In [1]:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pylab as plt
```

executed in 31.3s, finished 21:59:40 2019-07-09

In [2]:

```
train = pd.read_csv('train mod.csv')
test = pd.read_csv('test mod.csv')
```

executed in 473ms, finished 21:59:40 2019-07-09

In [3]:

```
train.isnull().sum()
```

executed in 62ms, finished 21:59:41 2019-07-09

Out[3]:

```
ID                                  0
Q1                                  0
Q2                                  0
Q3                                  0
Q4                                  0
Q5                                  0
Q6                                  0
Q7                                  0
Q8_1                                0
Q8_2                                0
Q8_3                                0
Q8_4                                0
Q8_5                                0
Q8_6                                0
Q8_7                                0
Q8_8                                0
Q8_9                                0
Q8_10                               0
Q8_11                               0
Q9                                  0
Q10                                 0
Q11                                 0
Q12                                 0
Q13                                 0
Q14                                 0
Q15                                 0
Q16                                 0
Q17                                 0
Q18                                 0
Q19                                 0
Latitude                            0
Longitude                           0
mobile_money                        0
savings                             0
borrowing                           0
insurance                           0
mobile_money_classification         0
region                            143
dtype: int64
```

In [4]:

```
train['region']=train['region'].fillna('0')
```

executed in 203ms, finished 21:59:41 2019-07-09

In [5]:

```
train.describe().T
```

executed in 645ms, finished 21:59:41 2019-07-09

Out[5]:

	count	mean	std	min	25%	
ID	7094.0	4742.627291	2731.120086	1.000000	2397.250000	4744.50
Q1	7094.0	38.239498	16.332148	16.000000	25.000000	35.00
Q2	7094.0	1.559910	0.496433	1.000000	1.000000	2.00
Q3	7094.0	1.787426	1.165160	1.000000	1.000000	1.00
Q4	7094.0	3.060051	1.557779	1.000000	2.000000	3.00
Q5	7094.0	2.548915	1.534257	1.000000	1.000000	3.00
Q6	7094.0	1.840569	0.366103	1.000000	2.000000	2.00
Q7	7094.0	1.397942	0.489508	1.000000	1.000000	1.00
Q8_1	7094.0	0.062165	0.241472	0.000000	0.000000	0.00
Q8_2	7094.0	0.630110	0.482809	0.000000	0.000000	1.00
Q8_3	7094.0	0.058077	0.233906	0.000000	0.000000	0.00
Q8_4	7094.0	0.337327	0.472831	0.000000	0.000000	0.00
Q8_5	7094.0	0.009445	0.096730	0.000000	0.000000	0.00
Q8_6	7094.0	0.004793	0.069069	0.000000	0.000000	0.00
Q8_7	7094.0	0.004793	0.069069	0.000000	0.000000	0.00
Q8_8	7094.0	0.018466	0.134640	0.000000	0.000000	0.00
Q8_9	7094.0	0.156752	0.363593	0.000000	0.000000	0.00
Q8_10	7094.0	0.057795	0.233372	0.000000	0.000000	0.00
Q8_11	7094.0	0.001269	0.035598	0.000000	0.000000	0.00
Q9	7094.0	-0.794615	0.895007	-1.000000	-1.000000	-1.00
Q10	7094.0	0.876092	2.172787	-1.000000	-1.000000	1.00
Q11	7094.0	-0.692134	1.411600	-1.000000	-1.000000	-1.00
Q12	7094.0	1.700733	0.457969	1.000000	1.000000	2.00
Q13	7094.0	0.407668	2.281322	-1.000000	-1.000000	-1.00
Q14	7094.0	1.622639	0.484761	1.000000	1.000000	2.00
Q15	7094.0	0.761066	2.420599	-1.000000	-1.000000	-1.00
Q16	7094.0	1.951508	1.580819	1.000000	1.000000	1.00
Q17	7094.0	-0.431914	1.489879	-1.000000	-1.000000	-1.00
Q18	7094.0	1.860164	1.351372	1.000000	1.000000	1.00
Q19	7094.0	3.163378	1.317691	1.000000	2.000000	4.00
Latitude	7094.0	-6.034378	2.720888	-11.467463	-8.275387	-6.08
Longitude	7094.0	35.354029	2.899511	29.639578	32.935429	35.07
mobile_money	7094.0	0.553989	0.497112	0.000000	0.000000	1.00
savings	7094.0	0.461517	0.498552	0.000000	0.000000	0.00

	count	mean	std	min	25%	
borrowing	7094.0	0.432901	0.495512	0.000000	0.000000	0.00
insurance	7094.0	0.151255	0.358322	0.000000	0.000000	0.00
mobile_money_classification	7094.0	1.799267	1.196955	0.000000	1.000000	2.00

In [6]:

```
train.drop('ID',axis=1).duplicated().sum()
```

executed in 78ms, finished 21:59:41 2019-07-09

Out[6]:

```
0
```

In [7]:

```
train.info()
```

executed in 99ms, finished 21:59:42 2019-07-09

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7094 entries, 0 to 7093
Data columns (total 38 columns):
ID                             7094 non-null int64
Q1                             7094 non-null int64
Q2                             7094 non-null int64
Q3                             7094 non-null int64
Q4                             7094 non-null int64
Q5                             7094 non-null int64
Q6                             7094 non-null int64
Q7                             7094 non-null int64
Q8_1                           7094 non-null int64
Q8_2                           7094 non-null int64
Q8_3                           7094 non-null int64
Q8_4                           7094 non-null int64
Q8_5                           7094 non-null int64
Q8_6                           7094 non-null int64
Q8_7                           7094 non-null int64
Q8_8                           7094 non-null int64
Q8_9                           7094 non-null int64
Q8_10                          7094 non-null int64
Q8_11                          7094 non-null int64
Q9                             7094 non-null int64
Q10                            7094 non-null int64
Q11                            7094 non-null int64
Q12                            7094 non-null int64
Q13                            7094 non-null int64
Q14                            7094 non-null int64
Q15                            7094 non-null int64
Q16                            7094 non-null int64
Q17                            7094 non-null int64
Q18                            7094 non-null int64
Q19                            7094 non-null int64
Latitude                       7094 non-null float64
Longitude                      7094 non-null float64
mobile_money                   7094 non-null int64
savings                        7094 non-null int64
borrowing                      7094 non-null int64
insurance                      7094 non-null int64
mobile_money_classification    7094 non-null int64
region                         7094 non-null object
dtypes: float64(2), int64(35), object(1)
memory usage: 2.1+ MB
```

In [8]:

```
train.nunique()
```

executed in 422ms, finished 21:59:42 2019-07-09

Out[8]:

```
ID                             7094
Q1                               85
Q2                                2
Q3                                4
Q4                                8
Q5                                6
Q6                                2
Q7                                2
Q8_1                              2
Q8_2                              2
Q8_3                              2
Q8_4                              2
Q8_5                              2
Q8_6                              2
Q8_7                              2
Q8_8                              2
Q8_9                              2
Q8_10                             2
Q8_11                             2
Q9                                7
Q10                              11
Q11                              11
Q12                               2
Q13                               7
Q14                               2
Q15                               7
Q16                               5
Q17                               6
Q18                               5
Q19                               5
Latitude                       7056
Longitude                      7055
mobile_money                      2
savings                           2
borrowing                         2
insurance                         2
mobile_money_classification       4
region                           36
dtype: int64
```

In [9]:

```
uninformative=['ID','mobile_money', 'savings', 'borrowing', 'insurance']
un=['ID']
```

executed in 30ms, finished 21:59:42 2019-07-09

In [10]:

```
train.drop(uninformative,axis=1,inplace=True)
test.drop(un,axis=1,inplace=True)
```

executed in 140ms, finished 21:59:42 2019-07-09

In [11]:

```
cat_col=[col for col in train.columns if train[col].nunique()<40]
num_col=list(set(list(train.columns))-set(cat_col))
```

executed in 219ms, finished 21:59:42 2019-07-09

In [12]:

```
print('Categorical features are:',cat_col)
print('')
print('Numerical features are:',num_col)
```

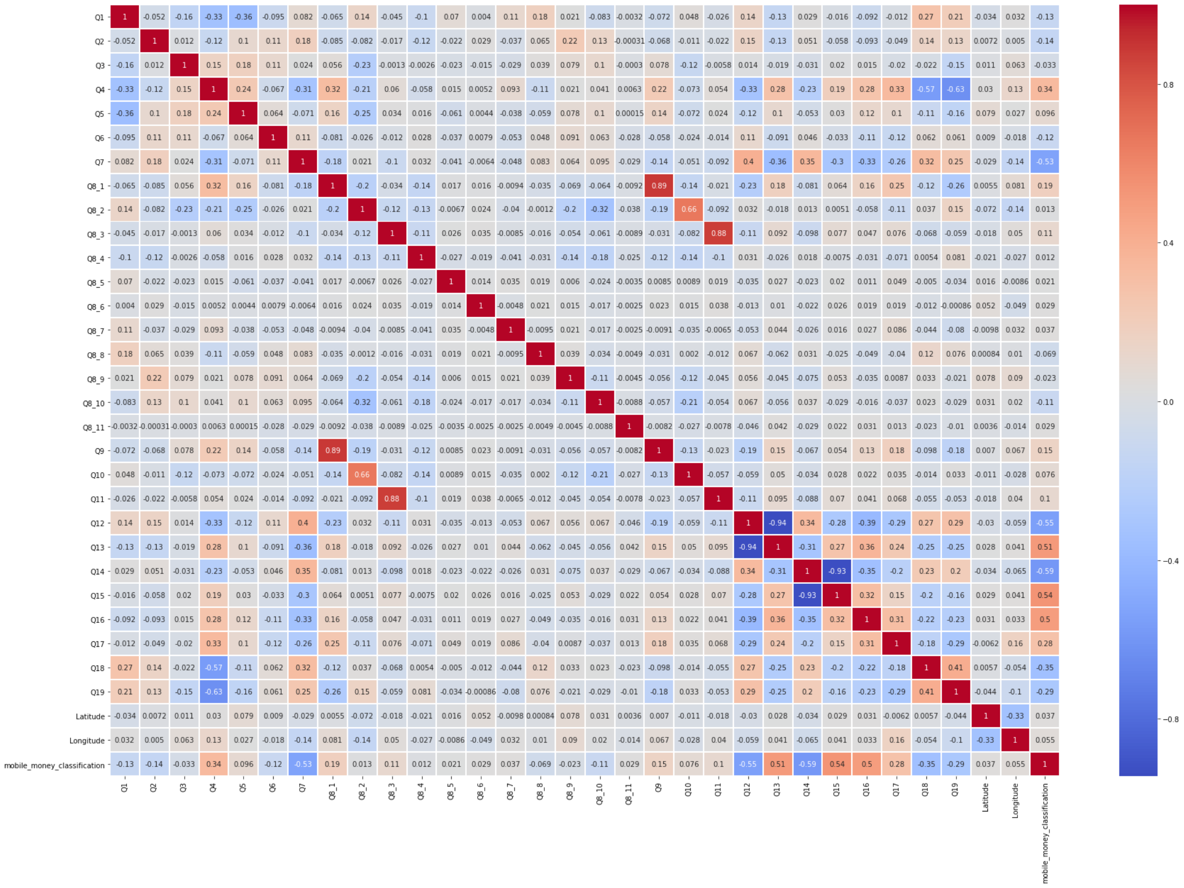
executed in 118ms, finished 21:59:43 2019-07-09

```
Categorical features are: ['Q2', 'Q3', 'Q4', 'Q5', 'Q6', 'Q7', 'Q8_1', 'Q8_
2', 'Q8_3', 'Q8_4', 'Q8_5', 'Q8_6', 'Q8_7', 'Q8_8', 'Q8_9', 'Q8_10', 'Q8_1
1', 'Q9', 'Q10', 'Q11', 'Q12', 'Q13', 'Q14', 'Q15', 'Q16', 'Q17', 'Q18', 'Q1
9', 'mobile_money_classification', 'region']

Numerical features are: ['Q1', 'Longitude', 'Latitude']
```

In [13]:

```
plt.figure(figsize=(30,20))
sns.heatmap(train.corr(),annot=True,linewidths=1, linecolor='white',cmap='coolwarm')
plt.show()
```

executed in 10.6s, finished 21:59:53 2019-07-09



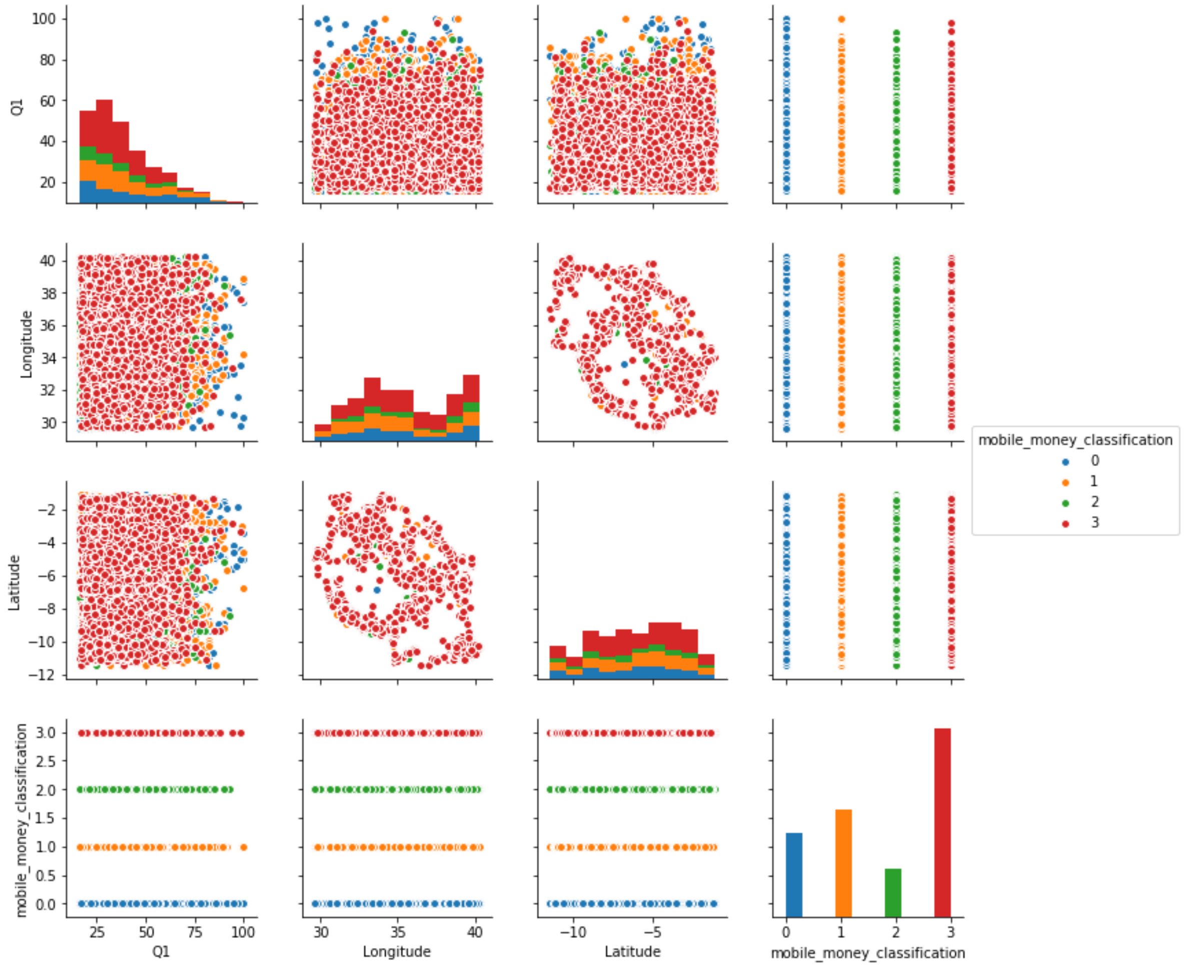
In [14]:

```
sc=num_col + ['mobile_money_classification']
sns.pairplot(train[sc],hue='mobile_money_classification',)
```

executed in 12.0s, finished 22:00:05 2019-07-09

Out[14]:

```
<seaborn.axisgrid.PairGrid at 0x1feb6708b38>
```

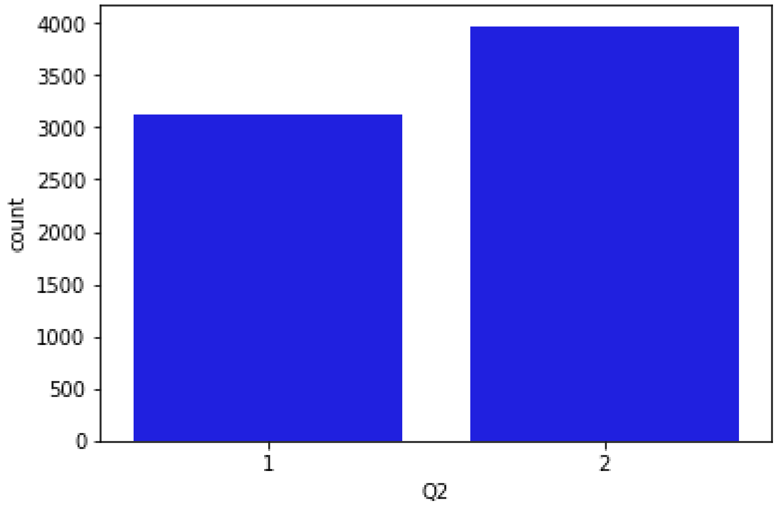


In [15]:

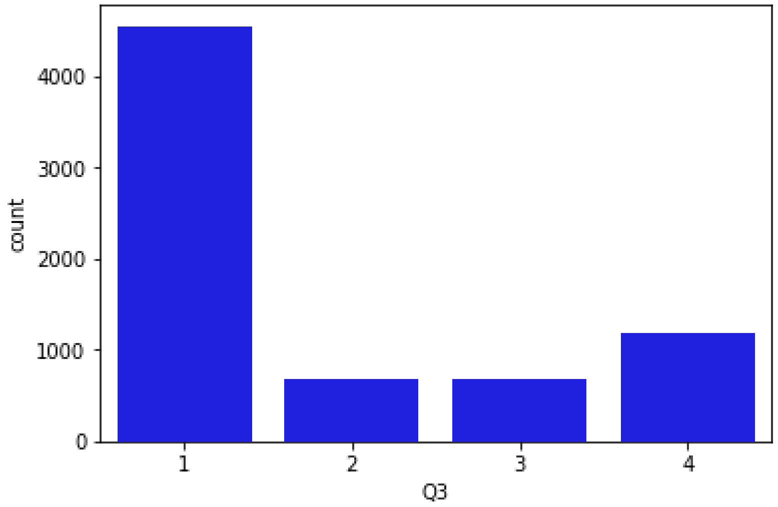
```
for col in cat_col:
    print(col)
    sns.countplot(data=train,x=col,color='blue')
    plt.show()
```

executed in 10.2s, finished 22:00:15 2019-07-09

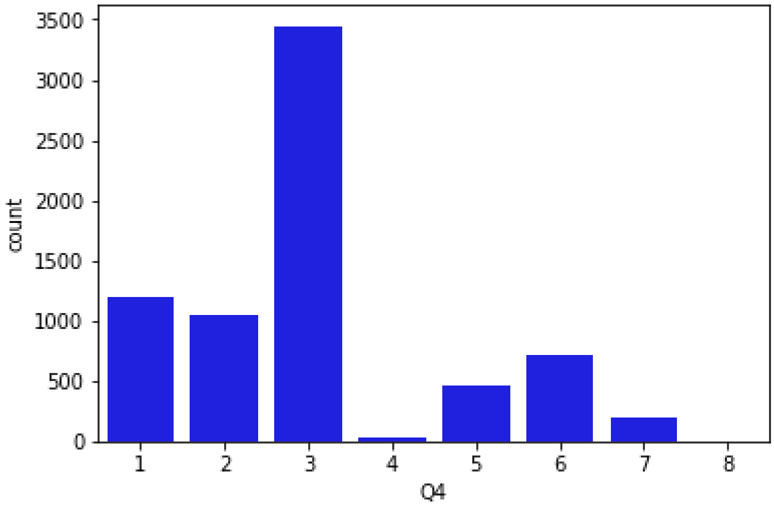
```
Q2
```



```
Q3
```

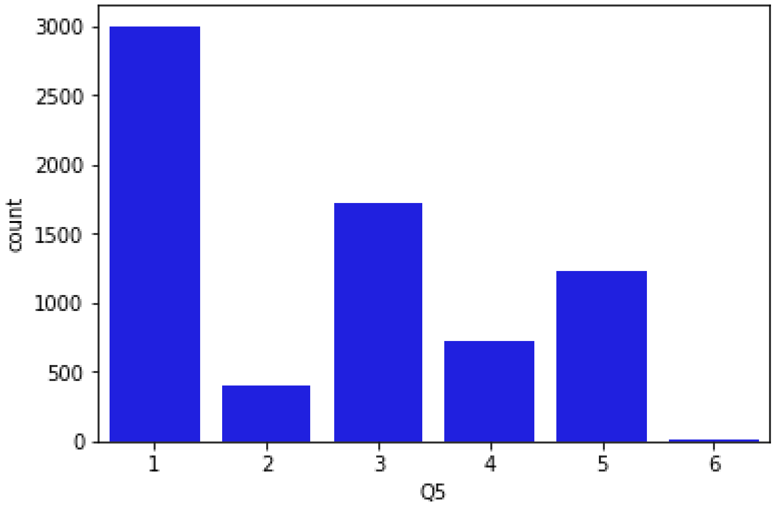


```
Q4
```



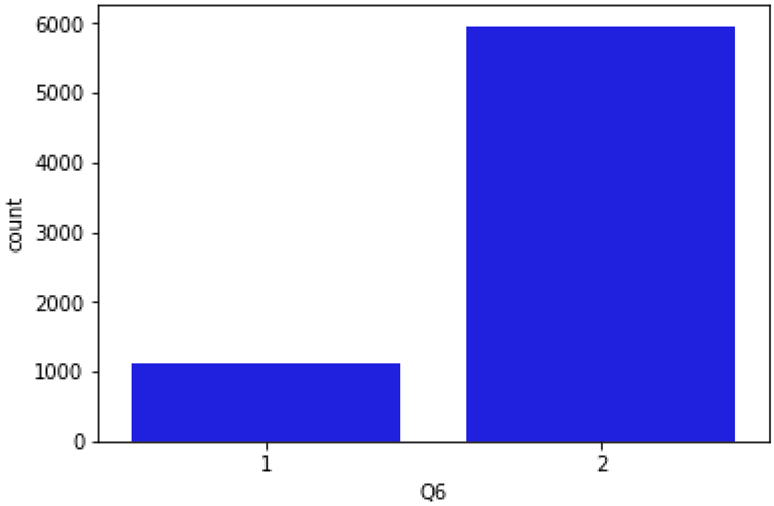
count
3500
3000
2500
2000
1500
1000
500
0
1
2
3
4
5
6
7
8
Q4

Q5



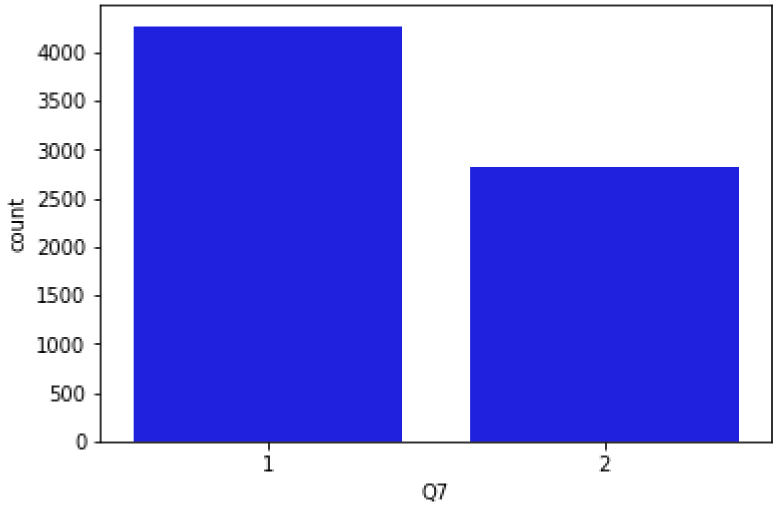
count
3000
2500
2000
1500
1000
500
0
1
2
3
4
5
6
Q5

Q6

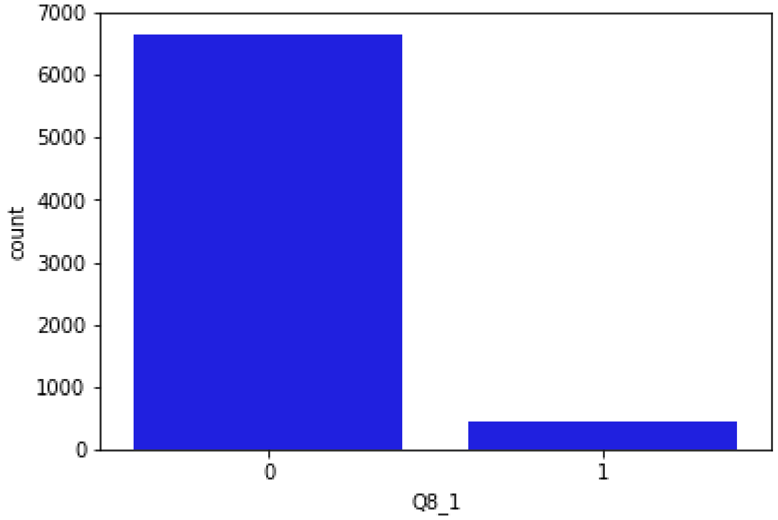


count
6000
5000
4000
3000
2000
1000
0
1
2
Q6

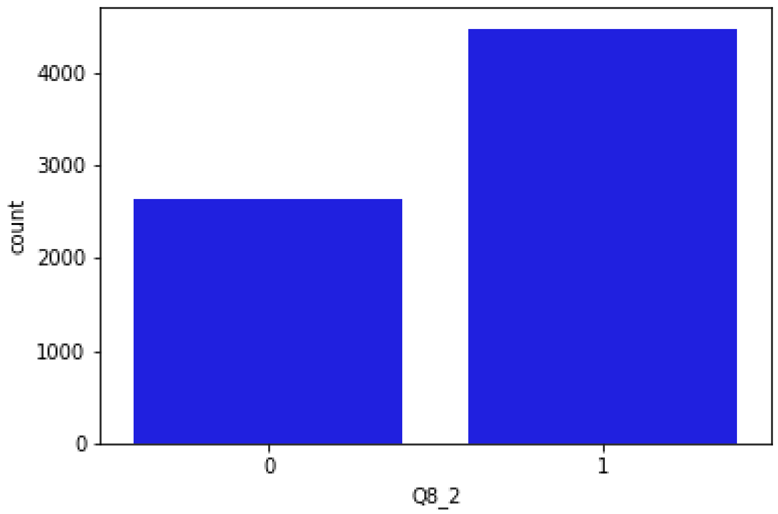
Q7


4000
3500
3000
2500
2000
1500
1000
500
0
count
1
2
Q7

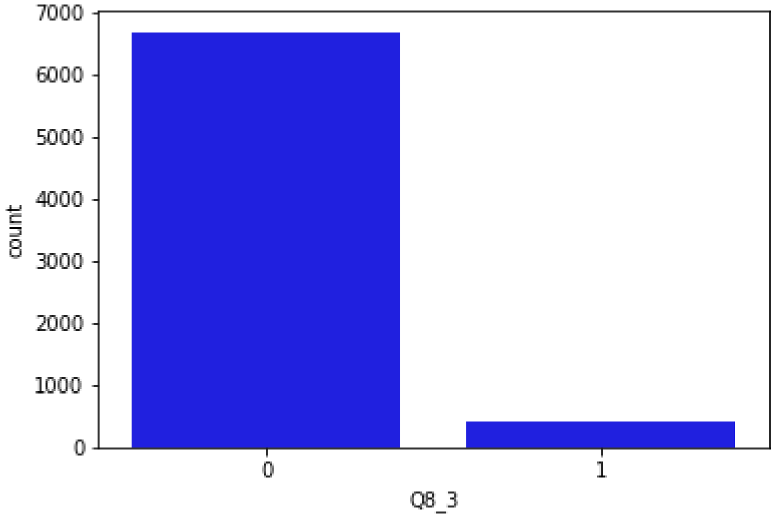
Q8_1


7000
6000
5000
4000
3000
2000
1000
0
count
0
1
Q8_1

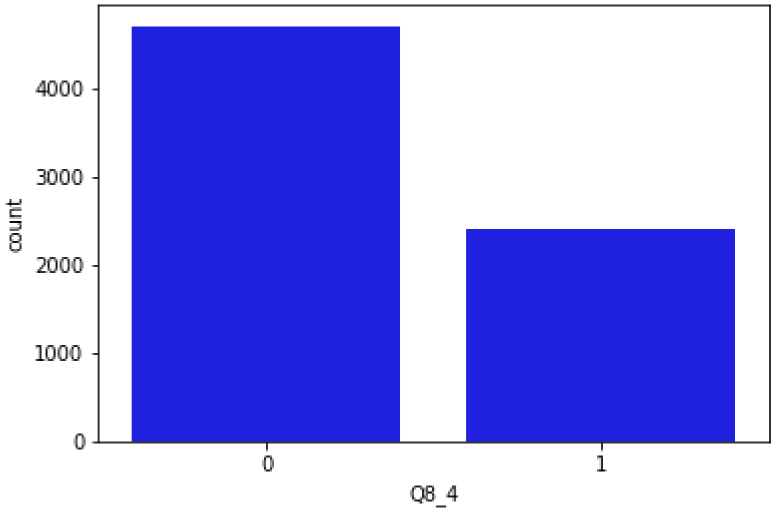
Q8_2


4000
3000
2000
1000
0
count
0
1
Q8_2

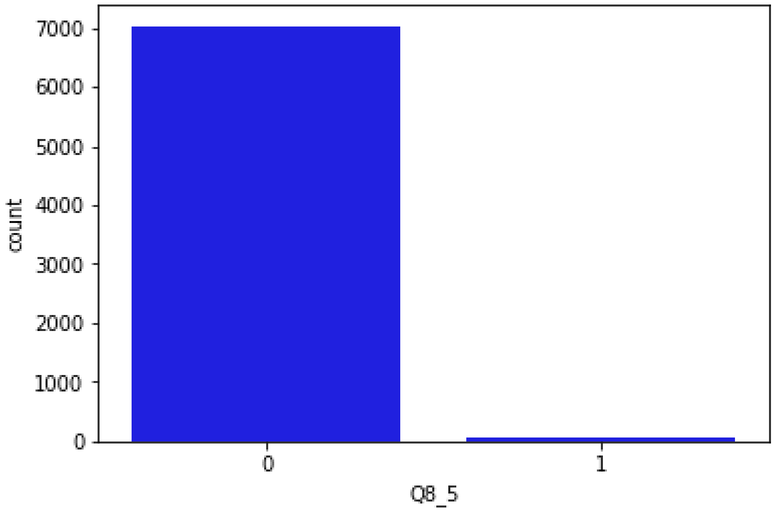
Q8_3


7000
6000
5000
4000
3000
2000
1000
0
count
0
1
Q8_3

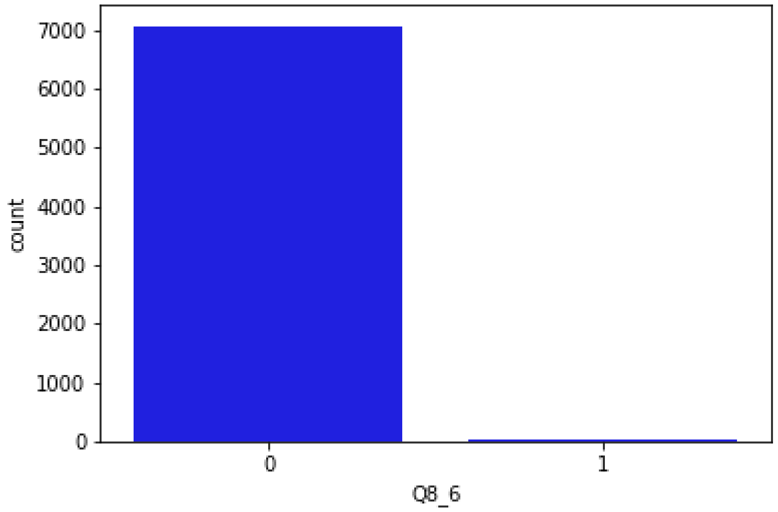
Q8_4


4000
3000
2000
1000
0
count
0
1
Q8_4

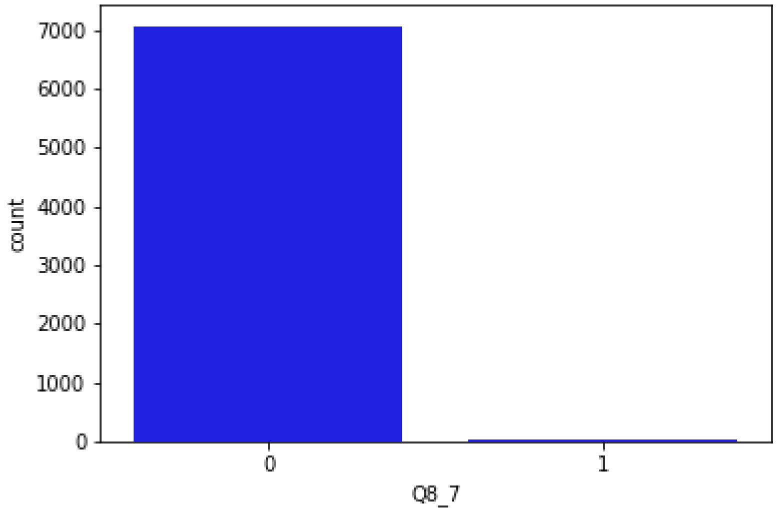
Q8_5


7000
6000
5000
4000
3000
2000
1000
0
count
0
1
Q8_5

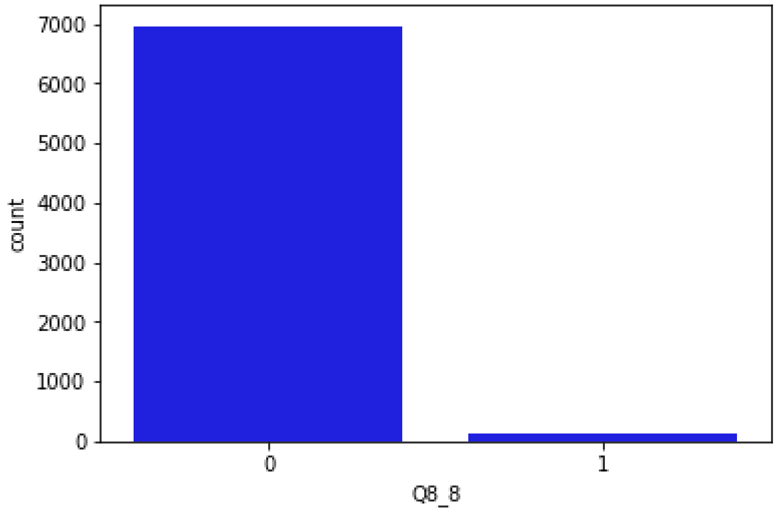
Q8_6


7000
6000
5000
4000
3000
2000
1000
0
count
0
1
Q8_6

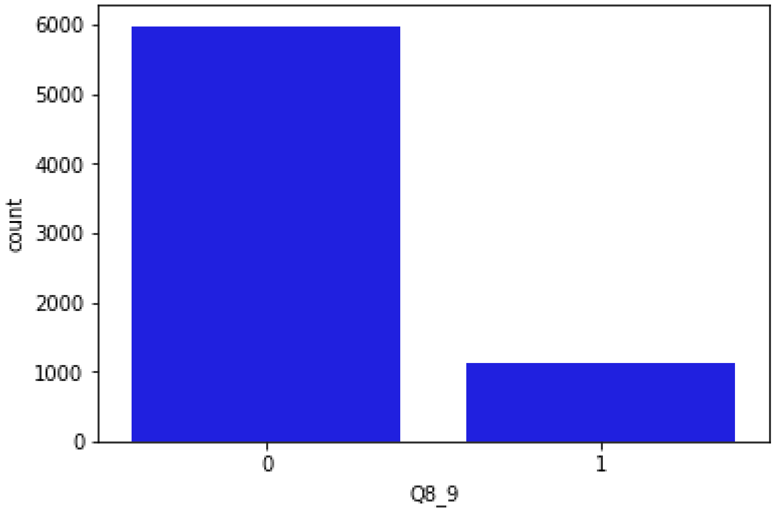
Q8_7


7000
6000
5000
4000
3000
2000
1000
0
count
0
1
Q8_7

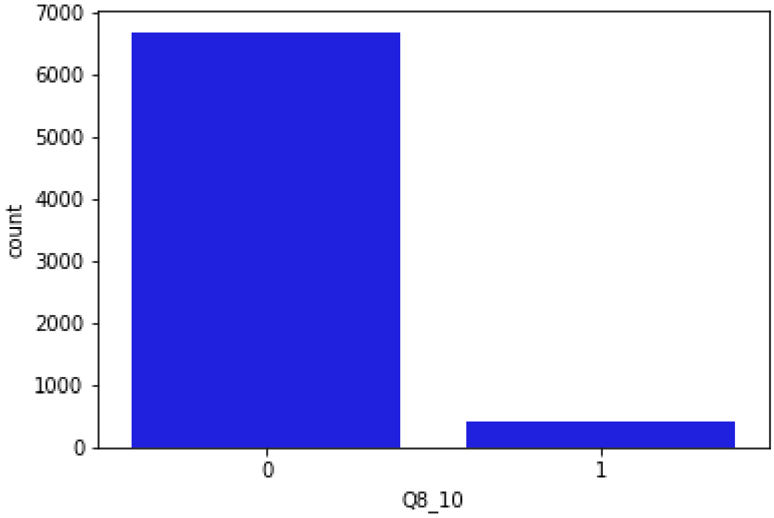
Q8_8


7000
6000
5000
4000
3000
2000
1000
0
count
0
1
Q8_8

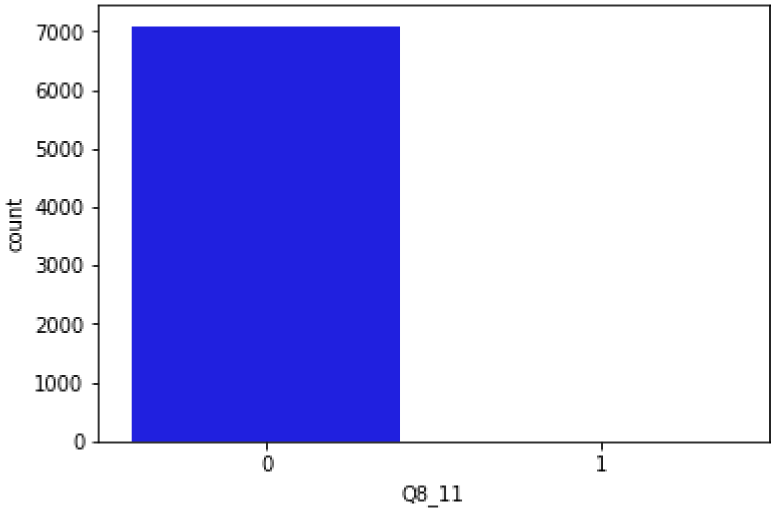
Q8_9


6000
5000
4000
3000
2000
1000
0
count
0
1
Q8_9

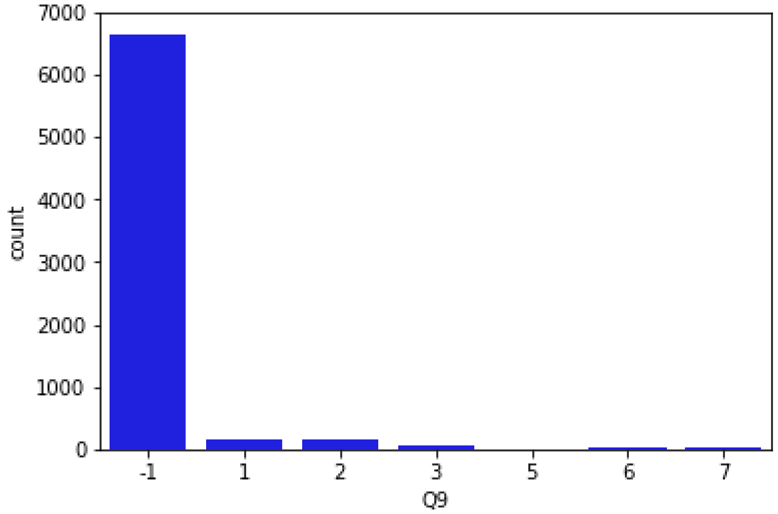
Q8_10


7000
6000
5000
4000
3000
2000
1000
0
count
0
1
Q8_10

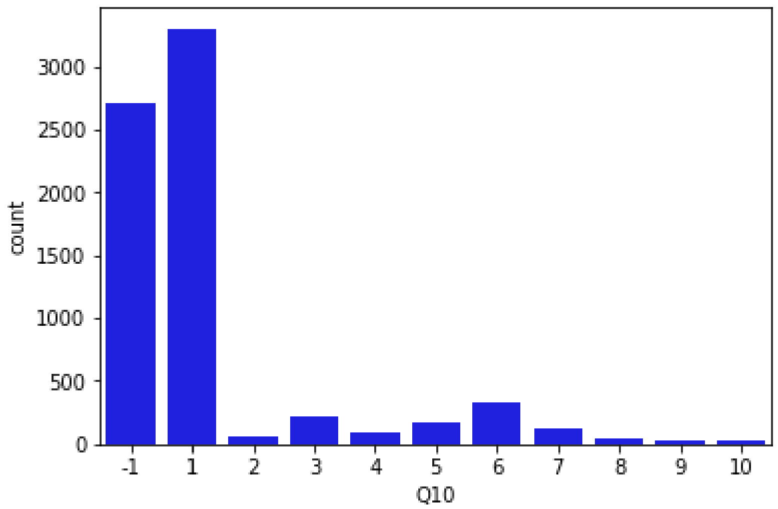
Q8_11


7000
6000
5000
4000
3000
2000
1000
0
count
0
1
Q8_11

```
Q9
```


7000
6000
5000
4000
3000
2000
1000
0
count
-1
1
2
3
5
6
7
Q9

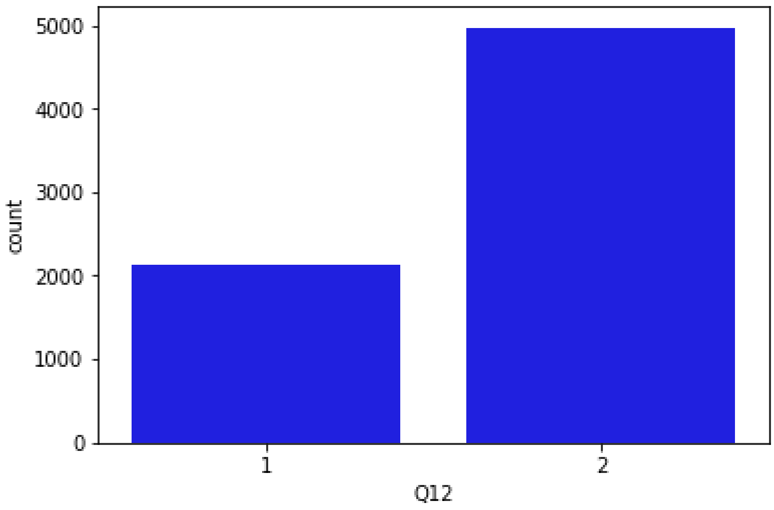
```
Q10
```


3000
2500
2000
1500
1000
500
0
count
-1
1
2
3
4
5
6
7
8
9
10
Q10

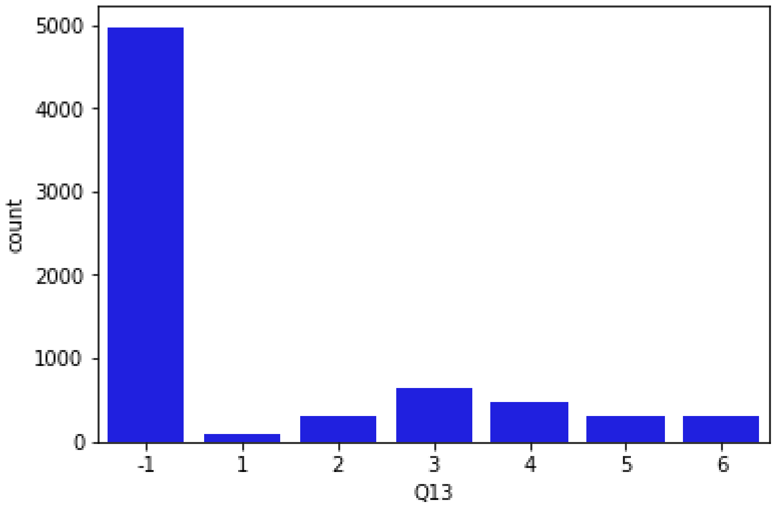
```
Q11
```

7000
6000
5000

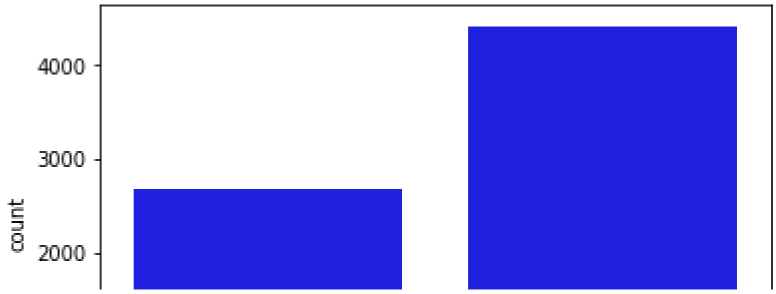
Q12


5000
4000
3000
2000
1000
0
count
1
2
Q12

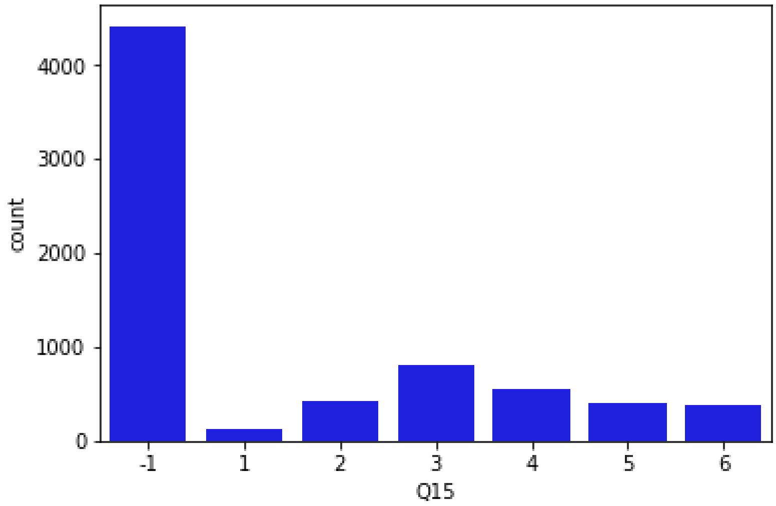
Q13


5000
4000
3000
2000
1000
0
count
-1
1
2
3
4
5
6
Q13

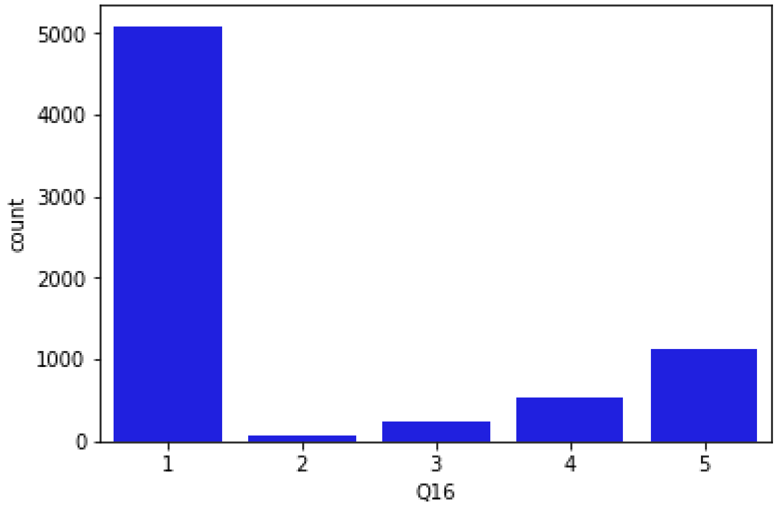
Q14


4000
3000
2000
count

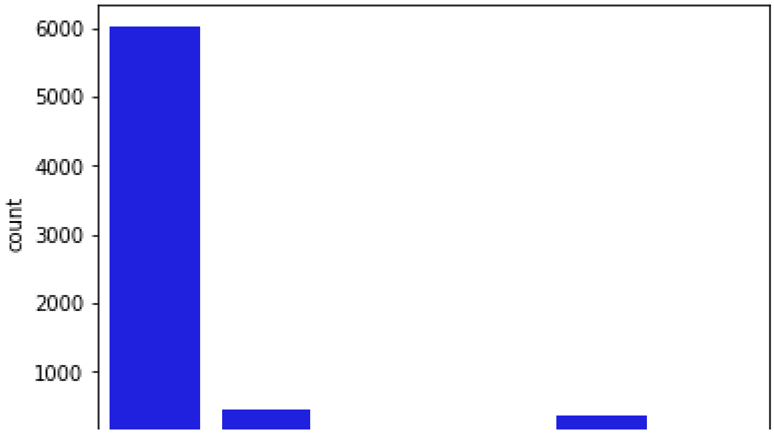
Q15


4000
3000
2000
1000
0
count
-1
1
2
3
4
5
6
Q15

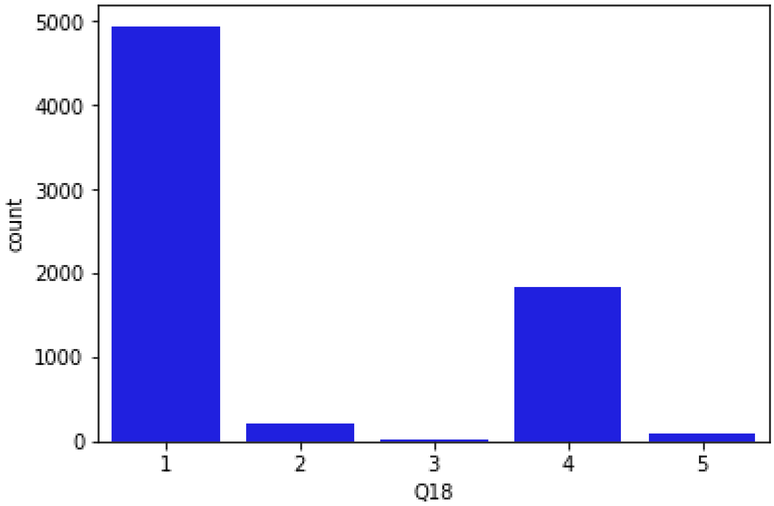
Q16


5000
4000
3000
2000
1000
0
count
1
2
3
4
5
Q16

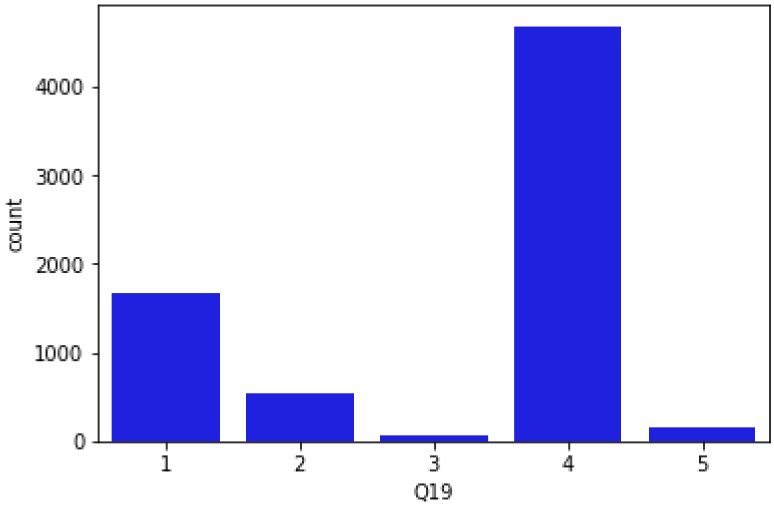
Q17


count
6000
5000
4000
3000
2000
1000

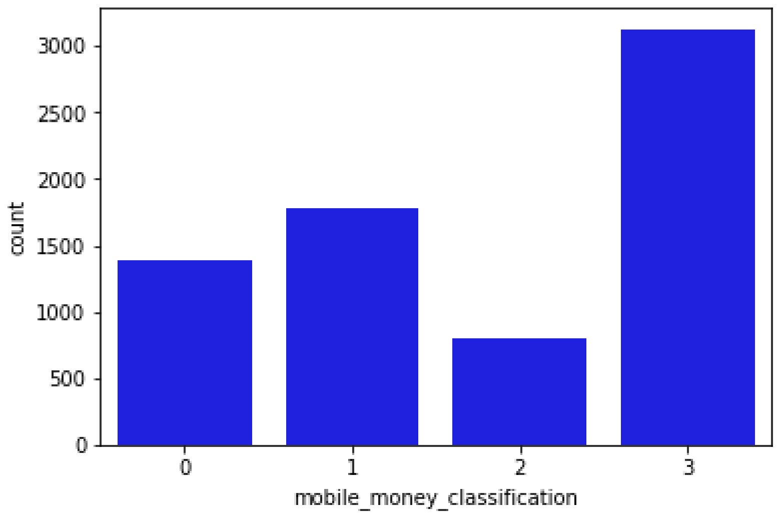
Q18


count
5000
4000
3000
2000
1000
0
1
2
3
4
5
Q18

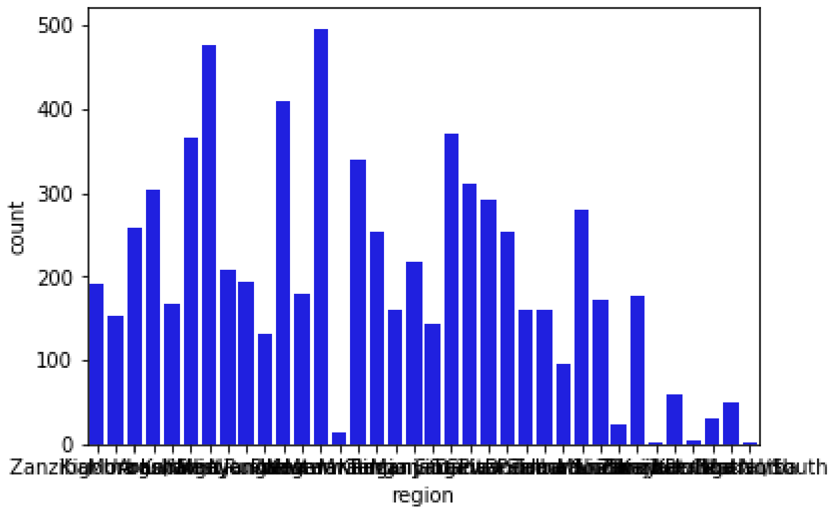
Q19


count
4000
3000
2000
1000
0
1
2
3
4
5
Q19

```
mobile_money_classification
```



```
region
```



In [16]:

```
train['mobile_money_classification'].value_counts()/train['mobile_money_classification'].
```

executed in 63ms, finished 22:00:15 2019-07-09

Out[16]:

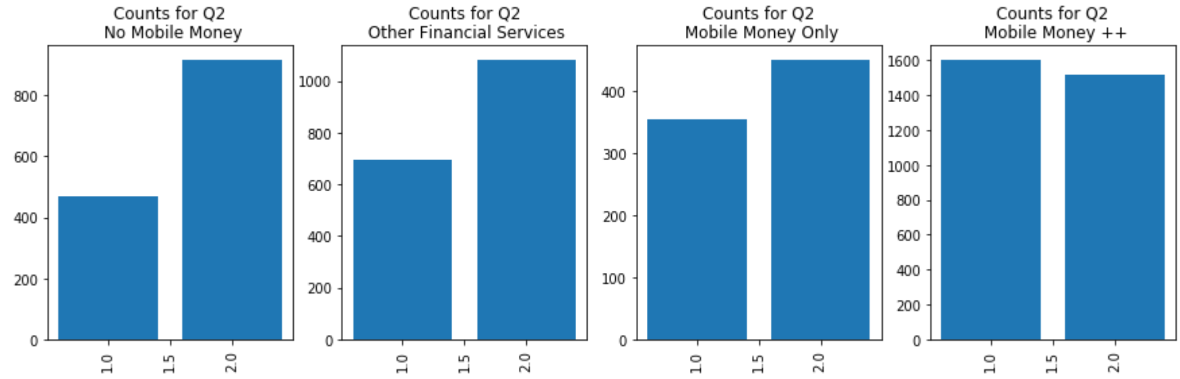
```
3    0.440654
1    0.250634
0    0.195376
2    0.113335
Name: mobile_money_classification, dtype: float64
```

In [17]:

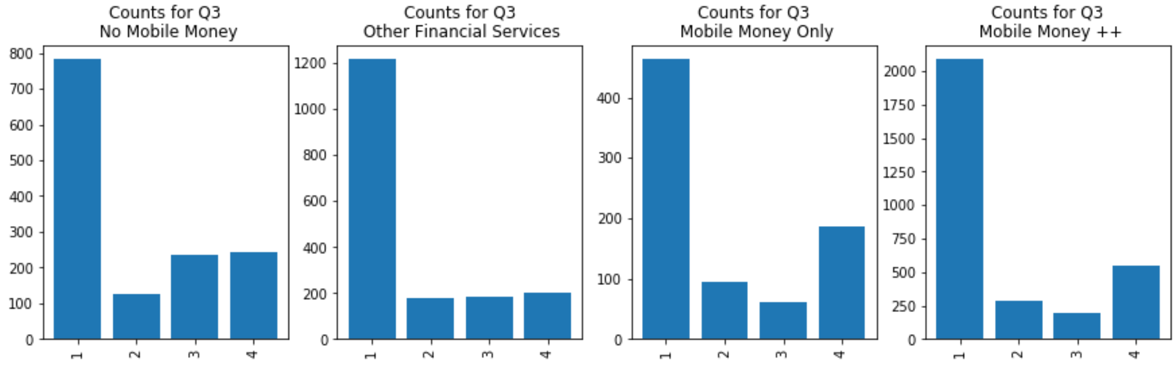
```
train['dummy'] = np.ones(shape = train.shape[0])
cat_col.remove('mobile_money_classification')

for col in cat_col:
    print(col)
    counts = train[['dummy', 'mobile_money_classification', col]].groupby(['mobile_money_
    temp = counts[counts['mobile_money_classification'] == 0][[col, 'dummy']]
    _ = plt.figure(figsize = (15,4))
    plt.subplot(1, 4, 1)
    temp = counts[counts['mobile_money_classification'] == 0][[col, 'dummy']]
    plt.bar(temp[col], temp.dummy)
    plt.xticks(rotation=90)
    plt.title('Counts for ' + col + '\n No Mobile Money')
    #plt.ylabel('count')
    plt.subplot(1, 4, 2)
    temp = counts[counts['mobile_money_classification'] == 1][[col, 'dummy']]
    plt.bar(temp[col], temp.dummy)
    plt.xticks(rotation=90)
    plt.title('Counts for ' + col + '\n Other Financial Services')
    #plt.ylabel('count')
    plt.subplot(1, 4, 3)
    temp = counts[counts['mobile_money_classification'] == 2][[col, 'dummy']]
    plt.bar(temp[col], temp.dummy)
    plt.xticks(rotation=90)
    plt.title('Counts for ' + col + '\n Mobile Money Only')
    #plt.ylabel('count')
    plt.subplot(1, 4, 4)
    temp = counts[counts['mobile_money_classification'] == 3][[col, 'dummy']]
    plt.bar(temp[col], temp.dummy)
    plt.xticks(rotation=90)
    plt.title('Counts for ' + col + '\n Mobile Money ++')
    #plt.ylabel('count')
    plt.show()

del train['dummy']
```

executed in 41.1s, finished 22:00:56 2019-07-09

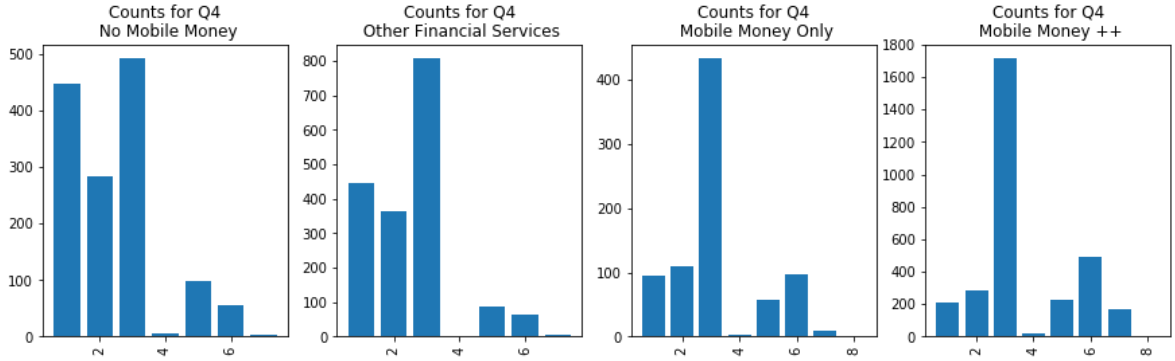
```
Q2
```



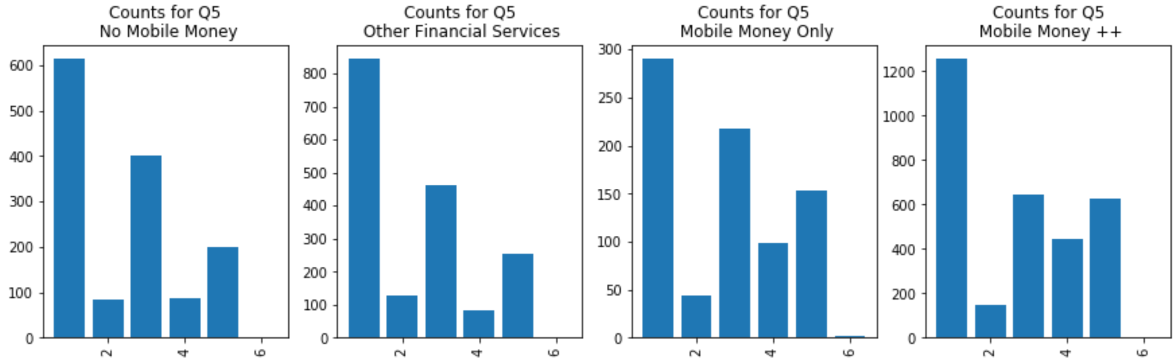
```
Q3
```


Counts for Q3
No Mobile Money
0
100
200
300
400
500
600
700
800
1
2
3
4
Counts for Q3
Other Financial Services
0
200
400
600
800
1000
1200
1
2
3
4
Counts for Q3
Mobile Money Only
0
100
200
300
400
1
2
3
4
Counts for Q3
Mobile Money ++
0
250
500
750
1000
1250
1500
1750
2000
1
2
3
4

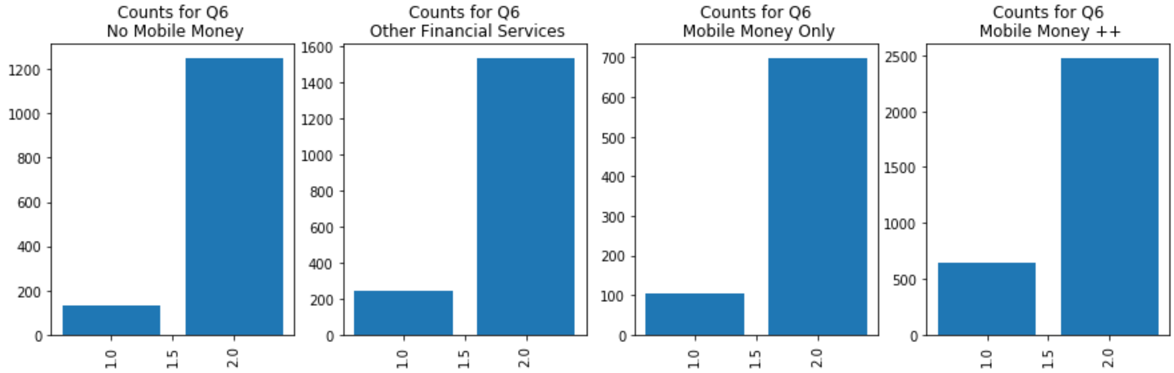
Q4


Counts for Q4
No Mobile Money
0
100
200
300
400
500
2
4
6
Counts for Q4
Other Financial Services
0
100
200
300
400
500
600
700
800
2
4
6
Counts for Q4
Mobile Money Only
0
100
200
300
400
2
4
6
8
Counts for Q4
Mobile Money ++
0
200
400
600
800
1000
1200
1400
1600
1800
2
4
6
8

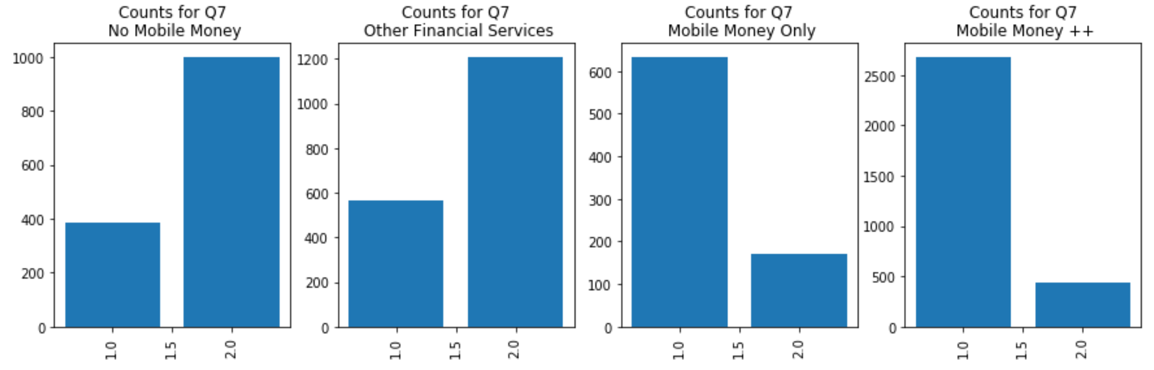
Q5


Counts for Q5
No Mobile Money
0
100
200
300
400
500
600
2
4
6
Counts for Q5
Other Financial Services
0
100
200
300
400
500
600
700
800
2
4
6
Counts for Q5
Mobile Money Only
0
50
100
150
200
250
300
2
4
6
Counts for Q5
Mobile Money ++
0
200
400
600
800
1000
1200
2
4
6

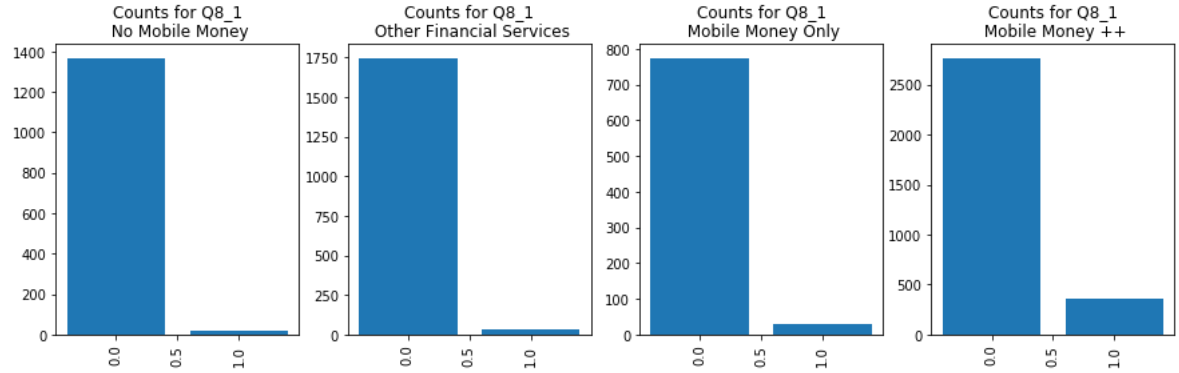
Q6


Counts for Q6
No Mobile Money
0
200
400
600
800
1000
1200
1.0
1.5
2.0
Counts for Q6
Other Financial Services
0
200
400
600
800
1000
1200
1400
1600
1.0
1.5
2.0
Counts for Q6
Mobile Money Only
0
100
200
300
400
500
600
700
1.0
1.5
2.0
Counts for Q6
Mobile Money ++
0
500
1000
1500
2000
2500
1.0
1.5
2.0

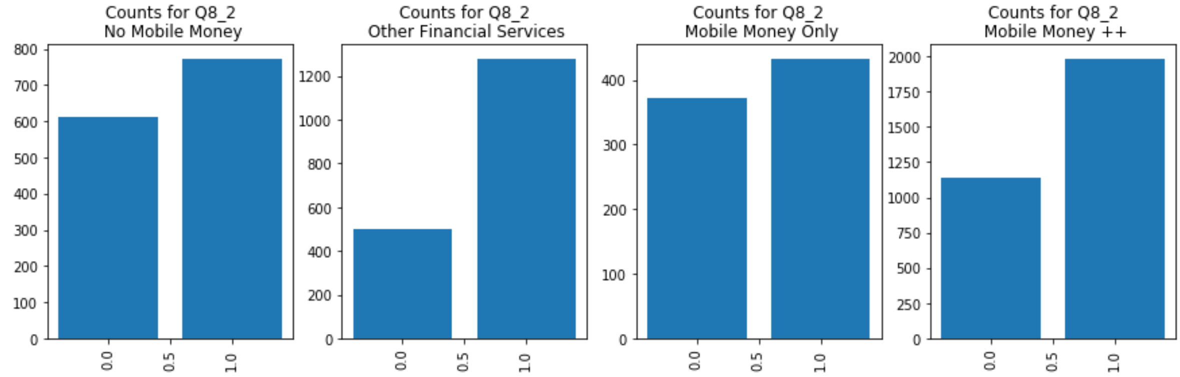
Q7



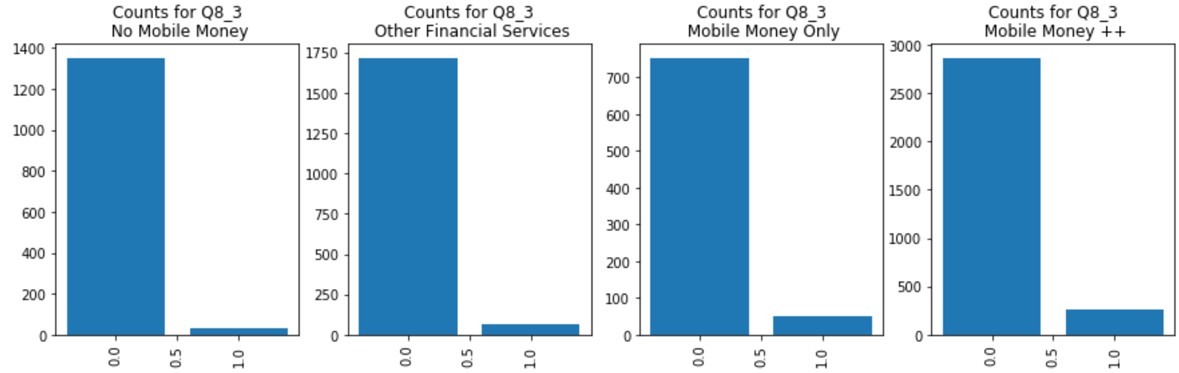
Q8_1



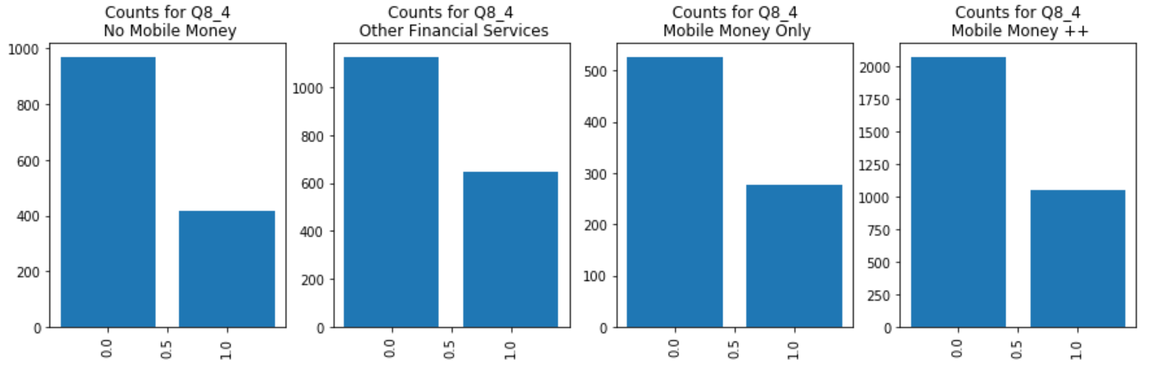
Q8_2



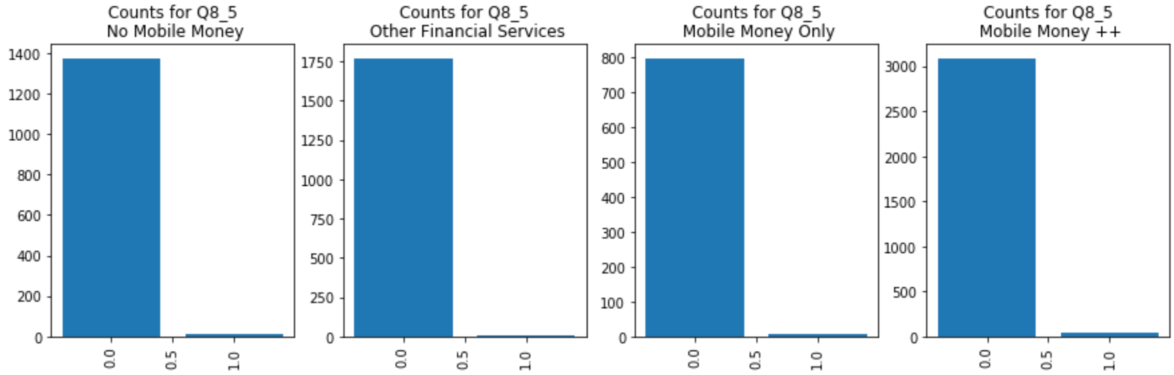
Q8_3



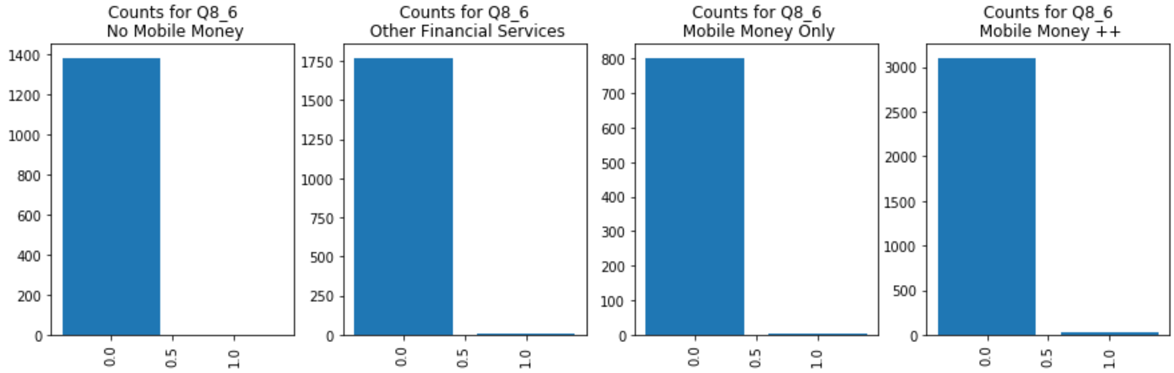
Q8_4



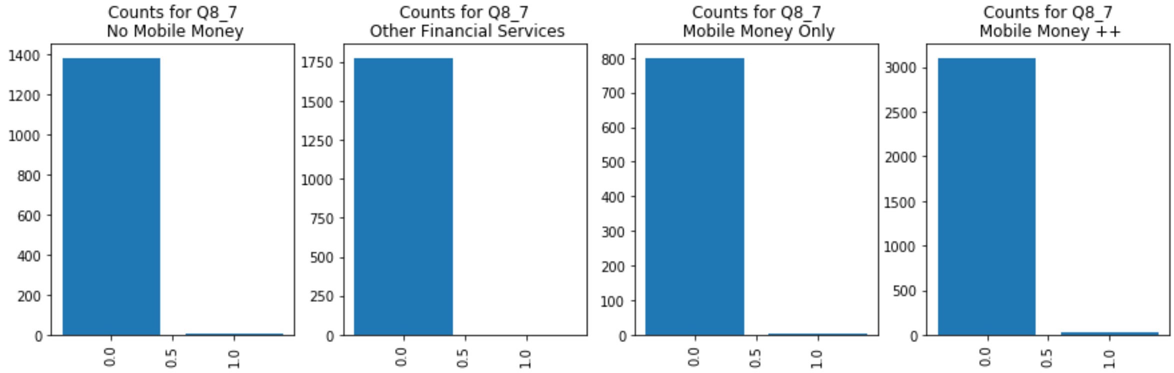
Q8_5



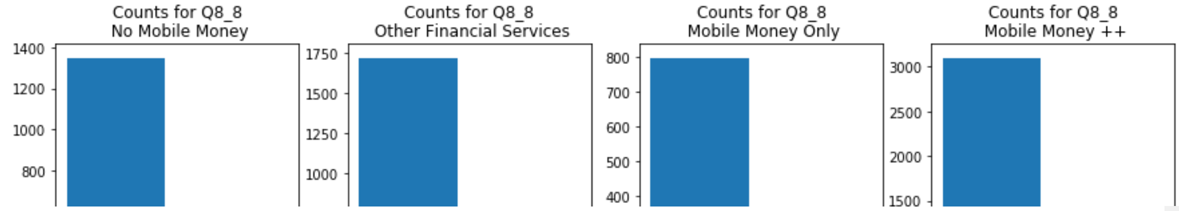
Q8_6



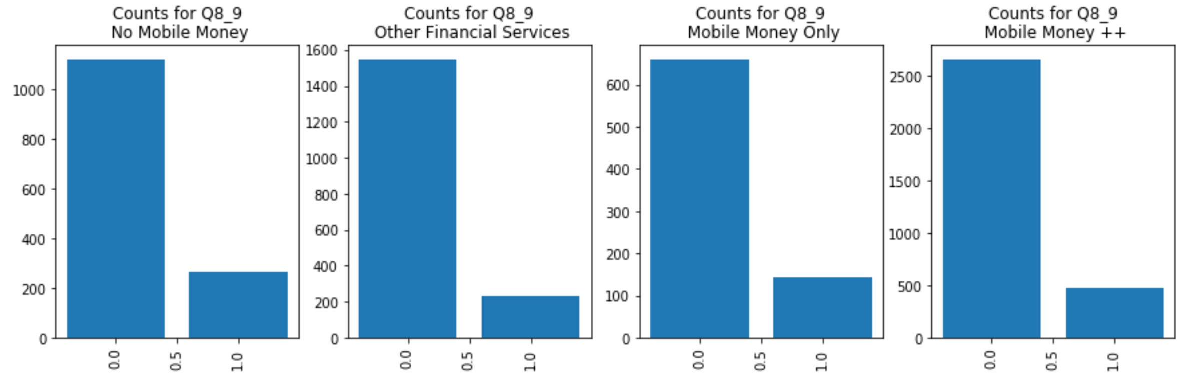
Q8_7



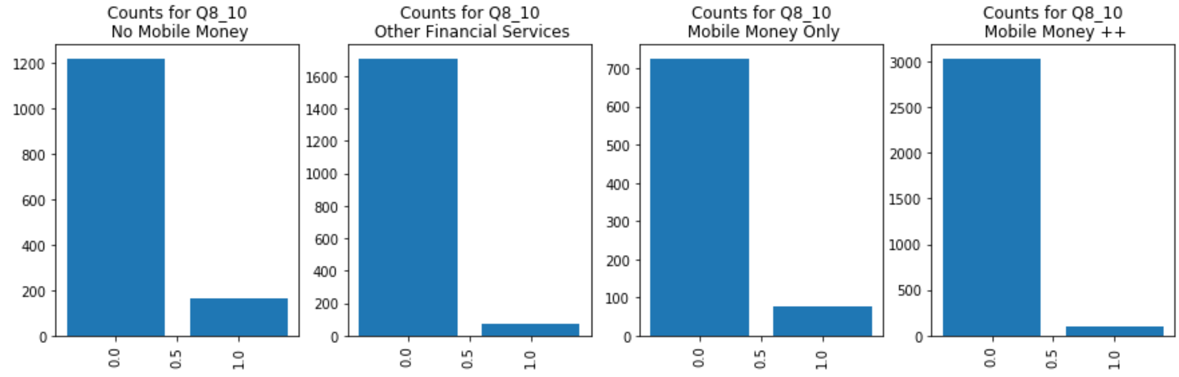
Q8_8



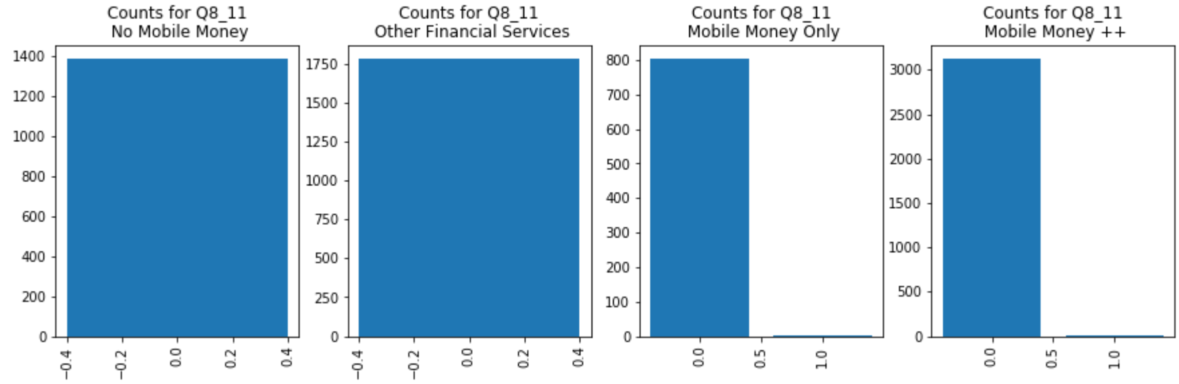
Q8_9



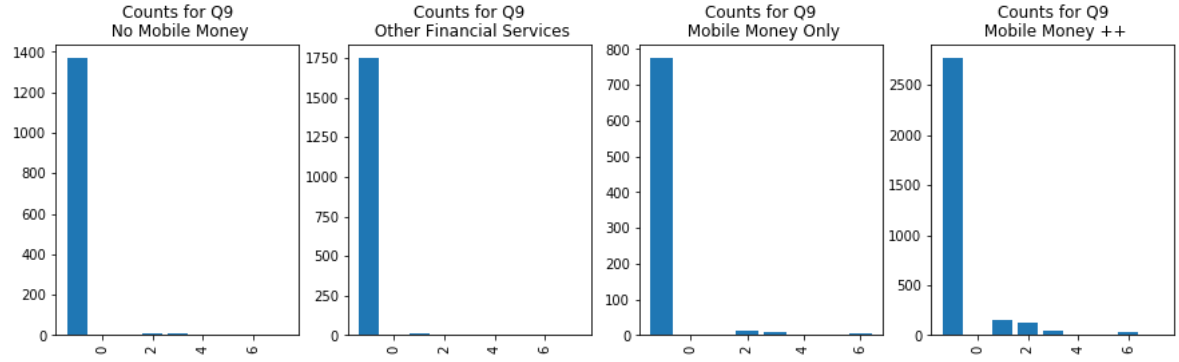
Q8_10



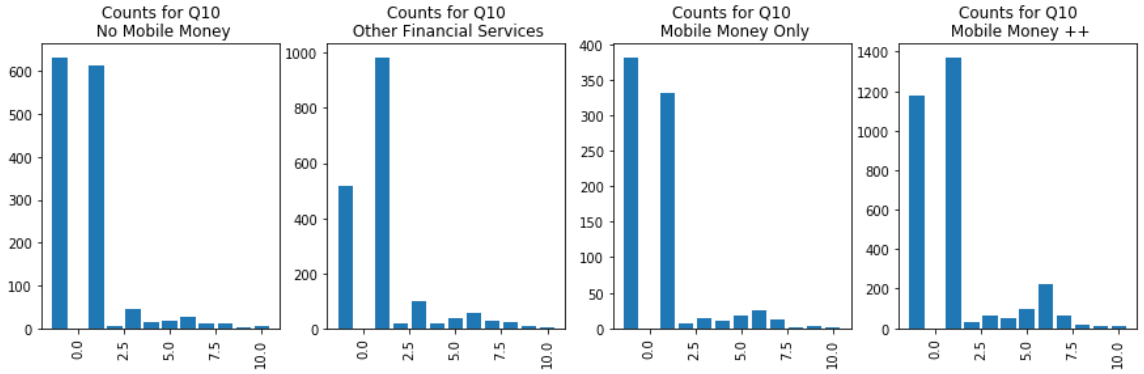
Q8_11



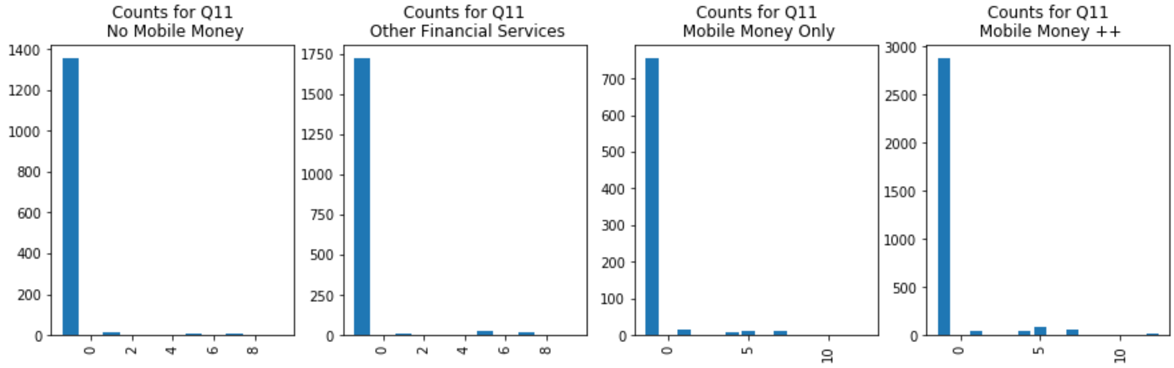
Q9



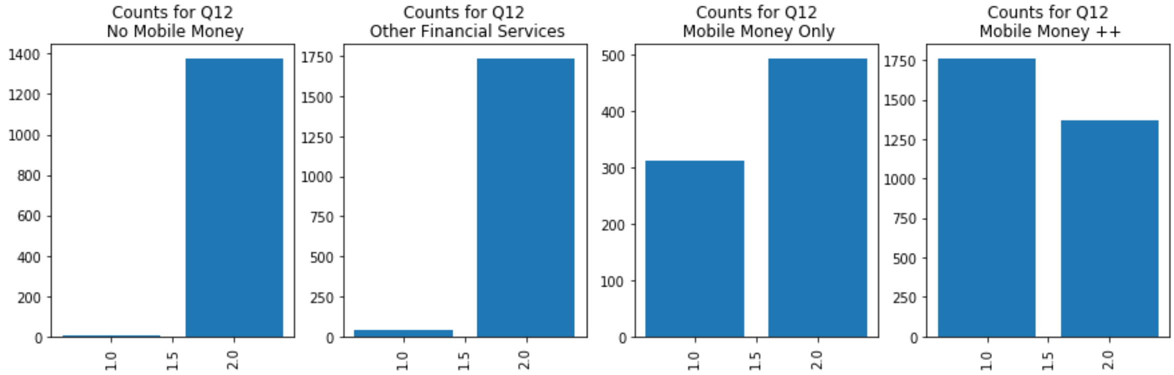
Q10



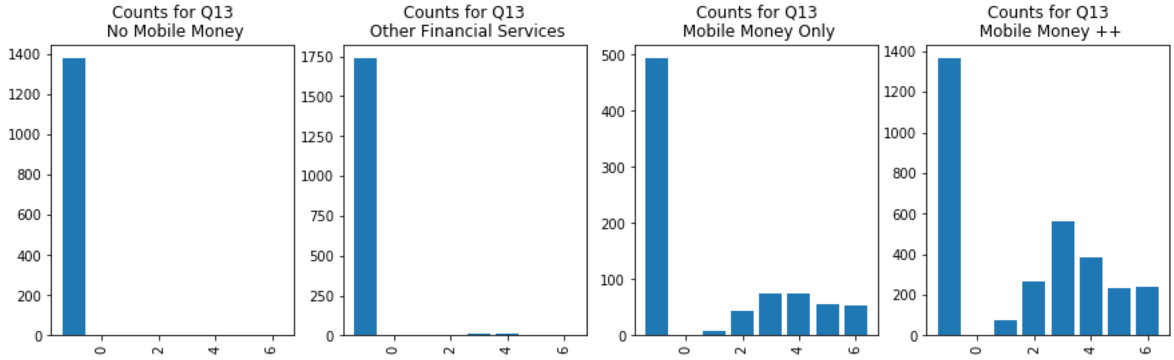
Q11



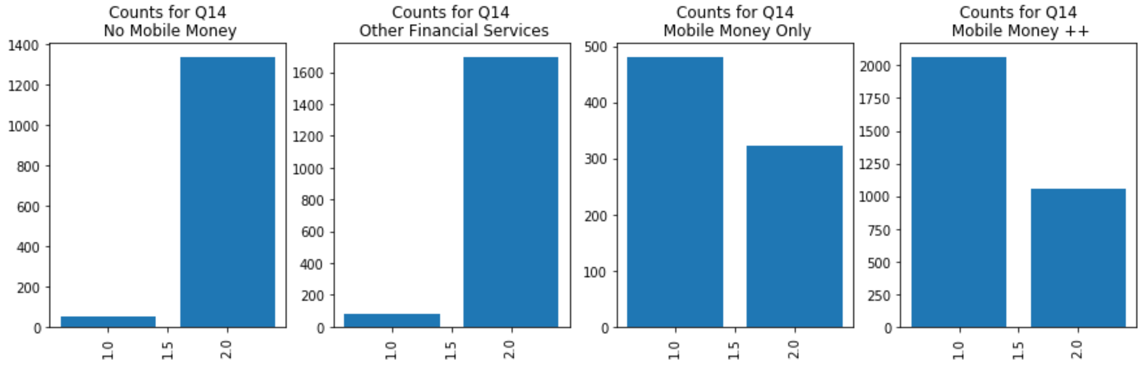
Q12



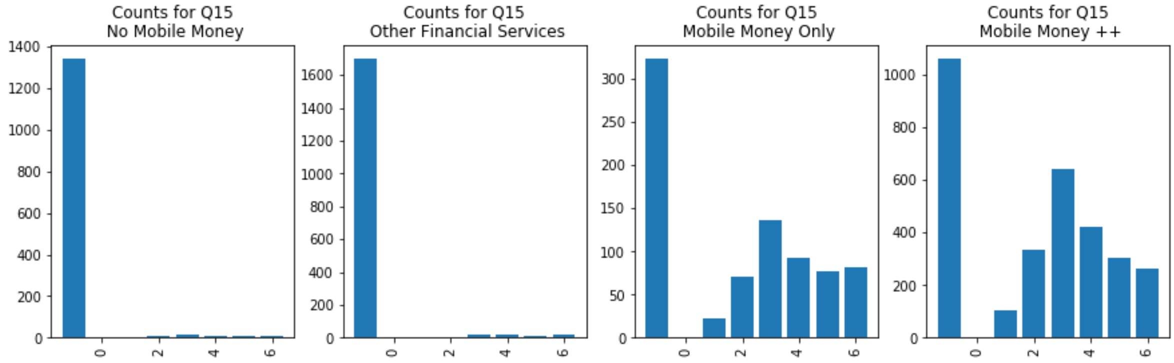
Q13



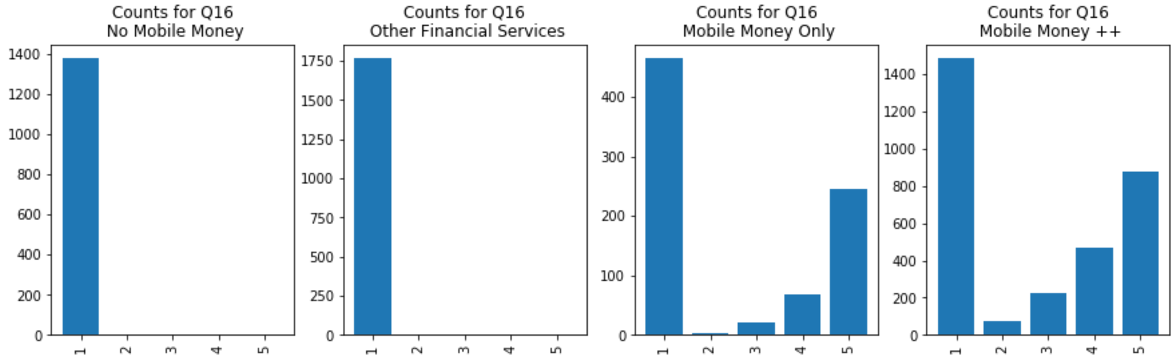
Q14



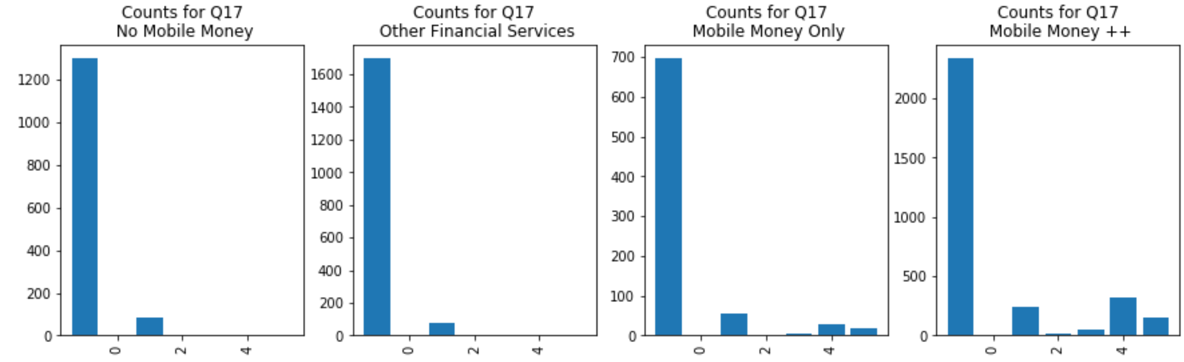
Q15



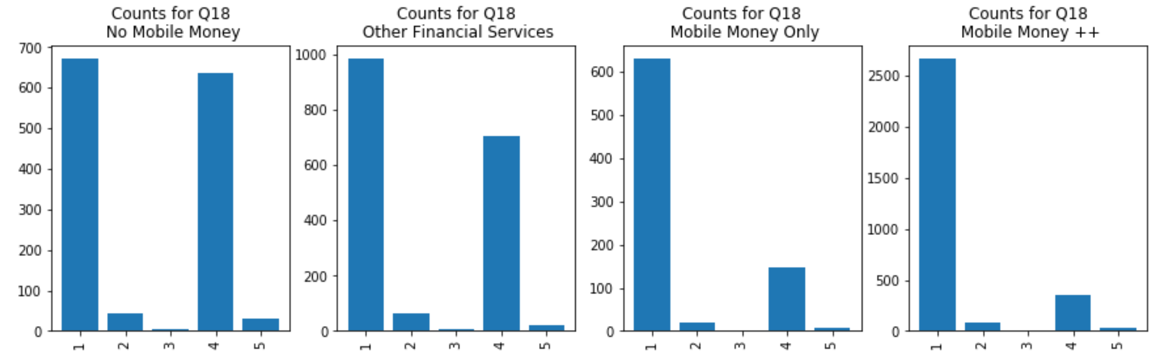
Q16



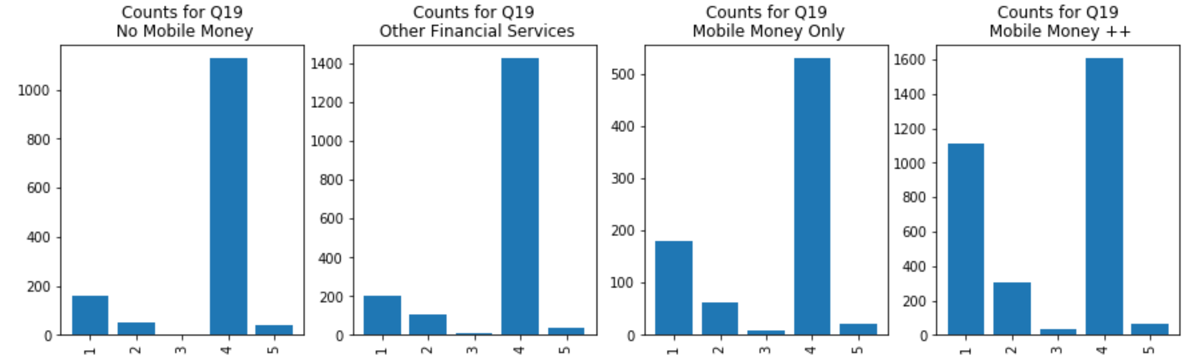
Q17



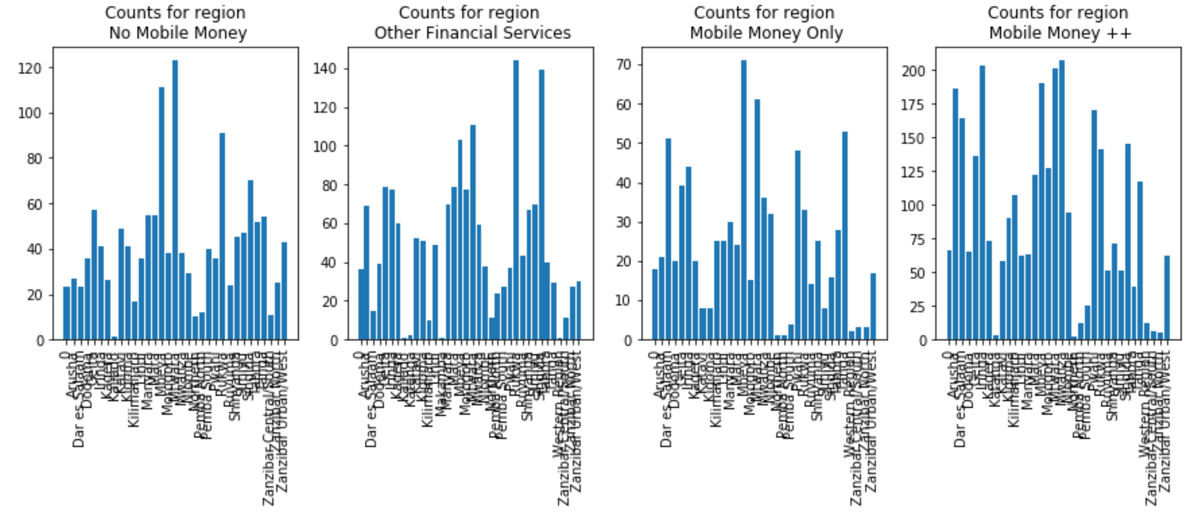
Q18



Q19



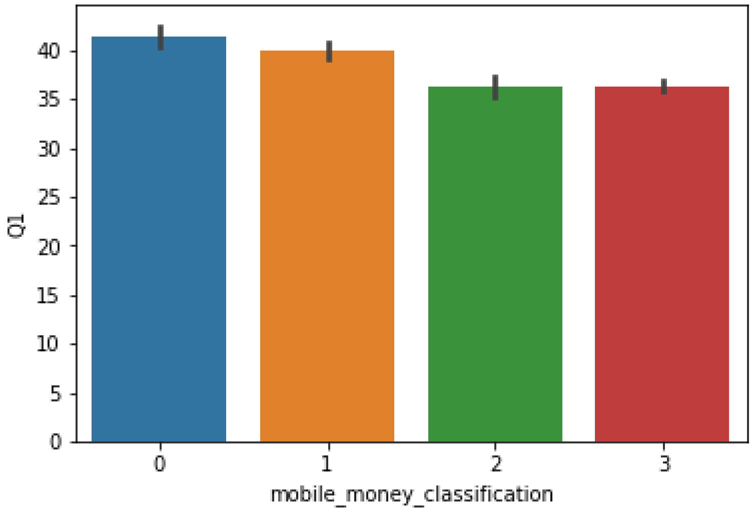
region

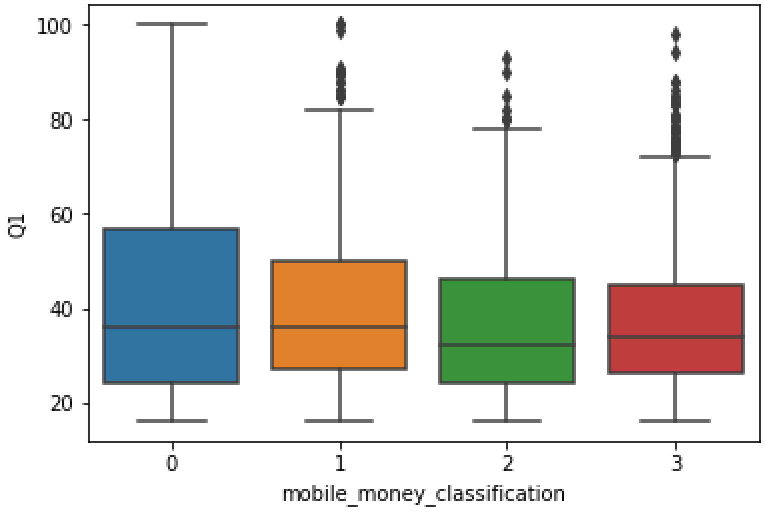


In [18]:

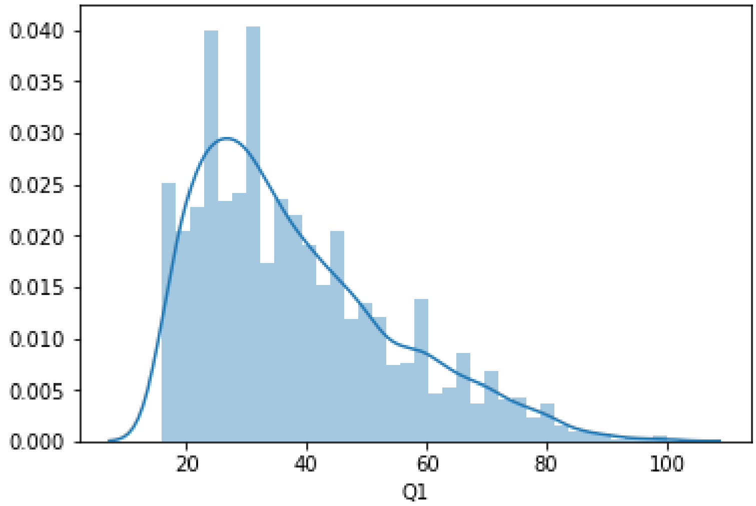
```
for col in num_col:
    sns.barplot(data=train,x='mobile_money_classification',y=col)
    plt.show()
    sns.boxplot(data=train,x='mobile_money_classification',y=col)
    plt.show()
    sns.distplot(train[col])
    plt.show()
```

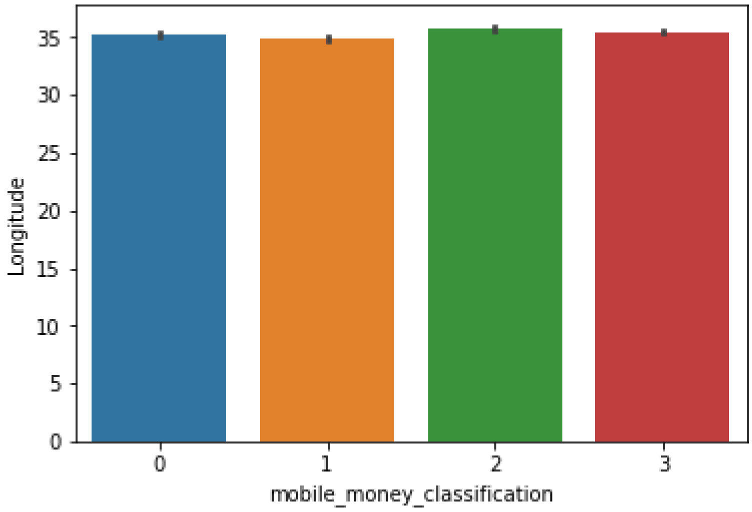
executed in 6.32s, finished 22:01:03 2019-07-09

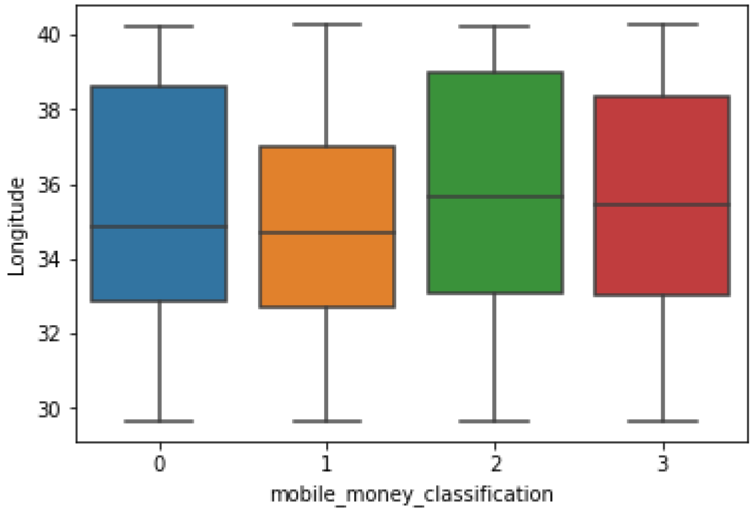


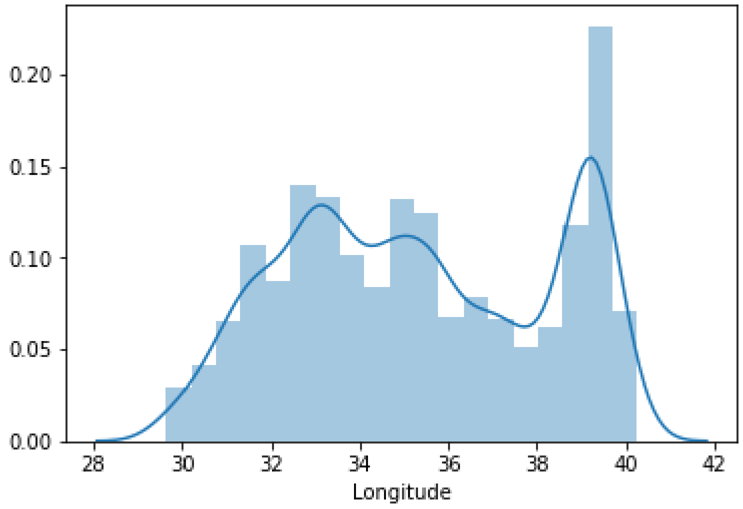


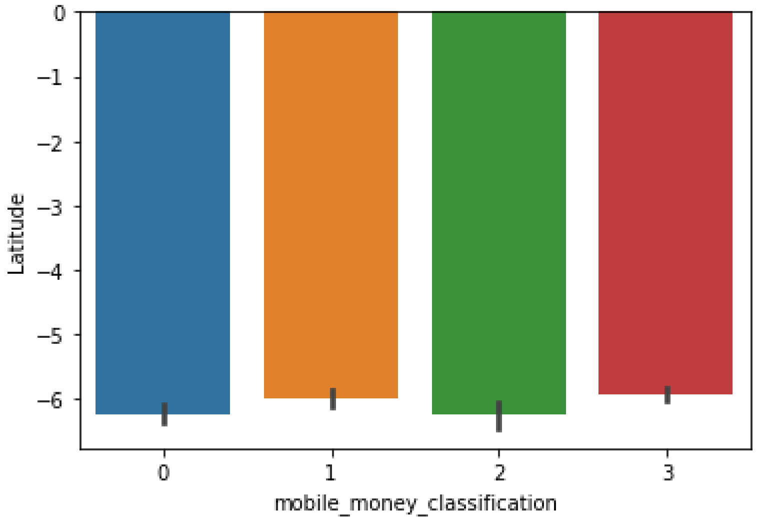
```
C:\Users\ADEBAYO\Anaconda3\lib\site-packages\matplotlib\axes\_axes.py:6462:
UserWarning: The 'normed' kwarg is deprecated, and has been replaced by the
'density' kwarg.
  warnings.warn("The 'normed' kwarg is deprecated, and has been "
```

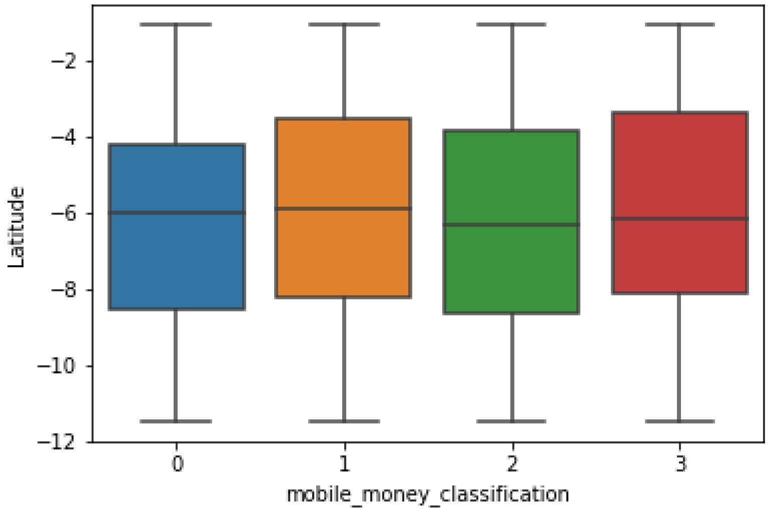

0.040
0.035
0.030
0.025
0.020
0.015
0.010
0.005
0.000
20
40
60
80
100
Q1

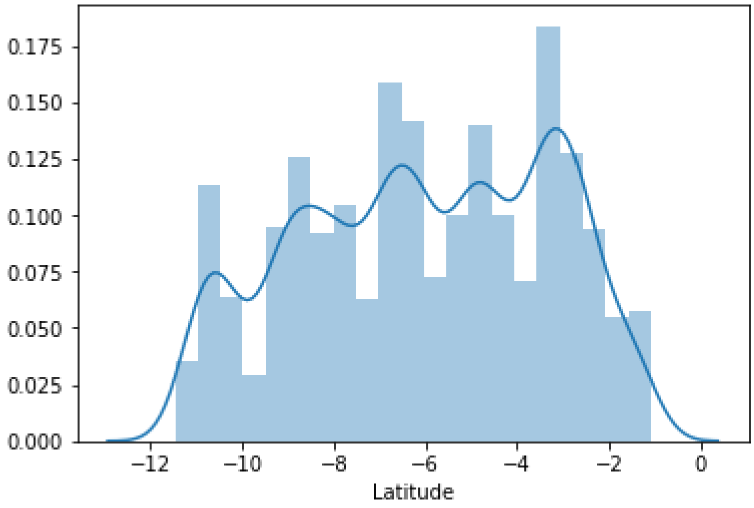

35
30
25
20
15
10
5
0
Longitude
0
1
2
3
mobile_money_classification


40
38
36
34
32
30
Longitude
0
1
2
3
mobile_money_classification


0.20
0.15
0.10
0.05
0.00
28
30
32
34
36
38
40
42
Longitude


0
−1
−2
−3
−4
−5
−6
Latitude
0
1
2
3
mobile_money_classification


−2
−4
−6
−8
−10
−12
Latitude
0
1
2
3
mobile_money_classification



In [19]:

```
X=train.drop(['mobile_money_classification'],axis=1)
Y=train['mobile_money_classification']
```

executed in 30ms, finished 22:01:03 2019-07-09

In [20]:

```
X.isnull().sum()
```

executed in 158ms, finished 22:01:03 2019-07-09

Out[20]:

```
Q1            0
Q2            0
Q3            0
Q4            0
Q5            0
Q6            0
Q7            0
Q8_1          0
Q8_2          0
Q8_3          0
Q8_4          0
Q8_5          0
Q8_6          0
Q8_7          0
Q8_8          0
Q8_9          0
Q8_10         0
Q8_11         0
Q9            0
Q10           0
Q11           0
Q12           0
Q13           0
Q14           0
Q15           0
Q16           0
Q17           0
Q18           0
Q19           0
Latitude      0
Longitude     0
region        0
dtype: int64
```

In [21]:

```
X['region'].fillna('unknown',inplace=True)
```

executed in 100ms, finished 22:01:03 2019-07-09

In [22]:

```
split_test_size=0.2

from sklearn.model_selection import train_test_split
Xtrain, Xtest, Ytrain, Ytest= train_test_split(X,Y, test_size=split_test_size, random_sta
```

executed in 4.58s, finished 22:01:08 2019-07-09

In [23]:

```
from catboost import CatBoostClassifier
```

executed in 1.69s, finished 22:01:09 2019-07-09

In [24]:

```
X.columns
```

executed in 47ms, finished 22:01:09 2019-07-09

Out[24]:

```
Index(['Q1', 'Q2', 'Q3', 'Q4', 'Q5', 'Q6', 'Q7', 'Q8_1', 'Q8_2', 'Q8_3',
       'Q8_4', 'Q8_5', 'Q8_6', 'Q8_7', 'Q8_8', 'Q8_9', 'Q8_10', 'Q8_11', 'Q
9',
       'Q10', 'Q11', 'Q12', 'Q13', 'Q14', 'Q15', 'Q16', 'Q17', 'Q18', 'Q19',
       'Latitude', 'Longitude', 'region'],
      dtype='object')
```

In [26]:

```
cb_cat=CatBoostClassifier(iterations=1000,depth=5,loss_function='MultiClass',
                          cat_features=[i for i in range(1,32) if i not in [29,30]],
                    random_seed=10,learning_rate=.5,verbose=False)

cb_cat.fit(Xtrain,Ytrain,use_best_model=True,
           eval_set=(Xtest,Ytest),early_stopping_rounds=100,verbose=50)
```

executed in 2m 7s, finished 22:05:08 2019-07-09

```
0:      learn: -0.9881089       test: -0.9935931        best: -0.9935931 (0)
total: 1.39s    remaining: 23m 10s
50:     learn: -0.6791892       test: -0.7682336        best: -0.7631158 (3
3)      total: 49.6s    remaining: 15m 23s
100:    learn: -0.6167393       test: -0.7762152        best: -0.7631158 (3
3)      total: 1m 35s   remaining: 14m 6s
Stopped by overfitting detector  (100 iterations wait)

bestTest = -0.7631158036
bestIteration = 33

Shrink model to first 34 iterations.
```

Out[26]:

```
<catboost.core.CatBoostClassifier at 0x1feb9afd780>
```

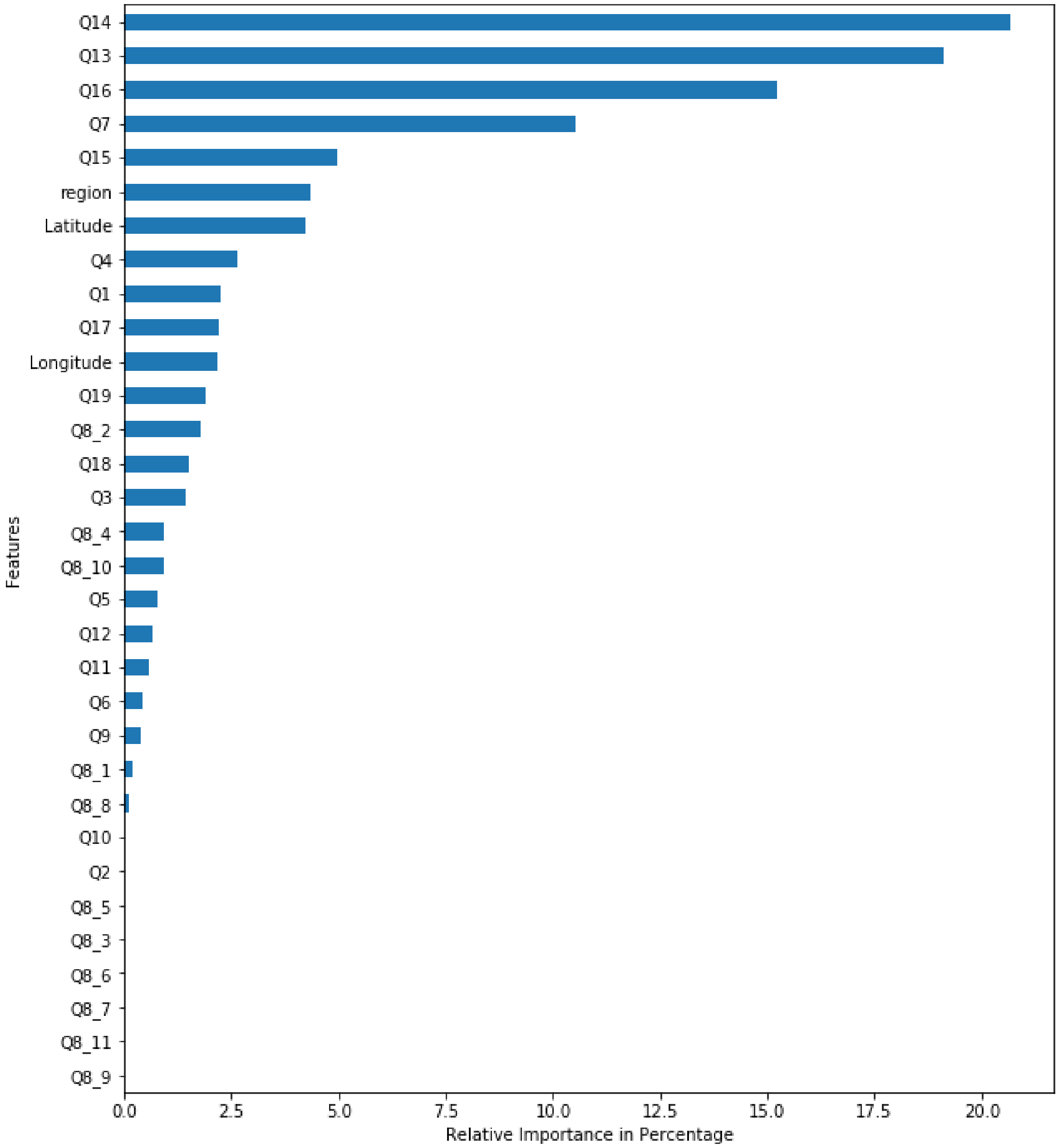
In [27]:

```
b=list(cb_cat.feature_importances_[:])
pd.DataFrame(index=X.columns,data=b).sort_values(0).plot.barh(figsize=(10,12),legend=Fals
plt.ylabel('Features')
plt.xlabel('Relative Importance in Percentage')
```

executed in 1.31s, finished 22:05:10 2019-07-09

Out[27]:

```
Text(0.5,0,'Relative Importance in Percentage')
```



In [ ]: