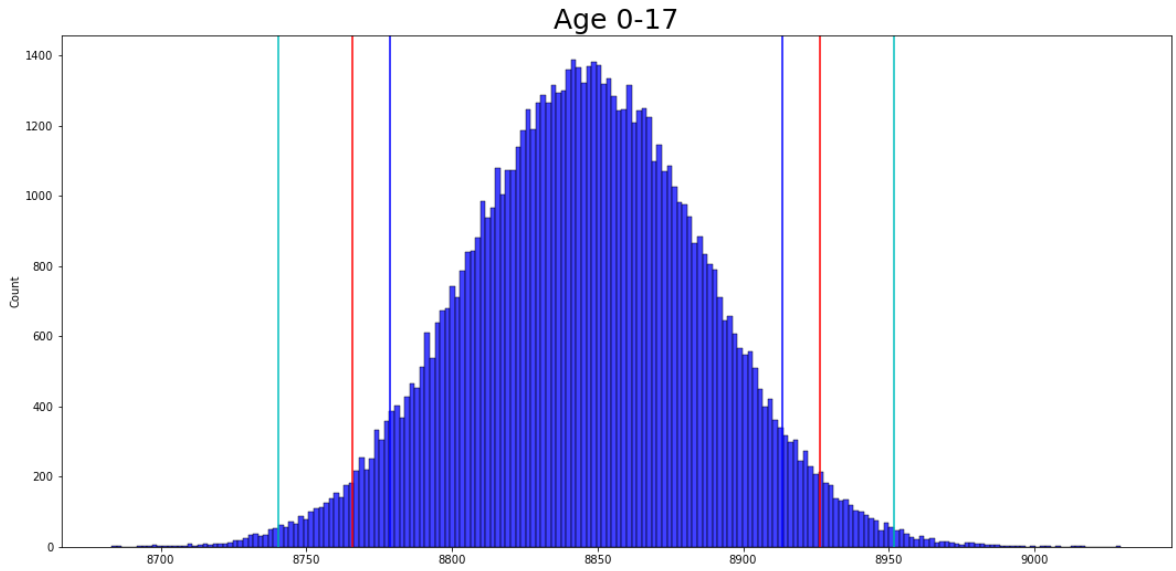
```
data=dff[dff['Age']=='0-17']
ls=[]
for i in range(80000):
    ans=np.random.choice(data['Purchase'],len(data),replace=True)
    l1=np.mean(ans)
    ls.append(l1)
fig, axes = plt.subplots(8, 1, figsize=(18, 80))
plt.subplot(8,1,1)
plt.axvline(x = 8778.68998720143, color = 'b')
plt.axvline(x = 8913.530672950867, color = 'b')
plt.axvline(x = 8765.774036502902, color = 'r')
plt.axvline(x = 8926.446623649394, color = 'r')
plt.axvline(x = 8740.530543530716, color = 'c')
plt.axvline(x = 8951.69011662158, color = 'c')
sns.histplot(ls,color='b',ax=axes[0],bins=200)
sns.histplot(ls,color='b',ax=axes[7],alpha=0.1,bins=200)


data=dff[dff['Age']=='18-25']
ls=[]
for i in range(80000):
    ans=np.random.choice(data['Purchase'],len(data),replace=True)
    l1=np.mean(ans)
    ls.append(l1)
plt.subplot(8,1,2)
plt.axvline(x = 9082.326032824567, color = 'b')
plt.axvline(x = 9134.293525818095, color = 'b')
plt.axvline(x = 9077.353368147267, color = 'r')
plt.axvline(x = 9139.20431780148, color = 'r')
plt.axvline(x = 9067.686842198116, color = 'c')
plt.axvline(x = 9148.930616922395, color = 'c')
sns.histplot(ls,color='g',ax=axes[1],bins=200)
sns.histplot(ls,color='g',ax=axes[7],alpha=0.2,bins=200)



data=dff[dff['Age']=='26-35']
ls=[]
for i in range(80000):
    ans=np.random.choice(data['Purchase'],len(data),replace=True)
    l1=np.mean(ans)
    ls.append(l1)
plt.subplot(8,1,3)
plt.axvline(x = 9152.011840893154, color = 'b')
plt.axvline(x = 9186.529130632056, color = 'b')
plt.axvline(x = 9148.736572896472, color = 'r')
plt.axvline(x = 9189.84858428089, color = 'r')
plt.axvline(x = 9142.427178232116, color = 'c')
plt.axvline(x = 9196.323326619095, color = 'c')
sns.histplot(ls,color='r',ax=axes[2],bins=200)
sns.histplot(ls,color='r',ax=axes[7],alpha=0.3,bins=200)



data=dff[dff['Age']=='36-45']
ls=[]
for i in range(80000):
    ans=np.random.choice(data['Purchase'],len(data),replace=True)
    l1=np.mean(ans)
```

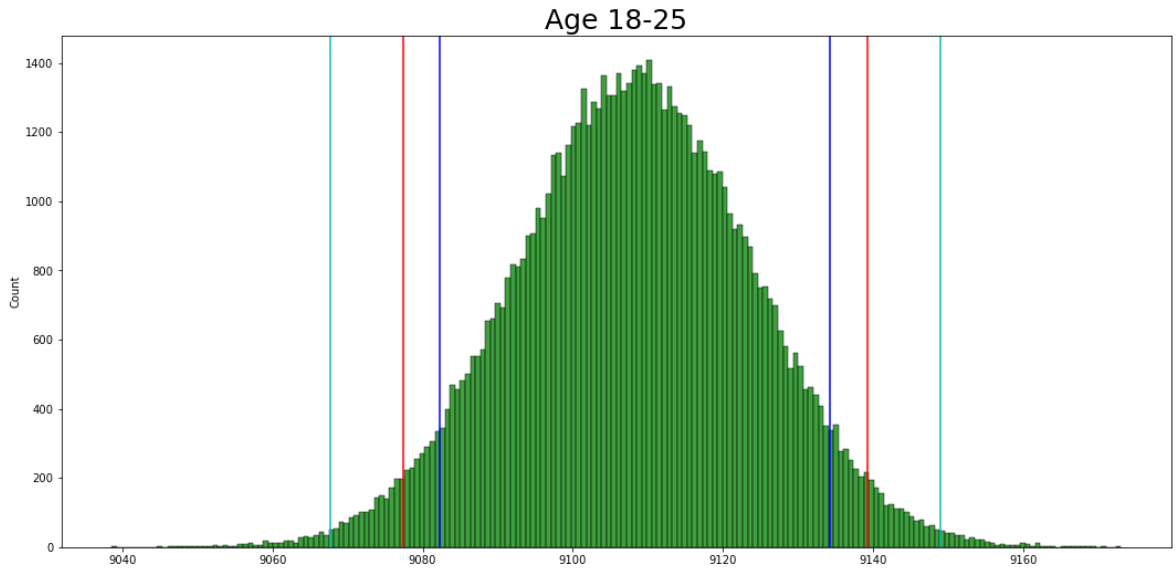
```
    ls.append(l1)
plt.subplot(8,1,4)
plt.axvline(x = 9202.196913653772, color = 'b')
plt.axvline(x = 9251.083900789798, color = 'b')
plt.axvline(x = 9197.679161321947, color = 'r')
plt.axvline(x = 9255.79458411977, color = 'r')
plt.axvline(x = 9188.473899736124, color = 'c')
plt.axvline(x = 9264.889758571402, color = 'c')
sns.histplot(ls,color='c',ax=axes[3],bins=200)
sns.histplot(ls,color='c',ax=axes[7],alpha=0.4,bins=200)



data=dff[dff['Age']=='46-50']
ls=[]
for i in range(80000):
    ans=np.random.choice(data['Purchase'],len(data),replace=True)
    l1=np.mean(ans)
    ls.append(l1)
plt.subplot(8,1,5)
plt.axvline(x = 9063.373719914403, color = 'b')
plt.axvline(x = 9137.900151117386, color = 'b')
plt.axvline(x = 9056.135459088002, color = 'r')
plt.axvline(x = 9145.049257098104, color = 'r')
plt.axvline(x = 9042.159357916566, color = 'c')
plt.axvline(x = 9158.336188091509, color = 'c')
sns.histplot(ls,color='m',ax=axes[4],bins=200)
sns.histplot(ls,color='m',ax=axes[7],alpha=0.5,bins=200)



data=dff[dff['Age']=='51-55']
ls=[]
for i in range(80000):
    ans=np.random.choice(data['Purchase'],len(data),replace=True)
    l1=np.mean(ans)
    ls.append(l1)
plt.subplot(8,1,6)
plt.axvline(x = 9345.910250078807, color = 'b')
plt.axvline(x = 9428.538028002522, color = 'b')
plt.axvline(x = 9338.308481533046, color = 'r')
plt.axvline(x = 9436.449856178417, color = 'r')
plt.axvline(x = 9322.947133681833, color = 'c')
plt.axvline(x = 9452.547997662079, color = 'c')
sns.histplot(ls,color='y',ax=axes[5],bins=200)
sns.histplot(ls,color='y',ax=axes[7],alpha=0.6,bins=200)



data=dff[dff['Age']=='55+']
ls=[]
for i in range(80000):
    ans=np.random.choice(data['Purchase'],len(data),replace=True)
    l1=np.mean(ans)
    ls.append(l1)
plt.subplot(8,1,7)
plt.axvline(x = 9129.892566201024, color = 'b')
plt.axvline(x = 9238.351533323927, color = 'b')
plt.axvline(x = 9119.612811015475, color = 'r')
plt.axvline(x = 9248.639851135884, color = 'r')
plt.axvline(x = 9099.703916796012, color = 'c')
```

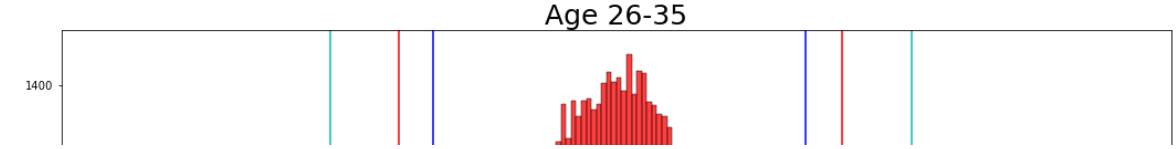
```
plt.axvline(x = 9268.314920511735, color = 'c')
sns.histplot(ls,color='k',ax=axes[6],bins=200)
sns.histplot(ls,color='k',ax=axes[7],alpha=0.7,bins=200)
axes[0].set_title("Age 0-17",fontsize=25)
axes[1].set_title("Age 18-25",fontsize=25)
axes[2].set_title("Age 26-35",fontsize=25)
axes[3].set_title("Age 36-45",fontsize=25)
axes[4].set_title("Age 46-50",fontsize=25)
axes[5].set_title("Age 51-55",fontsize=25)
axes[6].set_title("Age 55+",fontsize=25)
axes[7].set_title("Combined plot",fontsize=25)

# Hist plot for each age group in increaseing order. Confidence interval lines shown on blu
# Age 0-17 and 51-55  confidence interval does not coincide with any other group
# Age group 18-25,26-35,36-45, 46-50 and 55+ confidence interval are overlapping.
# Walmart can recommend same set of products to these age groups 18-25,26-35,36-45, 46-50 a
```

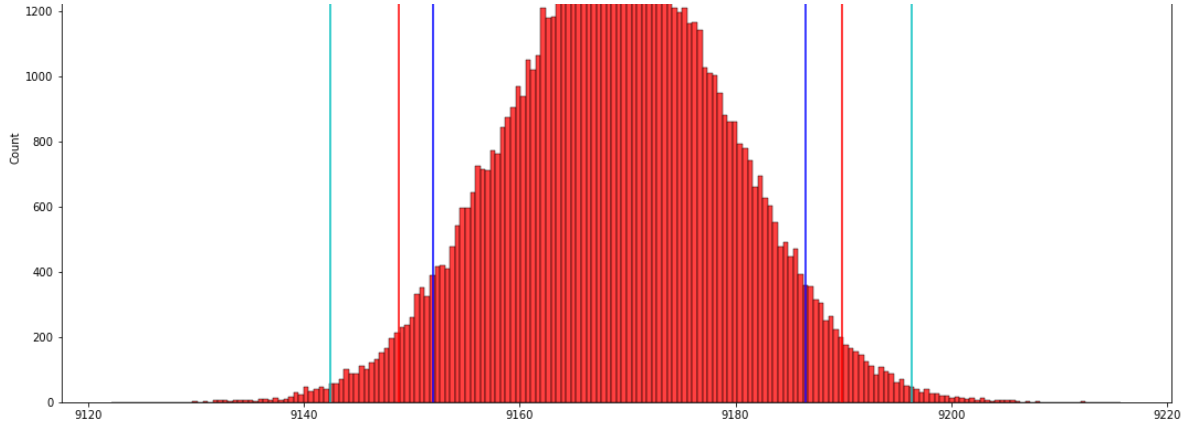
Out[49]:

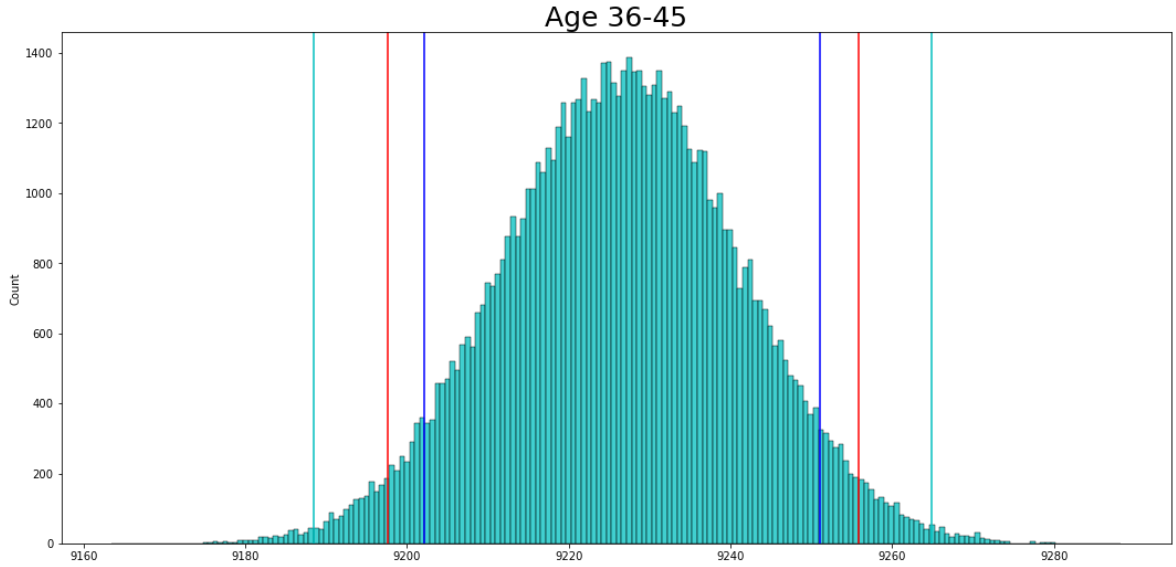
```
Text(0.5, 1.0, 'Combined plot')
```

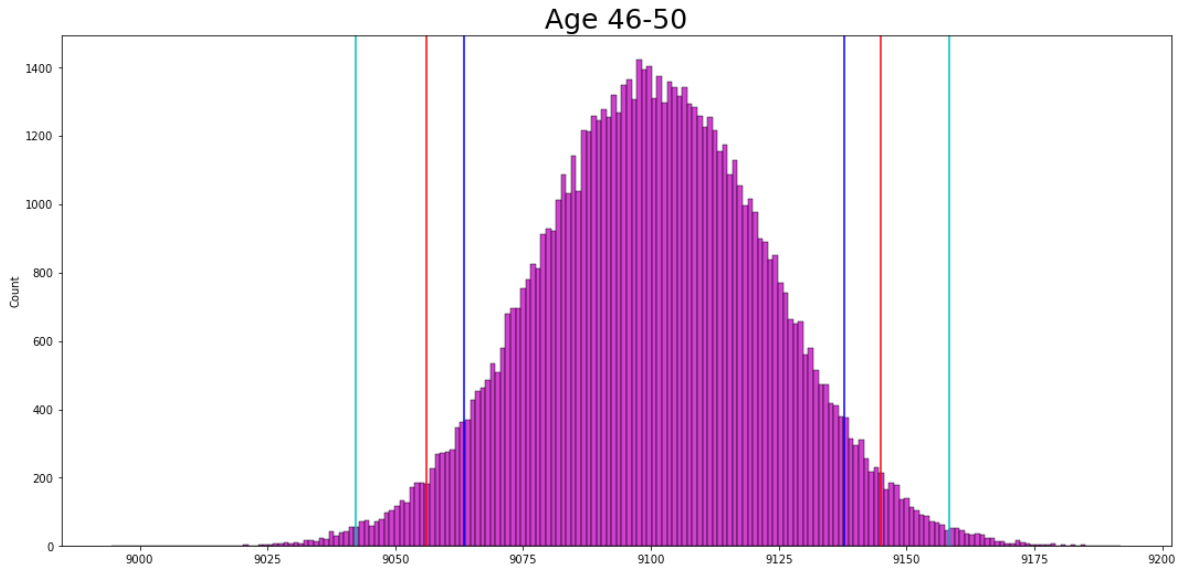


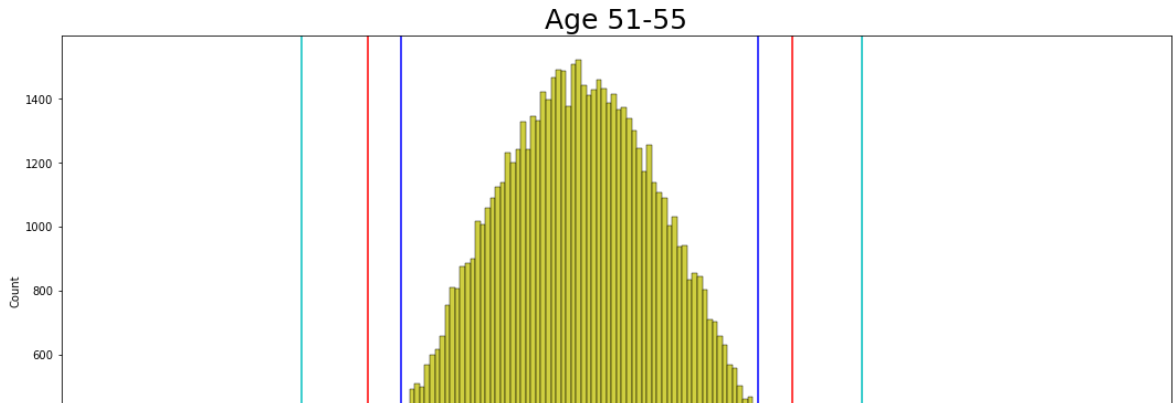


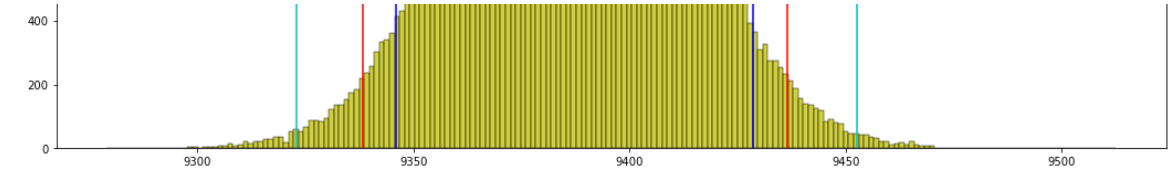


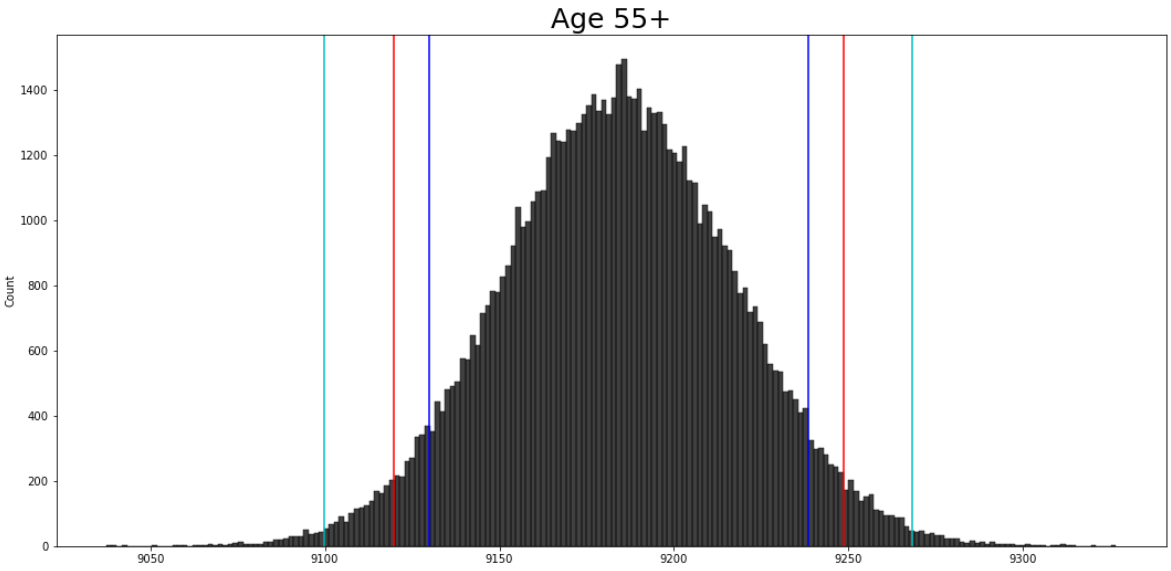

Count
1200
1000
800
600
400
200
0
9120
9140
9160
9180
9200
9220

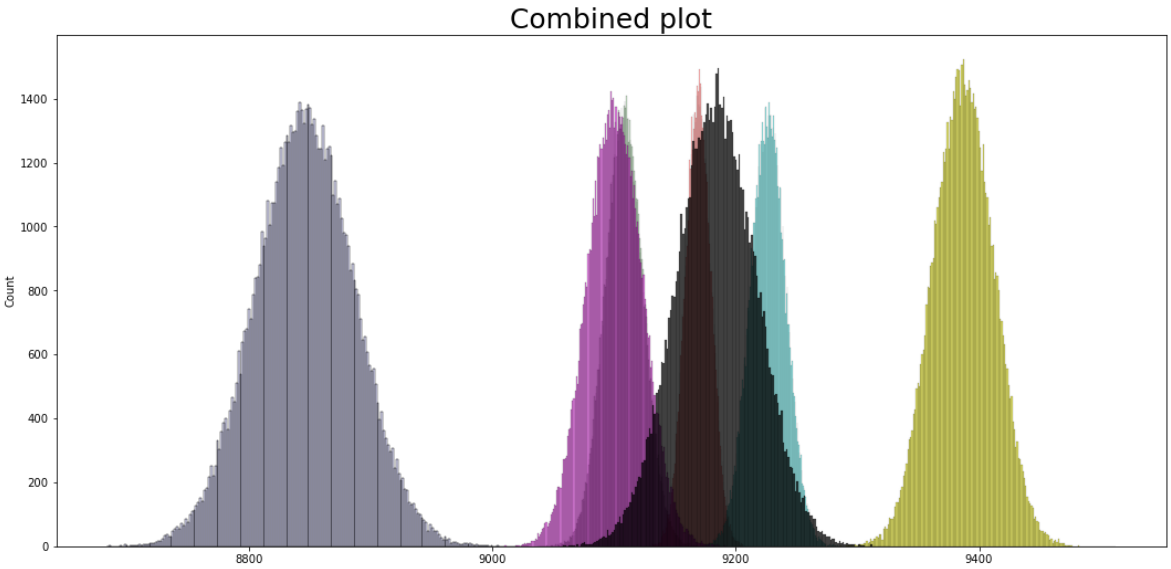

Age 36-45
Count
1400
1200
1000
800
600
400
200
0
9160
9180
9200
9220
9240
9260
9280


Age 46-50
Count
1400
1200
1000
800
600
400
200
0
9000
9025
9050
9075
9100
9125
9150
9175
9200


Age 51-55
Count
1400
1200
1000
800
600


400
200
0
9300
9350
9400
9450
9500


Age 55+
Count
1400
1200
1000
800
600
400
200
0
9050
9100
9150
9200
9250
9300


Combined plot
Count
1400
1200
1000
800
600
400
200
0
8800
9000
9200
9400

In [50]:

```
conf('Gender')
# Confidence interval for all the Gender groups with pure bootstraping and bootstraping+CLT
```

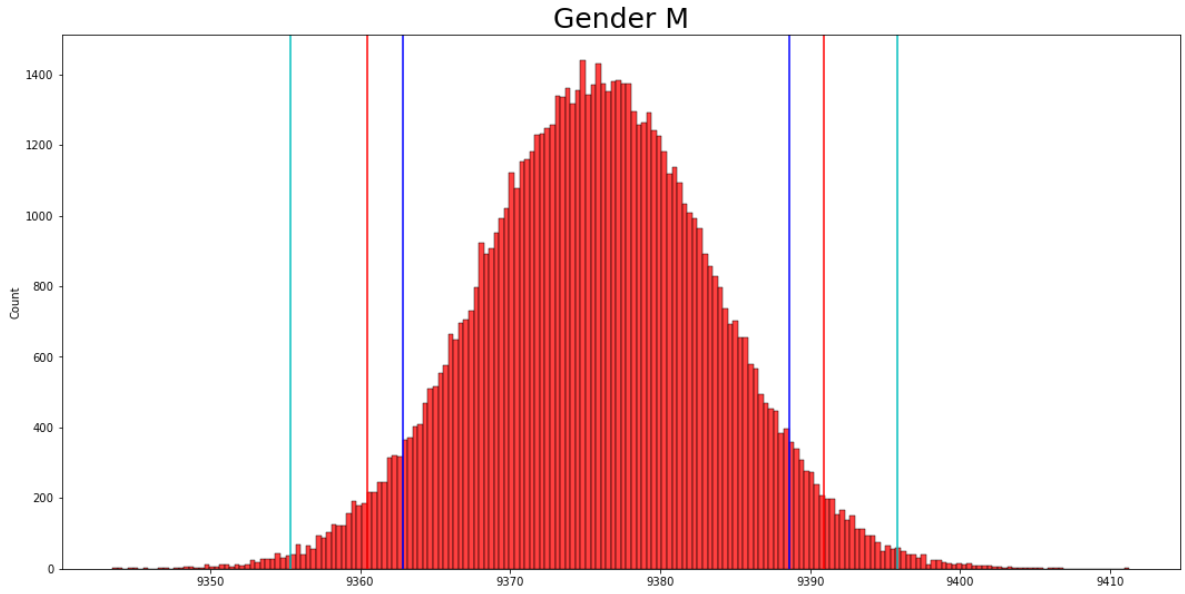
```
Confidence interval using Bootstrap :
Confidence interval for M Gender group is [9362.925427630702,9388.6733189961
38] with 90.0% confidence
Confidence interval for M Gender group is [9360.487847590115,9391.1253236173
38] with 95.0% confidence
Confidence interval for M Gender group is [9355.550190763905,9396.0618403940
39] with 99.0% confidence

Confidence interval using formula :
Confidence interval for M Gender group is [9362.911646598195,9388.6971861195
29] with 90.0% confidence
Confidence interval for M Gender group is [9360.441733638754,9391.1670990789
7] with 95.0% confidence
Confidence interval for M Gender group is [9355.614428867402,9395.9944038503
22] with 99.0% confidence
Standard Error of M Gender group is 7.8382474582624475
-------------------------
Confidence interval using Bootstrap :
Confidence interval for F Gender group is [8518.210458786936,8558.8385490190
21] with 90.0% confidence
Confidence interval for F Gender group is [8514.293912997968,8562.7089660096
9] with 95.0% confidence
Confidence interval for F Gender group is [8506.510455197991,8570.5421626390
73] with 99.0% confidence

Confidence interval using formula :
Confidence interval for F Gender group is [8518.18887550363,8558.8381066323
9] with 90.0% confidence
Confidence interval for F Gender group is [8514.295217871484,8562.7317642645
36] with 95.0% confidence
Confidence interval for F Gender group is [8506.685284900435,8570.3416972355
84] with 99.0% confidence
Standard Error of F Gender group is 12.356488888344776
-------------------------
```

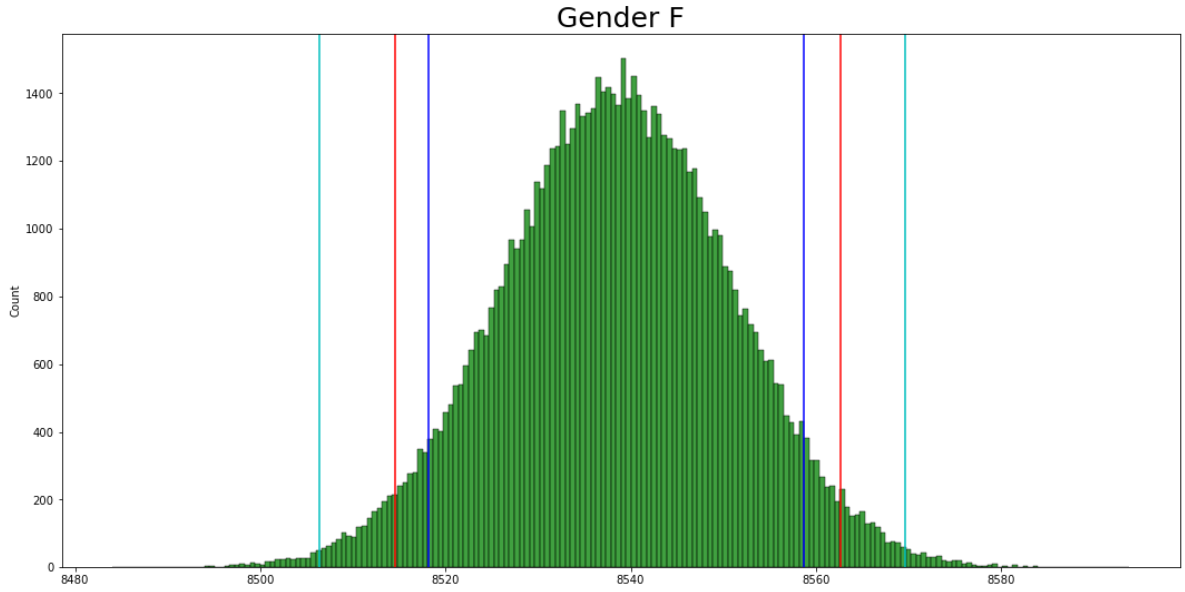
In [51]:

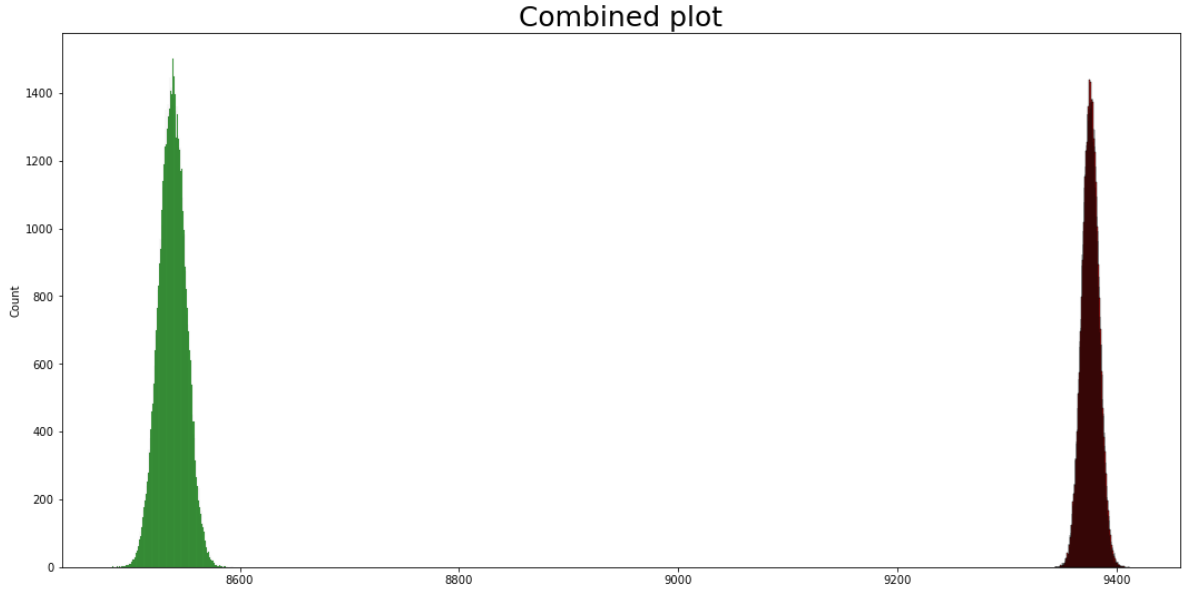
```
data=dff[dff['Gender']=='M']
ls=[]
for i in range(80000):
    ans=np.random.choice(data['Purchase'],len(data),replace=True)
    l1=np.mean(ans)
    ls.append(l1)
#sns.histplot(ls,color='r')
fig, axes = plt.subplots(3, 1, figsize=(18, 30))
plt.subplot(3,1,1)
plt.axvline(x = 9362.883360407519, color = 'b')
plt.axvline(x = 9388.616827859096, color = 'b')
plt.axvline(x = 9360.501878059484, color = 'r')
plt.axvline(x = 9390.93134905818, color = 'r')
plt.axvline(x = 9355.37927838001, color = 'c')
plt.axvline(x = 9395.833137833466, color = 'c')
sns.histplot(ls,color='r',ax=axes[0],bins=200)
sns.histplot(ls,color='r',ax=axes[2],bins=200)

data=dff[dff['Gender']=='F']
ls=[]
for i in range(80000):
    ans=np.random.choice(data['Purchase'],len(data),replace=True)
    l1=np.mean(ans)
    ls.append(l1)
plt.subplot(3,1,2)
sns.histplot(ls,color='g',ax=axes[1],bins=200)
plt.axvline(x = 8518.135753005741, color = 'b')
plt.axvline(x = 8558.712708233641, color = 'b')
plt.axvline(x = 8514.54182972694, color = 'r')
plt.axvline(x = 8562.678987468596, color = 'r')
plt.axvline(x = 8506.36232619781, color = 'c')
plt.axvline(x = 8569.673823835088, color = 'c')
sns.histplot(ls,color='g',ax=axes[2],bins=200)
axes[0].set_title("Gender M",fontsize=25)
axes[1].set_title("Gender F",fontsize=25)
axes[2].set_title("Combined plot",fontsize=25)
# Hist plot for each gender group. Confidence interval lines shown on blue, red and cyan co
# Gender M and F confidence interval does not coincide.
# Walmart need to provide separate product recommendations for male and female
```

Out[51]:

```
Text(0.5, 1.0, 'Combined plot')
```


Gender M
Count
0
200
400
600
800
1000
1200
1400
9350
9360
9370
9380
9390
9400
9410


Gender F
Count
0
200
400
600
800
1000
1200
1400
8480
8500
8520
8540
8560
8580


Combined plot
Count
0
200
400
600
800
1000
1200
1400
8600
8800
9000
9200
9400

In [52]:

```
conf('Marital_Status')
# Confidence interval for all the Marital_Status groups with pure bootstraping and bootstra
```

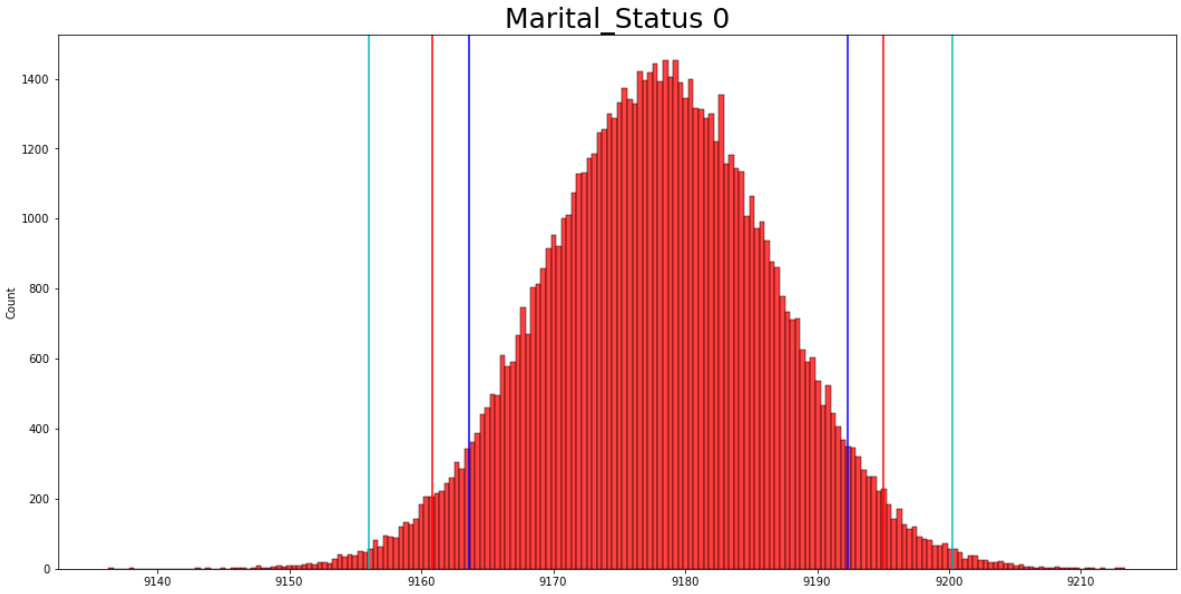
```
Confidence interval using Bootstrap :
Confidence interval for 0 Marital_Status group is [9163.819304301729,9192.37
0339897325] with 90.0% confidence
Confidence interval for 0 Marital_Status group is [9161.114833228963,9195.22
2985076014] with 95.0% confidence
Confidence interval for 0 Marital_Status group is [9155.66322296384,9200.738
182212743] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 0 Marital_Status group is [9163.797564009896,9192.40
9412633711] with 90.0% confidence
Confidence interval for 0 Marital_Status group is [9161.056928094518,9195.15
0048549089] with 95.0% confidence
Confidence interval for 0 Marital_Status group is [9155.700510662775,9200.50
6465980832] with 99.0% confidence
Standard Error of 0 Marital_Status group is 8.697384422237286
-------------------------
Confidence interval using Bootstrap :
Confidence interval for 1 Marital_Status group is [9143.359306511406,9177.49
471923729] with 90.0% confidence
Confidence interval for 1 Marital_Status group is [9140.08492339786,9180.856
12537619] with 95.0% confidence
Confidence interval for 1 Marital_Status group is [9133.476693385743,9187.14
6882655183] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 1 Marital_Status group is [9143.25827429847,9177.453
538842397] with 90.0% confidence
Confidence interval for 1 Marital_Status group is [9139.982821132999,9180.72
899200787] with 95.0% confidence
Confidence interval for 1 Marital_Status group is [9133.581133723897,9187.13
067941697] with 99.0% confidence
Standard Error of 1 Marital_Status group is 10.394622349255114
-------------------------
```

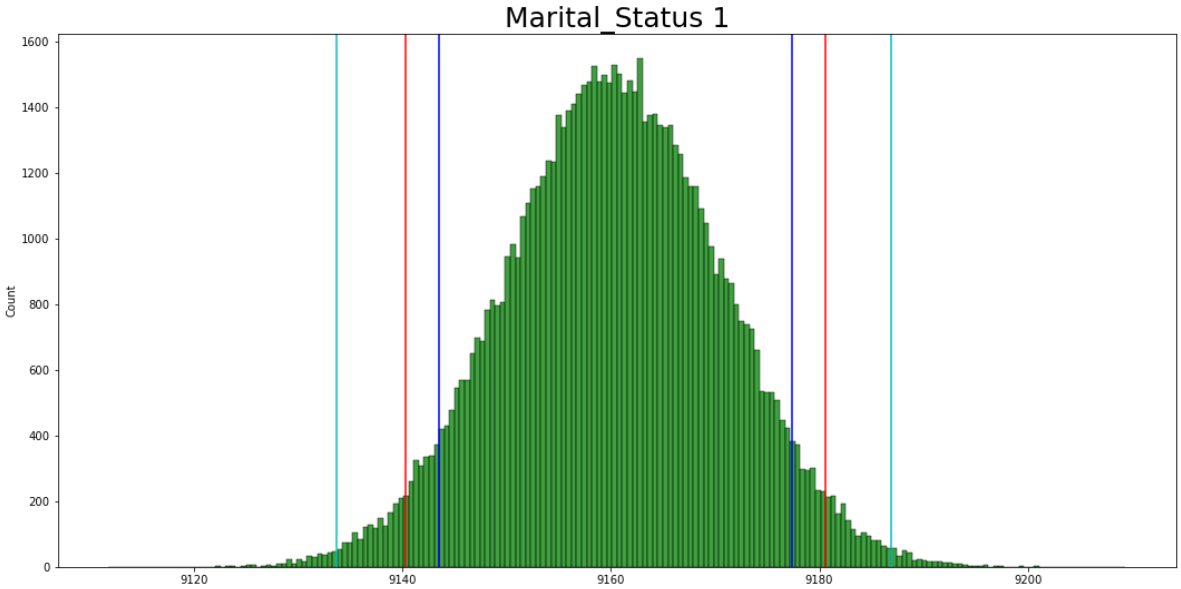
In [53]:

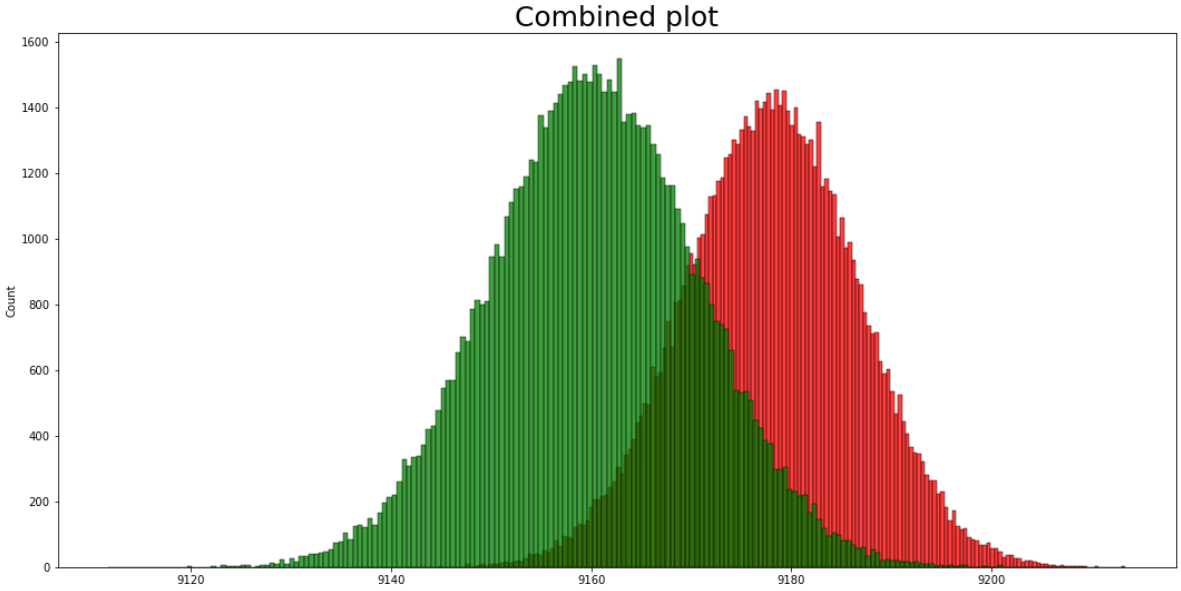
```
data=dff[dff['Marital_Status']==0]
ls=[]
for i in range(80000):
    ans=np.random.choice(data['Purchase'],len(data),replace=True)
    l1=np.mean(ans)
    ls.append(l1)
#sns.histplot(ls,color='r')
fig, axes = plt.subplots(3, 1, figsize=(18, 30))
plt.subplot(3,1,1)
plt.axvline(x = 9163.598699654027, color = 'b')
plt.axvline(x = 9192.334477846533, color = 'b')
plt.axvline(x = 9160.853929714045, color = 'r')
plt.axvline(x = 9195.059070490564, color = 'r')
plt.axvline(x = 9156.066792226755, color = 'c')
plt.axvline(x = 9200.223922258252, color = 'c')
sns.histplot(ls,color='r',ax=axes[0],bins=200)
sns.histplot(ls,color='r',ax=axes[2],bins=200)

data=dff[dff['Marital_Status']==1]
ls=[]
for i in range(80000):
    ans=np.random.choice(data['Purchase'],len(data),replace=True)
    l1=np.mean(ans)
    ls.append(l1)
plt.subplot(3,1,2)
sns.histplot(ls,color='g',ax=axes[1],bins=200)
plt.axvline(x = 9143.515349494462, color = 'b')
plt.axvline(x = 9177.339473131118, color = 'b')
plt.axvline(x = 9140.336320056524, color = 'r')
plt.axvline(x = 9180.535914574977, color = 'r')
plt.axvline(x = 9133.694100226721, color = 'c')
plt.axvline(x = 9186.846329268457, color = 'c')
sns.histplot(ls,color='g',ax=axes[2],bins=200)

axes[0].set_title("Marital_Status 0",fontsize=25)
axes[1].set_title("Marital_Status 1",fontsize=25)
axes[2].set_title("Combined plot",fontsize=25)
# Hist plot for each Marital_Status group . Confidence interval lines shown on blue, red an
# Marital status 0 and 1 confidence interval are overlapping
# Walmart can recommend same set of products to all irrespective of the status
```

Out[53]:

```
Text(0.5, 1.0, 'Combined plot')
```


Marital_Status 0
Count
0
200
400
600
800
1000
1200
1400
9140
9150
9160
9170
9180
9190
9200
9210


Marital_Status 1
Count
0
200
400
600
800
1000
1200
1400
1600
9120
9140
9160
9180
9200


Combined plot
Count
0
200
400
600
800
1000
1200
1400
1600
9120
9140
9160
9180
9200

In [54]:

```
conf('Stay_In_Current_City_Years')
# Confidence interval for all the Stay_In_Current_City_Years groups with pure bootstraping
```

```
Confidence interval using Bootstrap :
Confidence interval for 4+ Stay_In_Current_City_Years group is [9159.48610
1209572,9215.453840212924] with 90.0% confidence
Confidence interval for 4+ Stay_In_Current_City_Years group is [9154.24035
1940305,9220.852027043084] with 95.0% confidence
Confidence interval for 4+ Stay_In_Current_City_Years group is [9144.05545
09755,9231.30803402961] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 4+ Stay_In_Current_City_Years group is [9159.49851
5879372,9215.423643877708] with 90.0% confidence
Confidence interval for 4+ Stay_In_Current_City_Years group is [9154.14162
977322,9220.78052998386] with 95.0% confidence
Confidence interval for 4+ Stay_In_Current_City_Years group is [9143.67189
975258,9231.250260004499] with 99.0% confidence
Standard Error of 4+ Stay_In_Current_City_Years group is 17.00003182106388
5
-------------------------
Confidence interval using Bootstrap :
Confidence interval for 3 Stay_In_Current_City_Years group is [9163.383086
81264,9215.70001004409] with 90.0% confidence
Confidence interval for 3 Stay_In_Current_City_Years group is [9158.244164
384721,9220.684632280641] with 95.0% confidence
Confidence interval for 3 Stay_In_Current_City_Years group is [9148.268523
677616,9230.703974921498] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 3 Stay_In_Current_City_Years group is [9163.292645
373953,9215.749168177375] with 90.0% confidence
Confidence interval for 3 Stay_In_Current_City_Years group is [9158.268005
677266,9220.773807874062] with 95.0% confidence
Confidence interval for 3 Stay_In_Current_City_Years group is [9148.447632
416972,9230.594181134356] with 99.0% confidence
Standard Error of 3 Stay_In_Current_City_Years group is 15.945650708338176
-------------------------
Confidence interval using Bootstrap :
Confidence interval for 2 Stay_In_Current_City_Years group is [9211.478338
702665,9262.532943809789] with 90.0% confidence
Confidence interval for 2 Stay_In_Current_City_Years group is [9206.668902
185944,9267.618778411337] with 95.0% confidence
Confidence interval for 2 Stay_In_Current_City_Years group is [9196.927955
421377,9277.468715511108] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 2 Stay_In_Current_City_Years group is [9211.574400
723184,9262.570496528086] with 90.0% confidence
Confidence interval for 2 Stay_In_Current_City_Years group is [9206.689650
57843,9267.45524667284] with 95.0% confidence
Confidence interval for 2 Stay_In_Current_City_Years group is [9197.142683
512035,9277.002213739235] with 99.0% confidence
Standard Error of 2 Stay_In_Current_City_Years group is 15.501712422708295
-------------------------
Confidence interval using Bootstrap :
Confidence interval for 1 Stay_In_Current_City_Years group is [9132.248336
452485,9168.982929403202] with 90.0% confidence
```

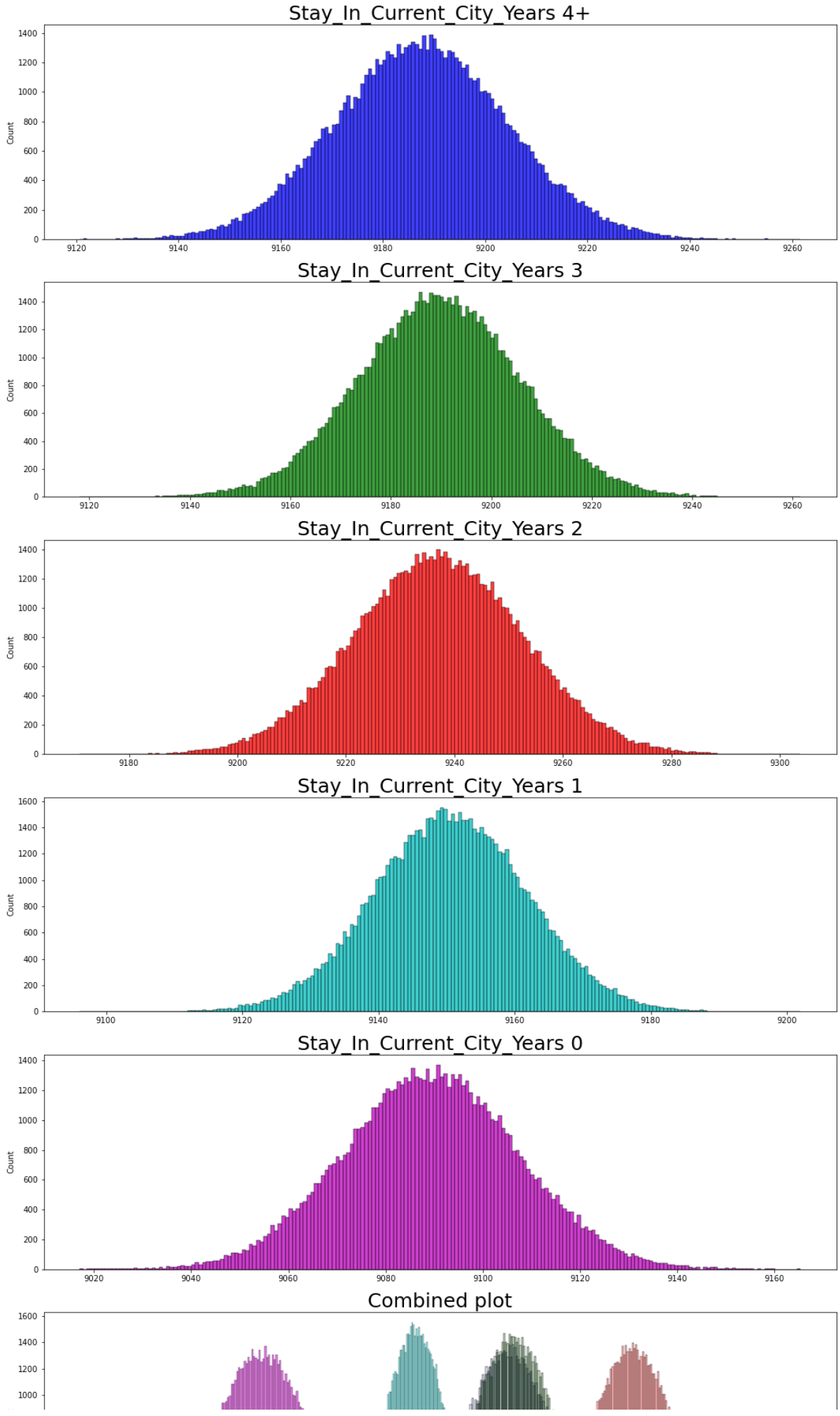
```
Confidence interval for 1 Stay_In_Current_City_Years group is [9128.644849
760864,9172.665518558952] with 95.0% confidence
Confidence interval for 1 Stay_In_Current_City_Years group is [9121.640373
75234,9179.249314904346] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 1 Stay_In_Current_City_Years group is [9132.237581
001675,9169.04539742615] with 90.0% confidence
Confidence interval for 1 Stay_In_Current_City_Years group is [9128.711879
989278,9172.571098438548] with 95.0% confidence
Confidence interval for 1 Stay_In_Current_City_Years group is [9121.821097
358883,9179.461881068943] with 99.0% confidence
Standard Error of 1 Stay_In_Current_City_Years group is 11.188781731507373
-------------------------
Confidence interval using Bootstrap :
Confidence interval for 0 Stay_In_Current_City_Years group is [9060.020426
556961,9119.126492678703] with 90.0% confidence
Confidence interval for 0 Stay_In_Current_City_Years group is [9054.437685
739033,9124.876201383062] with 95.0% confidence
Confidence interval for 0 Stay_In_Current_City_Years group is [9043.114216
512842,9135.405105827265] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 0 Stay_In_Current_City_Years group is [9059.986037
087672,9119.015471149802] with 90.0% confidence
Confidence interval for 0 Stay_In_Current_City_Years group is [9054.331799
600886,9124.669708636588] with 95.0% confidence
Confidence interval for 0 Stay_In_Current_City_Years group is [9043.280913
174303,9135.72059506317] with 99.0% confidence
Standard Error of 0 Stay_In_Current_City_Years group is 17.943673860978578
-------------------------
```

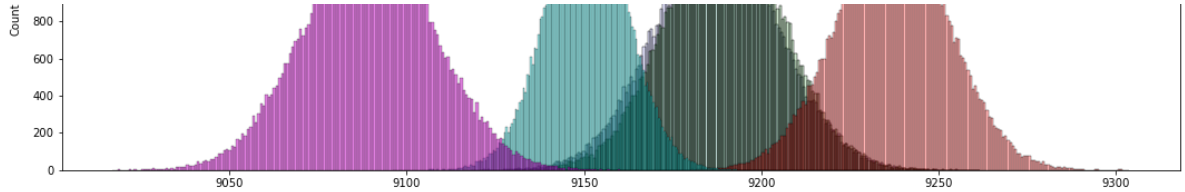
In [55]:

```
data=dff[dff['Stay_In_Current_City_Years']=='4+']
ls=[]
for i in range(80000):
    ans=np.random.choice(data['Purchase'],len(data),replace=True)
    l1=np.mean(ans)
    ls.append(l1)
fig, axes = plt.subplots(6, 1, figsize=(18, 35))
sns.histplot(ls,color='b',ax=axes[0],bins=200)
sns.histplot(ls,color='b',ax=axes[5],alpha=0.1,bins=200)

data=dff[dff['Stay_In_Current_City_Years']=='3']
ls=[]
for i in range(80000):
    ans=np.random.choice(data['Purchase'],len(data),replace=True)
    l1=np.mean(ans)
    ls.append(l1)
sns.histplot(ls,color='g',ax=axes[1],bins=200)
sns.histplot(ls,color='g',ax=axes[5],alpha=0.2,bins=200)


data=dff[dff['Stay_In_Current_City_Years']=='2']
ls=[]
for i in range(80000):
    ans=np.random.choice(data['Purchase'],len(data),replace=True)
    l1=np.mean(ans)
    ls.append(l1)
sns.histplot(ls,color='r',ax=axes[2],bins=200)
sns.histplot(ls,color='r',ax=axes[5],alpha=0.3,bins=200)


data=dff[dff['Stay_In_Current_City_Years']=='1']
ls=[]
for i in range(80000):
    ans=np.random.choice(data['Purchase'],len(data),replace=True)
    l1=np.mean(ans)
    ls.append(l1)
sns.histplot(ls,color='c',ax=axes[3],bins=200)
sns.histplot(ls,color='c',ax=axes[5],alpha=0.4,bins=200)


data=dff[dff['Stay_In_Current_City_Years']=='0']
ls=[]
for i in range(80000):
    ans=np.random.choice(data['Purchase'],len(data),replace=True)
    l1=np.mean(ans)
    ls.append(l1)
sns.histplot(ls,color='m',ax=axes[4],bins=200)
sns.histplot(ls,color='m',ax=axes[5],alpha=0.5,bins=200)
axes[0].set_title("Stay_In_Current_City_Years 4+",fontsize=25)
axes[1].set_title("Stay_In_Current_City_Years 3",fontsize=25)
axes[2].set_title("Stay_In_Current_City_Years 2",fontsize=25)
axes[3].set_title("Stay_In_Current_City_Years 1",fontsize=25)
axes[4].set_title("Stay_In_Current_City_Years 0",fontsize=25)
axes[5].set_title("Combined plot",fontsize=25)
# Hist plot for each Stay_In_Current_City_Years group.
# Stay_In_Current_City_Years 4+,3,2 and 1 are over lapping for 90% confidence.
# Walmart can have a common recommendation for Stay_In_Current_City_Years 4+,3,2 and 1 for
```

Out[55]:

```
Text(0.5, 1.0, 'Combined plot')
```




Count
800
600
400
200
0
9050
9100
9150
9200
9250
9300

In [56]:

```
conf('City_Category')
# Confidence interval for all the City_Category groups with pure bootstraping and bootstrap
```

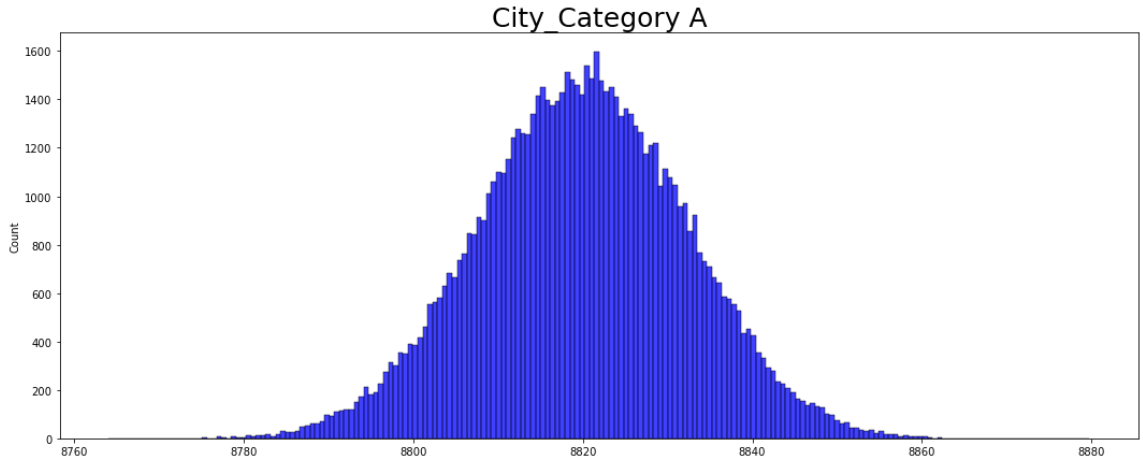
```
Confidence interval using Bootstrap :
Confidence interval for C City_Category group is [9597.927719772455,9638.525
291793278] with 90.0% confidence
Confidence interval for C City_Category group is [9594.143685222482,9642.337
788996067] with 95.0% confidence
Confidence interval for C City_Category group is [9586.363280686877,9649.960
784220197] with 99.0% confidence

Confidence interval using formula :
Confidence interval for C City_Category group is [9597.936158711493,9638.478
002986443] with 90.0% confidence
Confidence interval for C City_Category group is [9594.052787316781,9642.361
374381155] with 95.0% confidence
Confidence interval for C City_Category group is [9586.462958213302,9649.951
203484634] with 99.0% confidence
Standard Error of C City_Category group is 12.32384560262947
-------------------------
Confidence interval using Bootstrap :
Confidence interval for A City_Category group is [8799.593894667249,8840.394
623285803] with 90.0% confidence
Confidence interval for A City_Category group is [8795.483671174241,8844.250
125241962] with 95.0% confidence
Confidence interval for A City_Category group is [8787.582318288394,8851.559
287296544] with 99.0% confidence

Confidence interval using formula :
Confidence interval for A City_Category group is [8799.53037799453,8840.3853
13395467] with 90.0% confidence
Confidence interval for A City_Category group is [8795.617016619237,8844.298
67477076] with 95.0% confidence
Confidence interval for A City_Category group is [8787.968573799915,8851.947
117590082] with 99.0% confidence
Standard Error of A City_Category group is 12.419018547156448
-------------------------
Confidence interval using Bootstrap :
Confidence interval for B City_Category group is [9047.42013275974,9080.8374
92978844] with 90.0% confidence
Confidence interval for B City_Category group is [9044.158379016211,9083.946
565728915] with 95.0% confidence
Confidence interval for B City_Category group is [9038.02633049294,9090.3063
54092736] with 99.0% confidence

Confidence interval using formula :
Confidence interval for B City_Category group is [9047.473442150711,9080.832
340193201] with 90.0% confidence
Confidence interval for B City_Category group is [9044.278101812004,9084.027
680531908] with 95.0% confidence
Confidence interval for B City_Category group is [9038.032990378351,9090.272
79196556] with 99.0% confidence
Standard Error of B City_Category group is 10.140384984990051
-------------------------
```

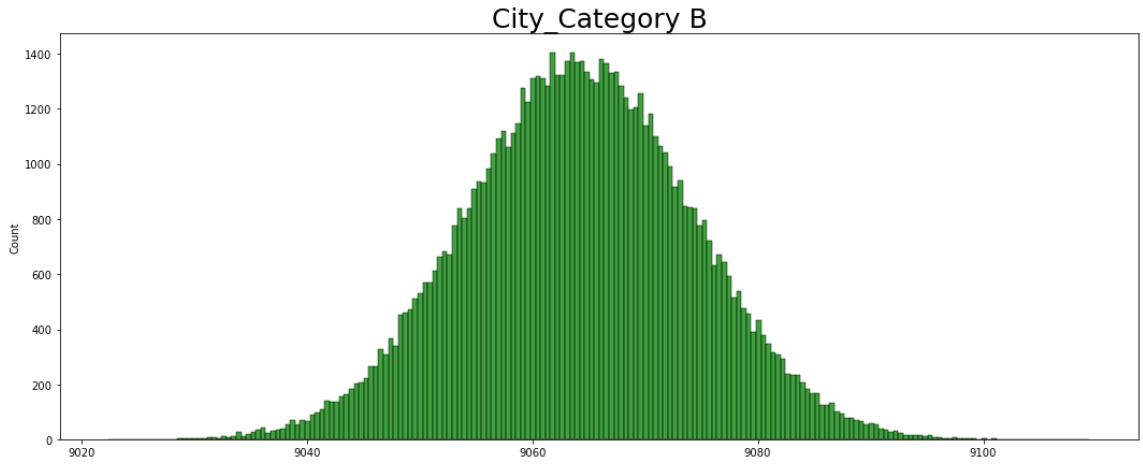
In [57]:

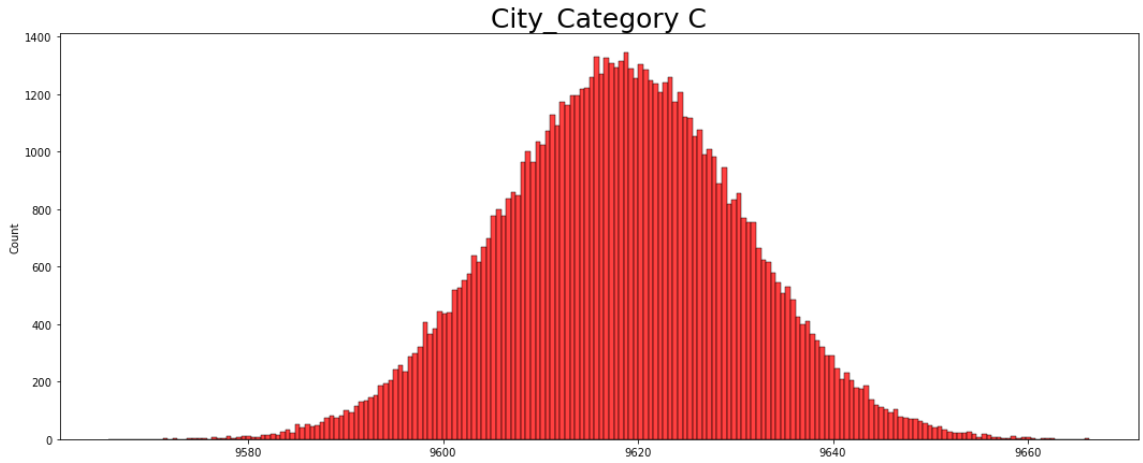
```
data=dff[dff['City_Category']=='A']
ls=[]
for i in range(80000):
    ans=np.random.choice(data['Purchase'],len(data),replace=True)
    l1=np.mean(ans)
    ls.append(l1)
fig, axes = plt.subplots(4, 1, figsize=(18, 32))
sns.histplot(ls,color='b',ax=axes[0],bins=200)
sns.histplot(ls,color='b',ax=axes[3],alpha=0.2,bins=200)

data=dff[dff['City_Category']=='B']
ls=[]
for i in range(80000):
    ans=np.random.choice(data['Purchase'],len(data),replace=True)
    l1=np.mean(ans)
    ls.append(l1)
sns.histplot(ls,color='g',ax=axes[1],bins=200)
sns.histplot(ls,color='g',ax=axes[3],alpha=0.4,bins=200)


data=dff[dff['City_Category']=='C']
ls=[]
for i in range(80000):
    ans=np.random.choice(data['Purchase'],len(data),replace=True)
    l1=np.mean(ans)
    ls.append(l1)
sns.histplot(ls,color='r',ax=axes[2],bins=200)
sns.histplot(ls,color='r',ax=axes[3],alpha=0.6,bins=200)
axes[0].set_title("City_Category A",fontsize=25)
axes[1].set_title("City_Category B",fontsize=25)
axes[2].set_title("City_Category C",fontsize=25)
axes[3].set_title("Combined plot",fontsize=25)
# Hist plot for each City_Category group
# There is no overlap of the confidence interval for city category.
# Walmart will need separate recommendation for each city category
```

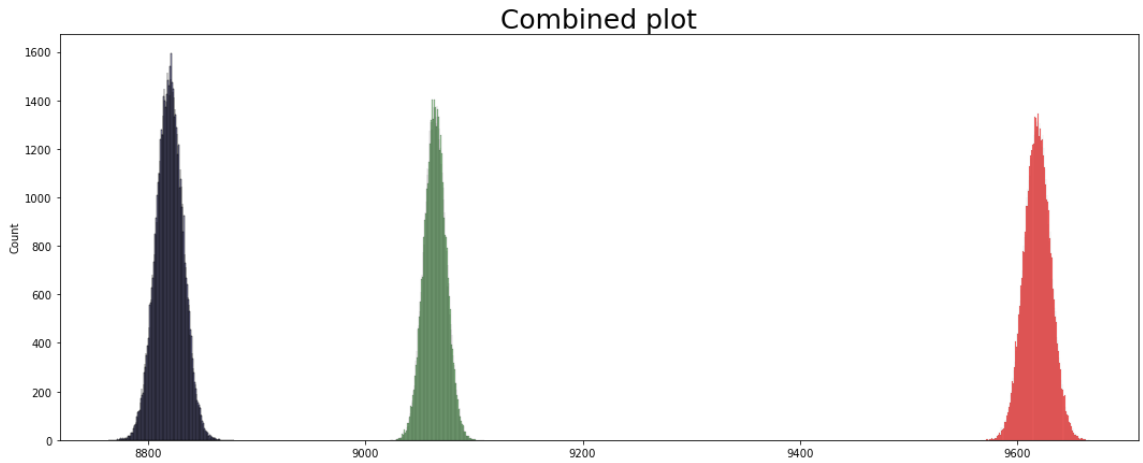
Out[57]:

```
Text(0.5, 1.0, 'Combined plot')
```


City_Category A
Count
0
200
400
600
800
1000
1200
1400
1600
8760
8780
8800
8820
8840
8860
8880


City_Category B
Count
0
200
400
600
800
1000
1200
1400
9020
9040
9060
9080
9100


City_Category C
Count
0
200
400
600
800
1000
1200
1400
9580
9600
9620
9640
9660


Combined plot
Count
0
200
400
600
800
1000
1200
1400
1600
8800
9000
9200
9400
9600

In [58]:

```
conf('Occupation')
# Confidence interval for all the Occupation groups with pure bootstraping and bootstraping
```

```
Confidence interval using Bootstrap :
Confidence interval for 16 Occupation group is [9264.674462727993,9366.74414
9484535] with 90.0% confidence
Confidence interval for 16 Occupation group is [9254.771296590008,9376.19353
9849326] with 95.0% confidence
Confidence interval for 16 Occupation group is [9235.115075337035,9394.38196
2133228] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 16 Occupation group is [9264.560393788619,9366.61175
5436206] with 90.0% confidence
Confidence interval for 16 Occupation group is [9254.785225793648,9376.38692
3431177] with 95.0% confidence
Confidence interval for 16 Occupation group is [9235.680214452042,9395.49193
4772783] with 99.0% confidence
Standard Error of 16 Occupation group is 31.02141125977499
-------------------------
Confidence interval using Bootstrap :
Confidence interval for 15 Occupation group is [9588.90369172059,9737.092695
370831] with 90.0% confidence
Confidence interval for 15 Occupation group is [9574.067705740832,9751.04867
471379] with 95.0% confidence
Confidence interval for 15 Occupation group is [9546.65537580886,9779.224539
986726] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 15 Occupation group is [9588.829429307381,9737.17728
5243804] with 90.0% confidence
Confidence interval for 15 Occupation group is [9574.619670710726,9751.38704
384046] with 95.0% confidence
Confidence interval for 15 Occupation group is [9546.847503593981,9779.15921
0957205] with 99.0% confidence
Standard Error of 15 Occupation group is 45.09454625800542
-------------------------
Confidence interval using Bootstrap :
Confidence interval for 7 Occupation group is [9314.039000221153,9382.098990
354354] with 90.0% confidence
Confidence interval for 7 Occupation group is [9307.333939234133,9388.591128
812752] with 95.0% confidence
Confidence interval for 7 Occupation group is [9294.915739839751,9400.657655
274485] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 7 Occupation group is [9314.140716968861,9382.021547
26825] with 90.0% confidence
Confidence interval for 7 Occupation group is [9307.638633026992,9388.523631
210119] with 95.0% confidence
Confidence interval for 7 Occupation group is [9294.930678798748,9401.231585
438363] with 99.0% confidence
Standard Error of 7 Occupation group is 20.63430726817885
-------------------------
Confidence interval using Bootstrap :
Confidence interval for 20 Occupation group is [8681.373839776517,8767.29522
2432489] with 90.0% confidence
Confidence interval for 20 Occupation group is [8673.161914631582,8775.48943
```

```
6334145] with 95.0% confidence
Confidence interval for 20 Occupation group is [8656.520373974949,8791.43706
1668319] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 20 Occupation group is [8681.152446139511,8767.25687
352196] with 90.0% confidence
Confidence interval for 20 Occupation group is [8672.90478310802,8775.504536
55345] with 95.0% confidence
Confidence interval for 20 Occupation group is [8656.785193568541,8791.62412
609293] with 99.0% confidence
Standard Error of 20 Occupation group is 26.173887442504917
-------------------------
Confidence interval using Bootstrap :
Confidence interval for 1 Occupation group is [8801.09097909089,8872.1431604
02442] with 90.0% confidence
Confidence interval for 1 Occupation group is [8794.739286473954,8878.840990
789782] with 95.0% confidence
Confidence interval for 1 Occupation group is [8780.587418905408,8891.814355
710123] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 1 Occupation group is [8801.006572707916,8872.074982
462993] with 90.0% confidence
Confidence interval for 1 Occupation group is [8794.199160898064,8878.882394
272845] with 95.0% confidence
Confidence interval for 1 Occupation group is [8780.894460671825,8892.187094
499084] with 99.0% confidence
Standard Error of 1 Occupation group is 21.60326262185217
-------------------------
Confidence interval using Bootstrap :
Confidence interval for 12 Occupation group is [9663.37803157487,9757.331519
984504] with 90.0% confidence
Confidence interval for 12 Occupation group is [9654.430138987538,9766.16534
8356688] with 95.0% confidence
Confidence interval for 12 Occupation group is [9636.832761832506,9784.56222
3962033] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 12 Occupation group is [9663.504395013728,9757.43547
6145953] with 90.0% confidence
Confidence interval for 12 Occupation group is [9654.507042276902,9766.43282
8882778] with 95.0% confidence
Confidence interval for 12 Occupation group is [9636.92222673512,9784.017644
42456] with 99.0% confidence
Standard Error of 12 Occupation group is 28.553021251597816
-------------------------
Confidence interval using Bootstrap :
Confidence interval for 17 Occupation group is [9708.277014640516,9791.55692
0971347] with 90.0% confidence
Confidence interval for 17 Occupation group is [9700.311380929661,9799.39218
5028] with 95.0% confidence
Confidence interval for 17 Occupation group is [9684.942181637829,9814.83856
5078726] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 17 Occupation group is [9708.310372074573,9791.43999
3153572] with 90.0% confidence
Confidence interval for 17 Occupation group is [9700.34765606289,9799.402709
165255] with 95.0% confidence
Confidence interval for 17 Occupation group is [9684.784979309765,9814.96538
```

```
591838] with 99.0% confidence
Standard Error of 17 Occupation group is 25.2696105346068
-------------------------
Confidence interval using Bootstrap :
Confidence interval for 0 Occupation group is [8995.080274028096,9055.933657
421472] with 90.0% confidence
Confidence interval for 0 Occupation group is [8988.98652719302,9061.5885563
60952] with 95.0% confidence
Confidence interval for 0 Occupation group is [8977.671678602948,9073.038292
749974] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 0 Occupation group is [8995.110766470092,9055.927770
626067] with 90.0% confidence
Confidence interval for 0 Occupation group is [8989.285303458353,9061.753233
637806] with 95.0% confidence
Confidence interval for 0 Occupation group is [8977.899766456165,9073.138770
639995] with 99.0% confidence
Standard Error of 0 Occupation group is 18.48705658651626
-------------------------
Confidence interval using Bootstrap :
Confidence interval for 10 Occupation group is [8820.036992849371,8966.43183
5846417] with 90.0% confidence
Confidence interval for 10 Occupation group is [8806.135582154515,8980.24588
2558682] with 95.0% confidence
Confidence interval for 10 Occupation group is [8778.172514378983,9006.87895
3054562] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 10 Occupation group is [8819.989349161897,8966.32307
809599] with 90.0% confidence
Confidence interval for 10 Occupation group is [8805.972517239548,8980.33991
0018338] with 95.0% confidence
Confidence interval for 10 Occupation group is [8778.577414363805,9007.73501
2894082] with 99.0% confidence
Standard Error of 10 Occupation group is 44.48229512230289
-------------------------
Confidence interval using Bootstrap :
Confidence interval for 4 Occupation group is [9108.90185471084,9169.8751787
25417] with 90.0% confidence
Confidence interval for 4 Occupation group is [9103.033977996605,9176.001746
223817] with 95.0% confidence
Confidence interval for 4 Occupation group is [9092.315532351387,9187.176482
5169] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 4 Occupation group is [9108.84610823757,9169.8595554
99594] with 90.0% confidence
Confidence interval for 4 Occupation group is [9103.001828579665,9175.703835
1575] with 95.0% confidence
Confidence interval for 4 Occupation group is [9091.579515509931,9187.126148
227233] with 99.0% confidence
Standard Error of 4 Occupation group is 18.54677105071764
-------------------------
Confidence interval using Bootstrap :
Confidence interval for 11 Occupation group is [9019.82106545961,9172.936242
165739] with 90.0% confidence
Confidence interval for 11 Occupation group is [9005.524982590528,9187.52093
2712396] with 95.0% confidence
Confidence interval for 11 Occupation group is [8976.079623084957,9217.69824
1643455] with 99.0% confidence
```

```
Confidence interval using formula :
Confidence interval for 11 Occupation group is [9019.804948912797,9172.80564
5890474] with 90.0% confidence
Confidence interval for 11 Occupation group is [9005.149509806859,9187.46108
4996412] with 95.0% confidence
Confidence interval for 11 Occupation group is [8976.506285416317,9216.10430
9386954] with 99.0% confidence
Standard Error of 11 Occupation group is 46.50890950742067
-------------------------


Confidence interval using Bootstrap :
Confidence interval for 3 Occupation group is [8993.254354045084,9114.211262
730289] with 90.0% confidence
Confidence interval for 3 Occupation group is [8982.194996567114,9125.952231
376588] with 95.0% confidence
Confidence interval for 3 Occupation group is [8960.3129311134,9149.16063508
4105] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 3 Occupation group is [8993.437668854516,9114.280606
893572] with 90.0% confidence
Confidence interval for 3 Occupation group is [8981.862516879408,9125.855758
86868] with 95.0% confidence
Confidence interval for 3 Occupation group is [8959.239538954216,9148.478736
793872] with 99.0% confidence
Standard Error of 3 Occupation group is 36.73364488456727
-------------------------
Confidence interval using Bootstrap :
Confidence interval for 8 Occupation group is [9279.048181818182,9682.974967
532467] with 90.0% confidence
Confidence interval for 8 Occupation group is [9239.890016233765,9724.112094
155844] with 95.0% confidence
Confidence interval for 8 Occupation group is [9164.205652597402,9801.007172
077921] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 8 Occupation group is [9278.590431956118,9683.383395
641287] with 90.0% confidence
Confidence interval for 8 Occupation group is [9239.816630614116,9722.157196
98329] with 95.0% confidence
Confidence interval for 8 Occupation group is [9164.035435367416,9797.938392
22999] with 99.0% confidence
Standard Error of 8 Occupation group is 123.04832389110562
-------------------------
Confidence interval using Bootstrap :
Confidence interval for 19 Occupation group is [8516.066819535437,8693.25532
4597976] with 90.0% confidence
Confidence interval for 19 Occupation group is [8499.453653960692,8709.95490
4705182] with 95.0% confidence
Confidence interval for 19 Occupation group is [8467.749901131625,8745.12714
353782] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 19 Occupation group is [8515.981206641482,8693.13104
1056553] with 90.0% confidence
Confidence interval for 19 Occupation group is [8499.012600209082,8710.09964
7488953] with 95.0% confidence
Confidence interval for 19 Occupation group is [8465.848421495317,8743.26382
6202718] with 99.0% confidence
```

```
Standard Error of 19 Occupation group is 53.849726052340756
-------------------------
Confidence interval using Bootstrap :
Confidence interval for 2 Occupation group is [8787.05378983308,8884.8094802
73141] with 90.0% confidence
Confidence interval for 2 Occupation group is [8777.884578907435,8894.083059
55994] with 95.0% confidence
Confidence interval for 2 Occupation group is [8760.52136494689,8913.2588909
33233] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 2 Occupation group is [8787.433501372076,8884.842895
321775] with 90.0% confidence
Confidence interval for 2 Occupation group is [8778.102972364195,8894.173424
329656] with 95.0% confidence
Confidence interval for 2 Occupation group is [8759.866982700338,8912.409413
993513] with 99.0% confidence
Standard Error of 2 Occupation group is 29.610353272052386
-------------------------
Confidence interval using Bootstrap :
Confidence interval for 18 Occupation group is [9017.543345460057,9217.30878
4295892] with 90.0% confidence
Confidence interval for 18 Occupation group is [8998.817625435804,9236.27766
408974] with 95.0% confidence
Confidence interval for 18 Occupation group is [8960.861572684553,9273.75842
0494165] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 18 Occupation group is [9018.12115054722,9217.510542
57086] with 90.0% confidence
Confidence interval for 18 Occupation group is [8999.022289228968,9236.60940
3889113] with 95.0% confidence
Confidence interval for 18 Occupation group is [8961.694648181294,9273.93704
4936787] with 99.0% confidence
Standard Error of 18 Occupation group is 60.61007154575354
-------------------------
Confidence interval using Bootstrap :
Confidence interval for 14 Occupation group is [9365.661069954298,9464.97428
4977148] with 90.0% confidence
Confidence interval for 14 Occupation group is [9356.02827104526,9474.562895
289695] with 95.0% confidence
Confidence interval for 14 Occupation group is [9337.053105189445,9492.71772
4642489] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 14 Occupation group is [9365.524568671719,9464.79395
5917326] with 90.0% confidence
Confidence interval for 14 Occupation group is [9356.015876955928,9474.30264
7633118] with 95.0% confidence
Confidence interval for 14 Occupation group is [9337.431678385889,9492.88684
6203157] with 99.0% confidence
Standard Error of 14 Occupation group is 30.17575108783137
-------------------------
Confidence interval using Bootstrap :
Confidence interval for 13 Occupation group is [9091.58184251763,9271.206960
041787] with 90.0% confidence
Confidence interval for 13 Occupation group is [9074.614360799165,9288.81540
5458345] with 95.0% confidence
Confidence interval for 13 Occupation group is [9040.093734003656,9322.11343
6928703] with 99.0% confidence
```

```
Confidence interval using formula :
Confidence interval for 13 Occupation group is [9091.33718270354,9271.169495
126833] with 90.0% confidence
Confidence interval for 13 Occupation group is [9074.111630427893,9288.39504
740248] with 95.0% confidence
Confidence interval for 13 Occupation group is [9040.445265638968,9322.06141
2191406] with 99.0% confidence
Standard Error of 13 Occupation group is 54.66514146811561
-------------------------
Confidence interval using Bootstrap :
Confidence interval for 9 Occupation group is [8426.549903738169,8613.729945
451629] with 90.0% confidence
Confidence interval for 9 Occupation group is [8408.14667896679,8631.7422188
35232] with 95.0% confidence
Confidence interval for 9 Occupation group is [8375.166453553666,8666.238844
055832] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 9 Occupation group is [8426.465142222285,8613.637890
62707] with 90.0% confidence
Confidence interval for 9 Occupation group is [8408.53647345983,8631.5665593
89525] with 95.0% confidence
Confidence interval for 9 Occupation group is [8373.495907382312,8666.607125
467042] with 99.0% confidence
Standard Error of 9 Occupation group is 56.89647557019612
-------------------------
Confidence interval using Bootstrap :
Confidence interval for 6 Occupation group is [9074.939454752926,9187.715630
581464] with 90.0% confidence
Confidence interval for 6 Occupation group is [9064.169920867236,9198.954583
00258] with 95.0% confidence
Confidence interval for 6 Occupation group is [9042.421610190515,9220.470589
154595] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 6 Occupation group is [9074.98982938325,9187.6895234
571] with 90.0% confidence
Confidence interval for 6 Occupation group is [9064.194692260455,9198.484660
579896] with 95.0% confidence
Confidence interval for 6 Occupation group is [9043.096209117115,9219.583143
723235] with 99.0% confidence
Standard Error of 6 Occupation group is 34.25827448328222
-------------------------
Confidence interval using Bootstrap :
Confidence interval for 5 Occupation group is [9194.13091936948,9342.4705743
99604] with 90.0% confidence
Confidence interval for 5 Occupation group is [9180.594804819675,9356.832462
655773] with 95.0% confidence
Confidence interval for 5 Occupation group is [9151.60986135182,9383.8251885
78032] with 99.0% confidence

Confidence interval using formula :
Confidence interval for 5 Occupation group is [9194.89150608211,9342.6064533
24093] with 90.0% confidence
Confidence interval for 5 Occupation group is [9180.742371750666,9356.755587
655536] with 95.0% confidence
Confidence interval for 5 Occupation group is [9153.088691321313,9384.409268
08489] with 99.0% confidence
Standard Error of 5 Occupation group is 44.902155675624975
-------------------------
```

In [59]:

```
vals=dff["Occupation"].unique()
vals
```

Out[59]:

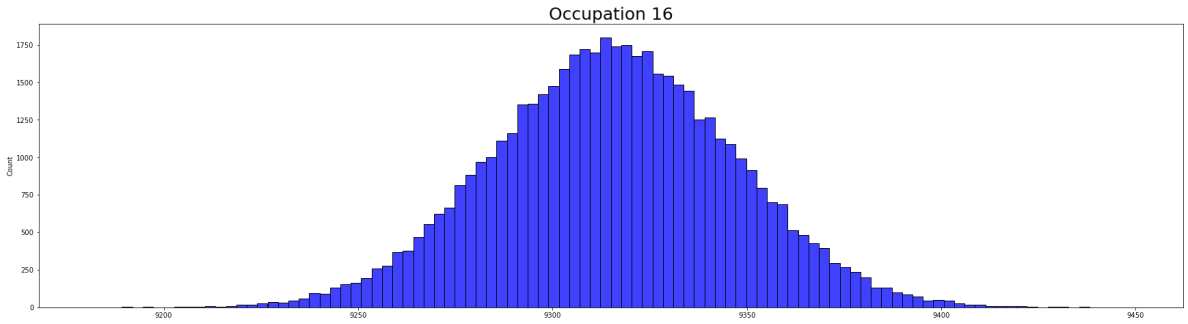
```
array([16, 15,  7, 20,  1, 12, 17,  0, 10,  4, 11,  3,  8, 19,  2, 18, 14,
       13,  9,  6,  5], dtype=int64)
```

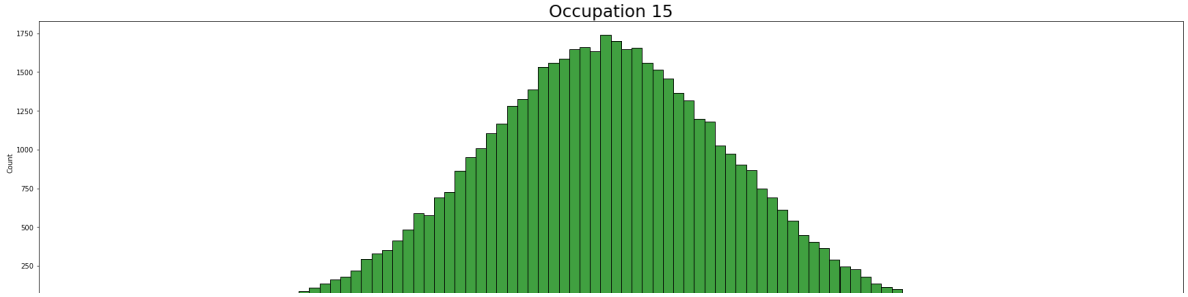
In [60]:

```
fig, axes = plt.subplots(22, 1, figsize=(30, 200))
cols=['b','g','r','c','m','y','k','b','g','r','c','m','y','k','b','g','r','c','m','y','k','
for i in range(22):
    if i==21:
        break
    data=dff[dff['Occupation']==vals[i]]
    ls=[]
    for j in range(50000):
        ans=np.random.choice(data['Purchase'],len(data),replace=True)
        l1=np.mean(ans)
        ls.append(l1)
    sns.histplot(ls,color=cols[i],ax=axes[i],bins=100)
    sns.histplot(ls,color=cols[i],ax=axes[21],alpha=0.2,bins=100)
axes[0].set_title("Occupation 16",fontsize=25)
axes[1].set_title("Occupation 15",fontsize=25)
axes[2].set_title("Occupation 7",fontsize=25)
axes[3].set_title("Occupation 20",fontsize=25)
axes[4].set_title("Occupation 1",fontsize=25)
axes[5].set_title("Occupation 12",fontsize=25)
axes[6].set_title("Occupation 17",fontsize=25)
axes[7].set_title("Occupation 0",fontsize=25)
axes[8].set_title("Occupation 10",fontsize=25)
axes[9].set_title("Occupation 4",fontsize=25)
axes[10].set_title("Occupation 11",fontsize=25)
axes[11].set_title("Occupation 3",fontsize=25)
axes[12].set_title("Occupation 8",fontsize=25)
axes[13].set_title("Occupation 19",fontsize=25)
axes[14].set_title("Occupation 2",fontsize=25)
axes[15].set_title("Occupation 18",fontsize=25)
axes[16].set_title("Occupation 14",fontsize=25)
axes[17].set_title("Occupation 13",fontsize=25)
axes[18].set_title("Occupation 9",fontsize=25)
axes[19].set_title("Occupation 6",fontsize=25)
axes[20].set_title("Occupation 5",fontsize=25)
axes[21].set_title("Combined plot",fontsize=25)
# Hist plot for each Occupation group.
```

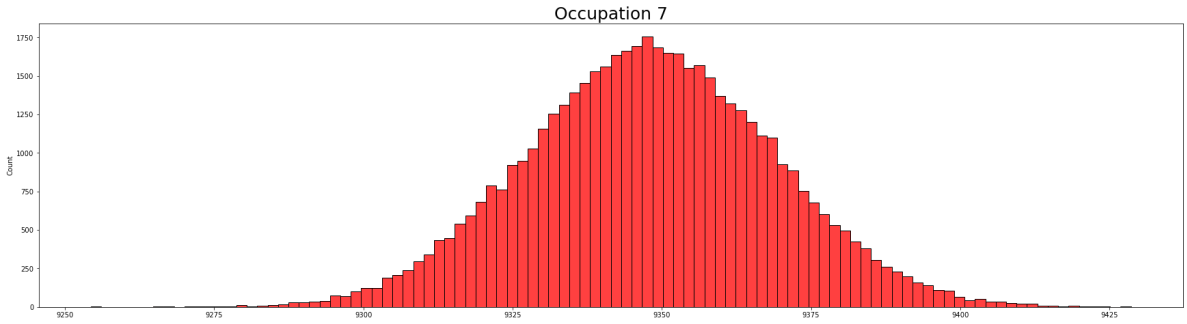
Out[60]:

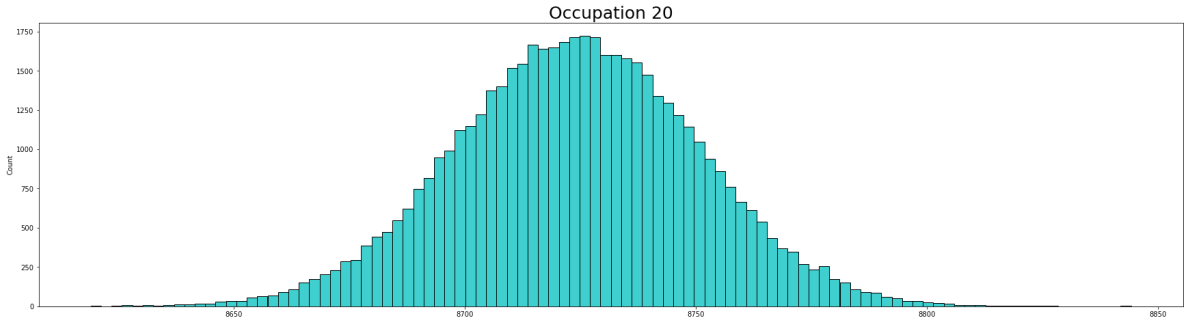
```
Text(0.5, 1.0, 'Combined plot')
```

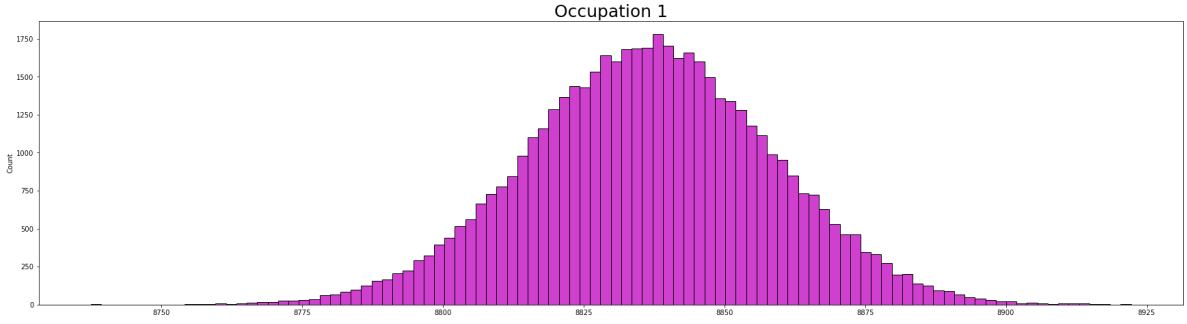


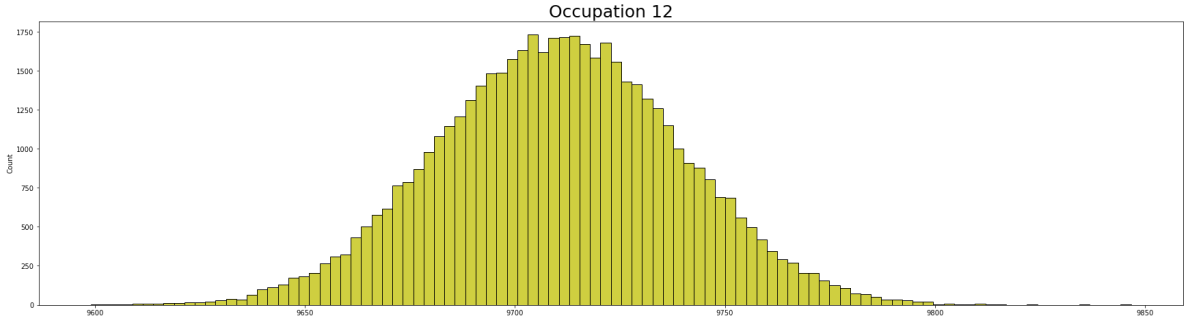


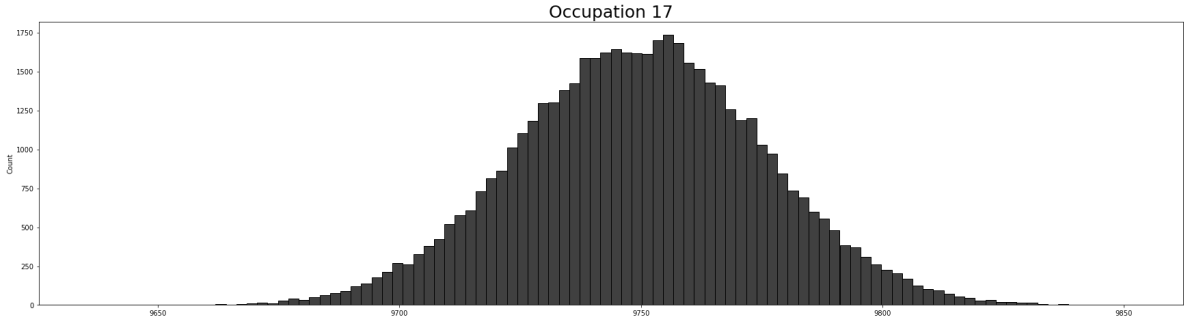

0
9500
9550
9600
9650
9700
9750
9800
9850

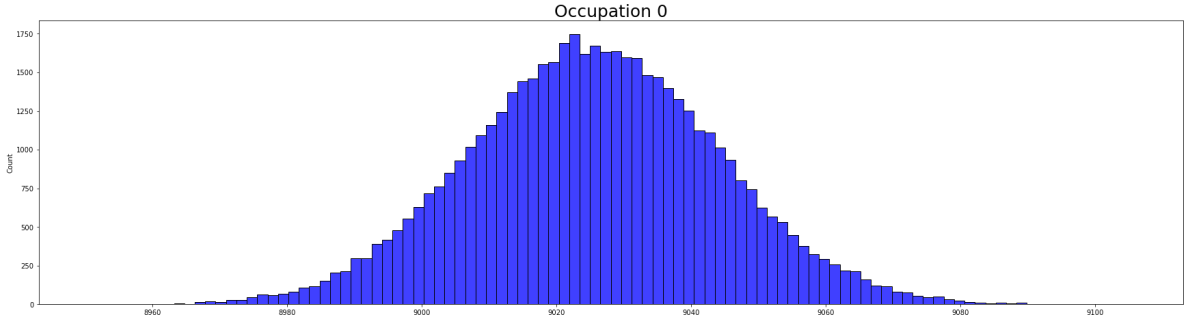

Occupation 7
Count
0
250
500
750
1000
1250
1500
1750
9250
9275
9300
9325
9350
9375
9400
9425

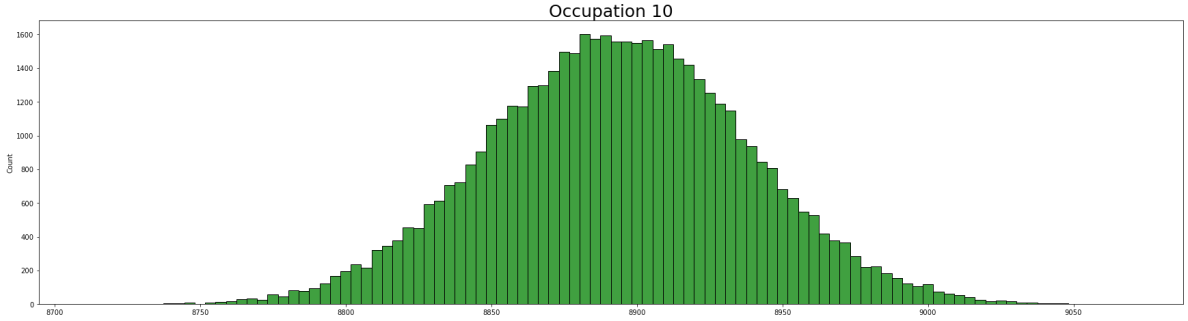

Occupation 20
Count
0
250
500
750
1000
1250
1500
1750
8650
8700
8750
8800
8850

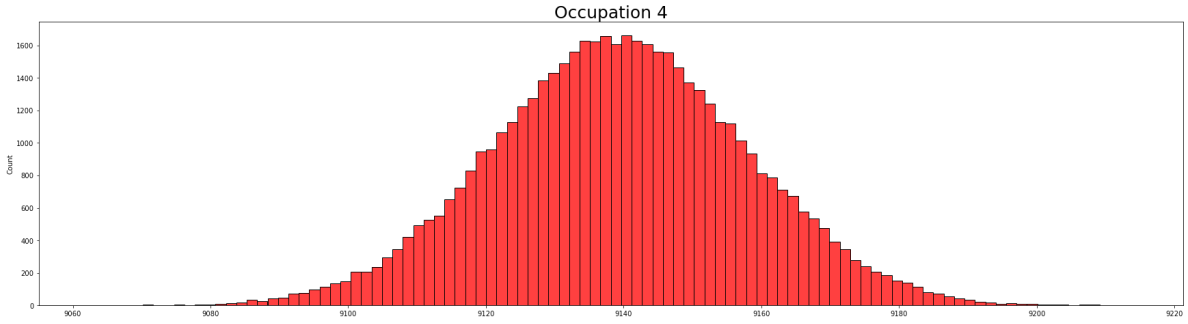

Occupation 1
Count
0
250
500
750
1000
1250
1500
1750
8750
8775
8800
8825
8850
8875
8900
8925

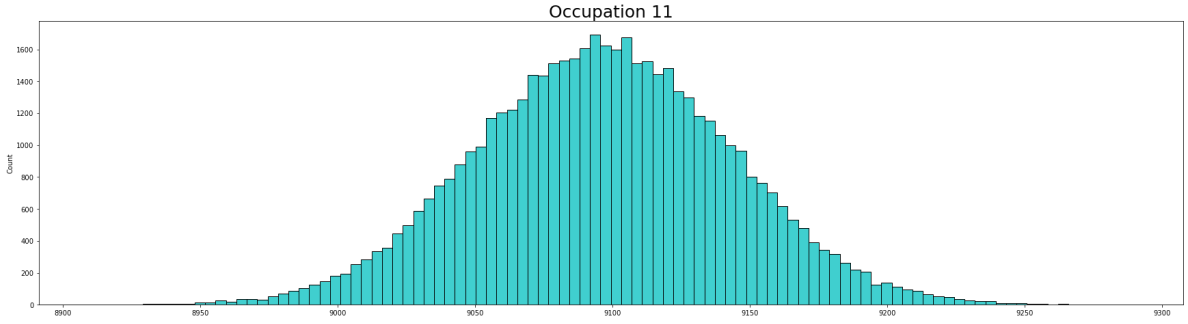

Occupation 12
Count
0
250
500
750
1000
1250
1500
1750
9600
9650
9700
9750
9800
9850

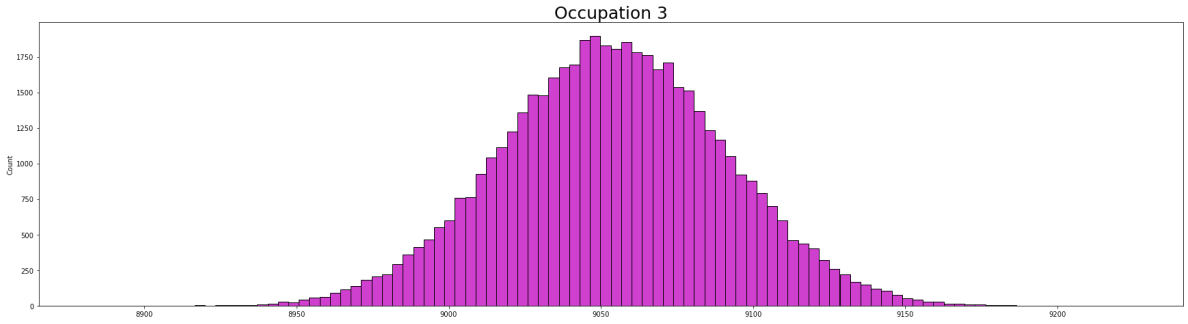

Occupation 17
Count
0
250
500
750
1000
1250
1500
1750
9650
9700
9750
9800
9850

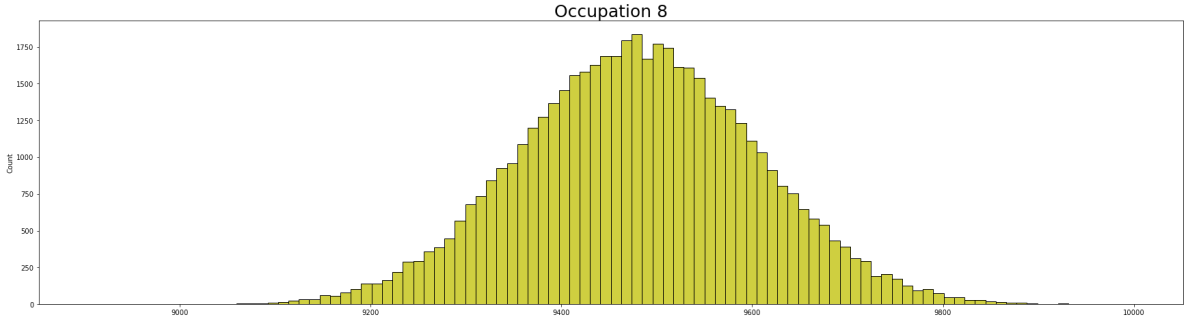

Occupation 0
Count
0
250
500
750
1000
1250
1500
1750
8960
8980
9000
9020
9040
9060
9080
9100

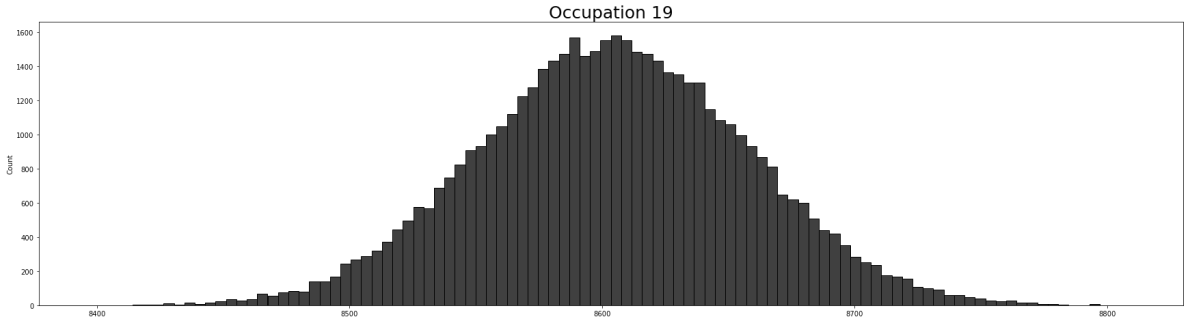

Occupation 10
Count

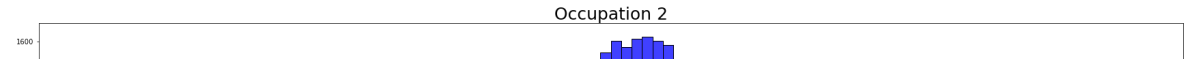

Occupation 4
Count

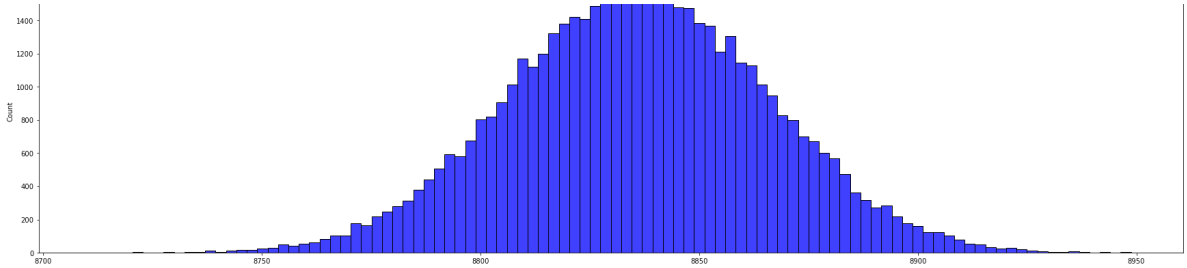

Occupation 11
Count


Occupation 3
Count

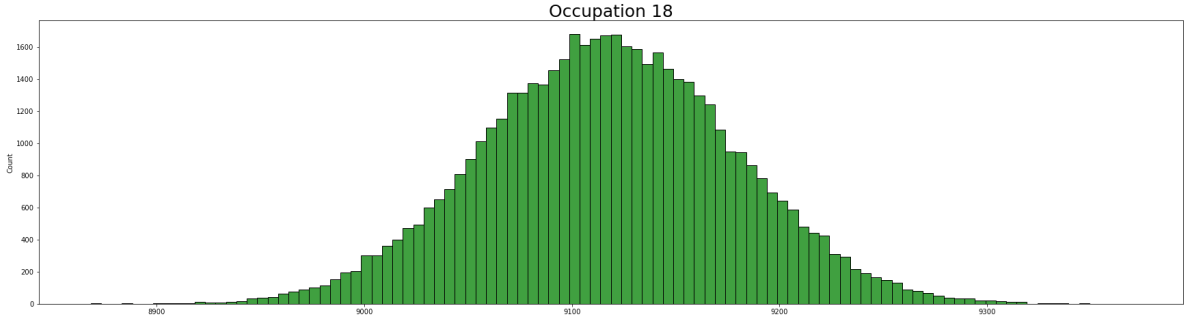

Occupation 8
Count


Occupation 19
Count

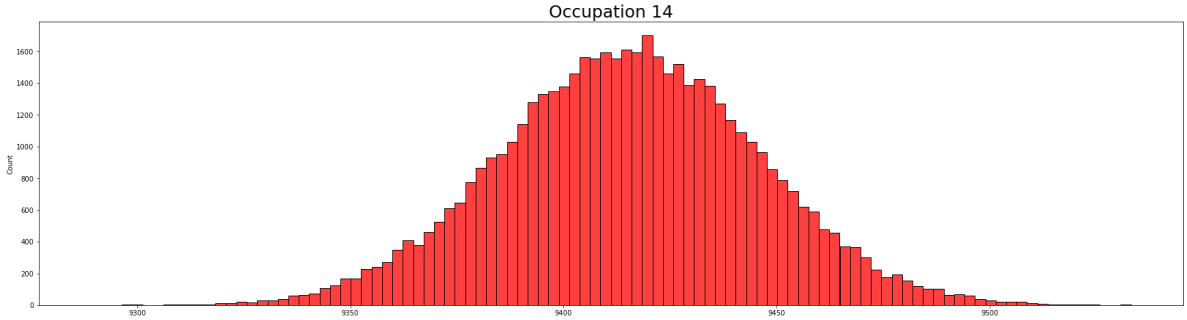

Occupation 2



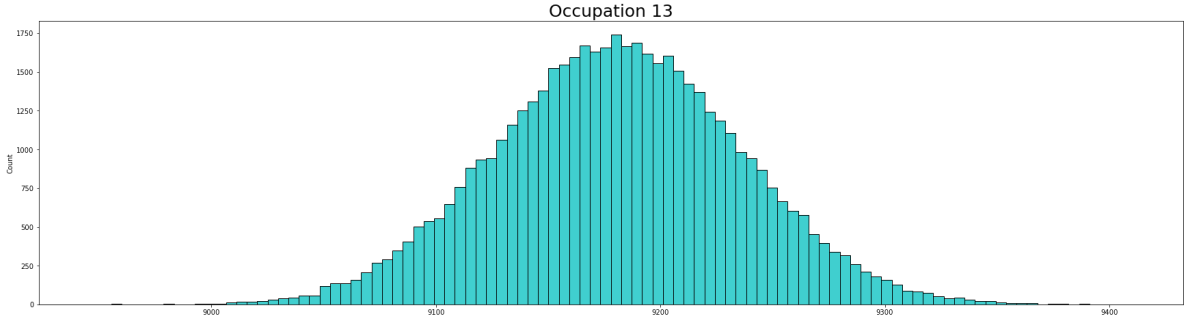
Occupation 18



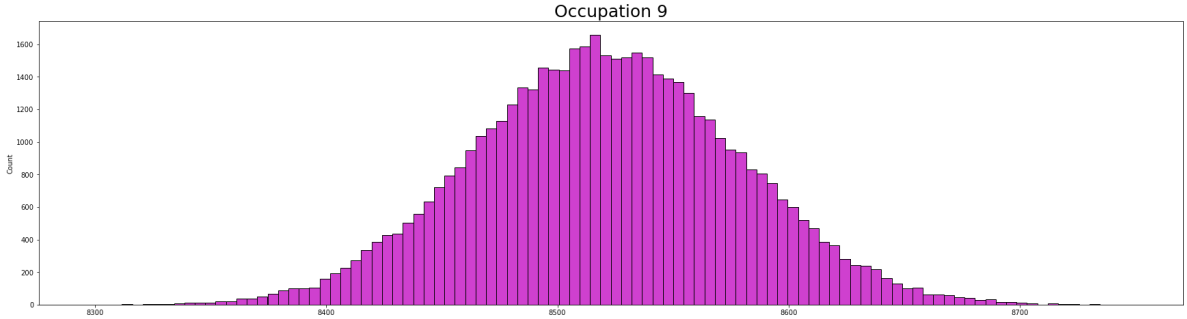
Occupation 14



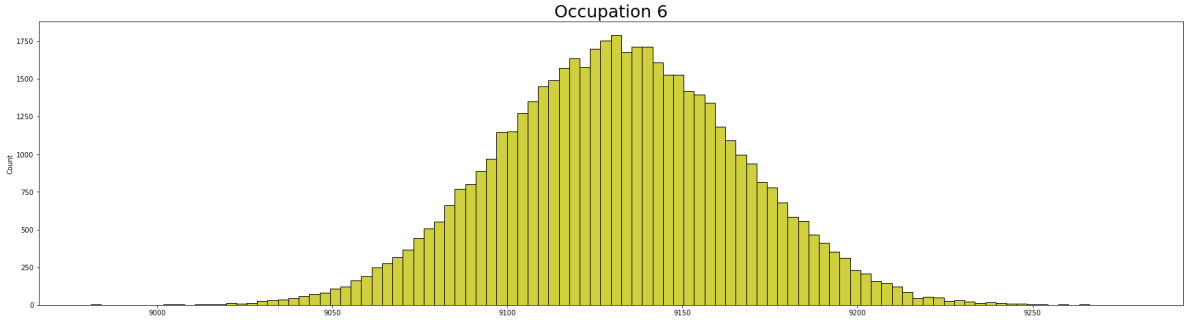
Occupation 13



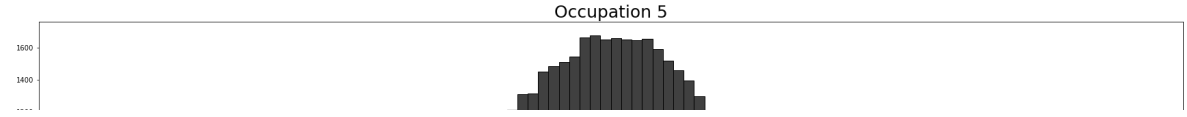
Occupation 9

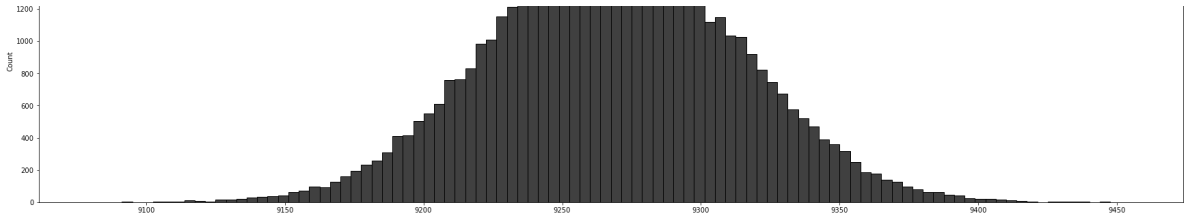


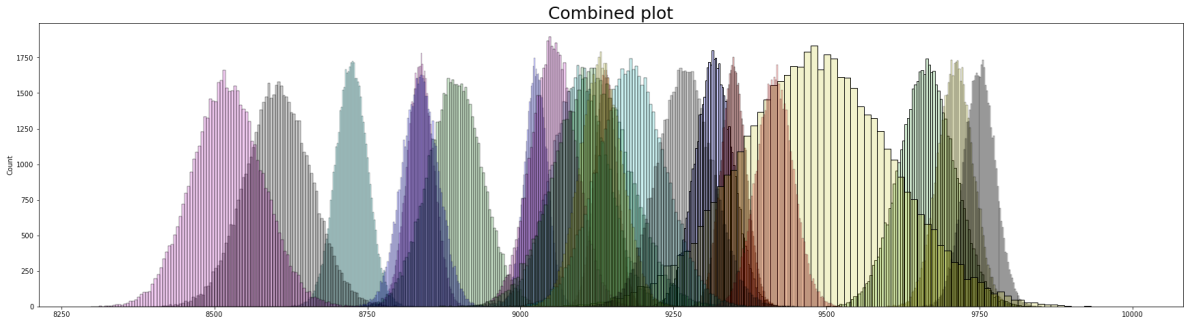
Occupation 6



Occupation 5




Count
1200
1000
800
600
400
200
0
9100
9150
9200
9250
9300
9350
9400
9450


Combined plot
Count
1750
1500
1250
1000
750
500
250
0
8250
8500
8750
9000
9250
9500
9750
10000

In [ ]: