In [1]:

```
import pandas as pd
import numpy as np
import seaborn as sns
```

In [2]:

```
df1=pd.read_csv(r'D:/Datasets/UTA2019/Final Cleaned Data.csv')
```

In [3]:

```
#df1['Utility'].head(10)
```

Exploratory Data Analysis

In [4]:

```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 192901 entries, 0 to 192900
Data columns (total 29 columns):
Class ID                       192900 non-null float64
Color                          187606 non-null object
Count Of Big Transactions      192900 non-null float64
Country Of Origin              192900 non-null object
Depth                          192893 non-null float64
Finish Name                    11404 non-null object
Height                         192900 non-null float64
Life Cycle Name                192900 non-null object
Material                       85769 non-null object
Product Name                   192900 non-null object
Quartile                       192900 non-null float64
SKU                            192900 non-null float64
Season                         74062 non-null object
Sentiment                      192900 non-null float64
Category                       182127 non-null object
Utility                        182127 non-null object
Weight                         192900 non-null float64
Width                          192900 non-null float64
Answercount                    192900 non-null float64
Averagerating                  155773 non-null float64
Commentcount                   192900 non-null float64
Display-Name                   88939 non-null object
Long-Description               65279 non-null object
Online-Flag                    192901 non-null bool
Questioncount                  192900 non-null float64
Reviewcount                    192900 non-null float64
Tags                           122298 non-null object
Text                           192894 non-null object
Title                          143113 non-null object
dtypes: bool(1), float64(14), object(14)
memory usage: 41.4+ MB
```

In [5]:

```
#df1.head(5)
```

```
In [6]: df1.columns
```

```
Out[6]: Index(['Class ID', 'Color', 'Count Of Big Transactions', 'Country Of Origin',
               'Depth', 'Finish Name', 'Height', 'Life Cycle Name', 'Material',
               'Product Name', 'Quartile', 'SKU', 'Season', 'Sentiment', 'Category',
               'Utility', 'Weight', 'Width', 'Answercount', 'Averagerating',
               'Commentcount', 'Display-Name', 'Long-Description', 'Online-Flag',
               'Questioncount', 'Reviewcount', 'Tags', 'Text', 'Title'],
              dtype='object')
```

```
In [7]: df1.isna().sum()
```

```
Out[7]: Class ID                          1
        Color                          5295
        Count Of Big Transactions         1
        Country Of Origin                 1
        Depth                             8
        Finish Name                  181497
        Height                            1
        Life Cycle Name                   1
        Material                     107132
        Product Name                      1
        Quartile                          1
        SKU                               1
        Season                       118839
        Sentiment                         1
        Category                      10774
        Utility                       10774
        Weight                            1
        Width                             1
        Answercount                       1
        Averagerating                 37128
        Commentcount                      1
        Display-Name                 103962
        Long-Description             127622
        Online-Flag                       0
        Questioncount                     1
        Reviewcount                       1
        Tags                          70603
        Text                              7
        Title                         49788
        dtype: int64
```

```
In [8]: df1.drop(['Finish Name','Material','Season','Display-Name','Long-Description','Tag
        s','Utility','Title'],axis=1,inplace=True)
```

```
In [9]: #df1.head(5)
```

In [10]:
```
df1.isna().sum()
```

Out[10]:
```
Class ID                         1
Color                         5295
Count Of Big Transactions        1
Country Of Origin                1
Depth                            8
Height                           1
Life Cycle Name                  1
Product Name                     1
Quartile                         1
SKU                              1
Sentiment                        1
Category                     10774
Weight                           1
Width                            1
Answercount                      1
Averagerating                37128
Commentcount                     1
Online-Flag                      0
Questioncount                    1
Reviewcount                      1
Text                             7
dtype: int64
```

In [11]:
```
df1.groupby('Quartile')['Quartile'].count()
```

Out[11]:
```
Quartile
1.0     10134
2.0     18934
3.0     32865
4.0    130967
Name: Quartile, dtype: int64
```

```
In [12]: df1.groupby('Quartile')['Averagerating'].value_counts()
```

```
Out[12]: Quartile  Averagerating
         1.0       0.0               6930
                   5.0               2171
                   4.0                218
                   3.0                 89
                   1.0                 39
                   2.0                 35
         2.0       0.0              12018
                   5.0               4573
                   4.0                447
                   3.0                309
                   1.0                141
                   2.0                118
         3.0       0.0              17371
                   5.0               9371
                   4.0                855
                   3.0                371
                   1.0                138
                   2.0                127
         4.0       0.0              88796
                   5.0              10557
                   4.0                807
                   3.0                176
                   2.0                 68
                   1.0                 48
         Name: Averagerating, dtype: int64
```

```
In [13]: df1['Color']=df1['Color'].fillna('Unknown')
         df1['Category'] = df1['Category'].fillna('Unknown')
         #df1['Averagerating']=df1['Averagerating'].fillna()
         #df1['Quartile'] = df1['Quartile'].dropna()
         df1['Averagerating'] = df1['Averagerating'].fillna(df1['Quartile'])
```

```
In [14]: df1.isna().sum()
```

```
Out[14]: Class ID                    1
         Color                       0
         Count Of Big Transactions   1
         Country Of Origin           1
         Depth                       8
         Height                      1
         Life Cycle Name             1
         Product Name                1
         Quartile                    1
         SKU                         1
         Sentiment                   1
         Category                    0
         Weight                      1
         Width                       1
         Answercount                 1
         Averagerating               1
         Commentcount                1
         Online-Flag                 0
         Questioncount               1
         Reviewcount                 1
         Text                        7
         dtype: int64
```

```
In [15]: df1.dropna(axis=0,inplace = True)
```

```
In [16]: df1['Averagerating'].value_counts()
```

```
Out[16]: 0.0    125106
         4.0     32840
         5.0     26670
         3.0      5577
         2.0      1676
         1.0      1018
         Name: Averagerating, dtype: int64
```

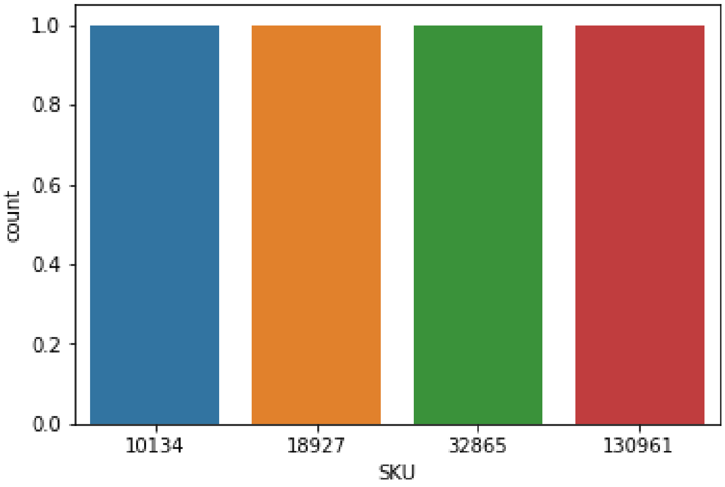
SKU countplots

In [17]:

```
sns.countplot(df1.groupby('Quartile')['SKU'].count())
```

Out[17]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1d35ac76198>
```



In [18]:

```
X=df1.drop('Quartile',axis=1)
y=df1['Quartile']
```

In [19]:

```
X.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 192887 entries, 0 to 192899
Data columns (total 20 columns):
Class ID                     192887 non-null float64
Color                        192887 non-null object
Count Of Big Transactions    192887 non-null float64
Country Of Origin            192887 non-null object
Depth                        192887 non-null float64
Height                       192887 non-null float64
Life Cycle Name              192887 non-null object
Product Name                 192887 non-null object
SKU                          192887 non-null float64
Sentiment                    192887 non-null float64
Category                     192887 non-null object
Weight                       192887 non-null float64
Width                        192887 non-null float64
Answercount                  192887 non-null float64
Averagerating                192887 non-null float64
Commentcount                 192887 non-null float64
Online-Flag                  192887 non-null bool
Questioncount                192887 non-null float64
Reviewcount                  192887 non-null float64
Text                         192887 non-null object
dtypes: bool(1), float64(13), object(6)
memory usage: 29.6+ MB
```

In [20]:

```
X.columns
```

Out[20]:

```
Index(['Class ID', 'Color', 'Count Of Big Transactions', 'Country Of Origin',
       'Depth', 'Height', 'Life Cycle Name', 'Product Name', 'SKU',
       'Sentiment', 'Category', 'Weight', 'Width', 'Answercount',
       'Averagerating', 'Commentcount', 'Online-Flag', 'Questioncount',
       'Reviewcount', 'Text'],
      dtype='object')
```

In [21]:

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()

for i in ['Color', 'Country Of Origin', 'Life Cycle Name','Product Name','Category', 'Online-Flag', 'Text']:
    X[i]=le.fit_transform(X[i])
```

In [22]:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state=42)
```

In [23]:

```
X.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 192887 entries, 0 to 192899
Data columns (total 20 columns):
Class ID                     192887 non-null float64
Color                        192887 non-null int32
Count Of Big Transactions    192887 non-null float64
Country Of Origin            192887 non-null int32
Depth                        192887 non-null float64
Height                       192887 non-null float64
Life Cycle Name              192887 non-null int32
Product Name                 192887 non-null int32
SKU                          192887 non-null float64
Sentiment                    192887 non-null float64
Category                     192887 non-null int32
Weight                       192887 non-null float64
Width                        192887 non-null float64
Answercount                  192887 non-null float64
Averagerating                192887 non-null float64
Commentcount                 192887 non-null float64
Online-Flag                  192887 non-null int64
Questioncount                192887 non-null float64
Reviewcount                  192887 non-null float64
Text                         192887 non-null int32
dtypes: float64(13), int32(6), int64(1)
memory usage: 26.5 MB
```

Random Forest

In [24]:

```
from sklearn.ensemble import RandomForestClassifier
rtree=RandomForestClassifier()
rtree.fit(X_train,y_train)
```

```
C:\Users\priya\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:246: Future
Warning: The default value of n_estimators will change from 10 in version 0.20 to
100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
```

Out[24]:

```
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
            max_depth=None, max_features='auto', max_leaf_nodes=None,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=10, n_jobs=None,
            oob_score=False, random_state=None, verbose=0,
            warm_start=False)
```

Important Features

In [25]:

```
pd.DataFrame(rtree.feature_importances_,index=X_train.columns,columns = ['importan
ce'])\
.sort_values(by='importance',ascending=False)
```

Out[25]:

	importance
Count Of Big Transactions	0.291397
Reviewcount	0.104494
SKU	0.091415
Product Name	0.084711
Weight	0.077813
Commentcount	0.071338
Class ID	0.050165
Width	0.047565
Height	0.043113
Depth	0.038208
Color	0.023161
Averagerating	0.017307
Category	0.014648
Online-Flag	0.013112
Country Of Origin	0.012103
Text	0.010175
Sentiment	0.009249
Life Cycle Name	0.000028
Answercount	0.000000
Questioncount	0.000000

Decision Tree Classifier

In [55]:

```
from sklearn.tree import DecisionTreeClassifier
clf=DecisionTreeClassifier(max_leaf_nodes=12)
clf.fit(X_train,y_train)
```

Out[55]:

```
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
            max_features=None, max_leaf_nodes=12,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, presort=False, random_state=None,
            splitter='best')
```

In [56]:

```
pd.DataFrame(clf.feature_importances_,index=X_train.columns,columns = ['importanc
e'])\
.sort_values(by='importance',ascending=False)
```

Out[56]:

	importance
Count Of Big Transactions	0.658586
Weight	0.229362
Reviewcount	0.071321
Commentcount	0.040731
Class ID	0.000000
Questioncount	0.000000
Online-Flag	0.000000
Averagerating	0.000000
Answercount	0.000000
Width	0.000000
Category	0.000000
Color	0.000000
Sentiment	0.000000
SKU	0.000000
Product Name	0.000000
Life Cycle Name	0.000000
Height	0.000000
Depth	0.000000
Country Of Origin	0.000000
Text	0.000000

In [57]:

```
from sklearn.metrics import confusion_matrix, classification_report, r2_score
from sklearn.metrics import f1_score
```

In [58]:

```
def score(model, test = X_test, y_true = y_test):

    pred = model.predict(test)

    print("R-Squared Score:\t",round(r2_score(y_true,pred),4)*100)
    print()
    print(classification_report(y_true,pred))
```

Classification Scores

In [59]:

```
score(clf)
```

```
R-Squared Score:         68.73

              precision    recall  f1-score   support

         1.0       0.82      0.37      0.51      3326
         2.0       0.50      0.59      0.54      6346
         3.0       0.54      0.56      0.55     10789
         4.0       0.93      0.94      0.93     43192

   micro avg       0.81      0.81      0.81     63653
   macro avg       0.70      0.62      0.63     63653
weighted avg       0.82      0.81      0.81     63653
```

Randomforest Score

In [32]:

```
score(rtree)
```

```
R-Squared Score:         97.61999999999999

              precision    recall  f1-score   support

         1.0       0.94      0.95      0.95      3326
         2.0       0.94      0.94      0.94      6346
         3.0       0.96      0.97      0.97     10789
         4.0       1.00      1.00      1.00     43192

   micro avg       0.98      0.98      0.98     63653
   macro avg       0.96      0.96      0.96     63653
weighted avg       0.98      0.98      0.98     63653
```

In [35]:

```
import graphviz
from sklearn import tree
```

In [40]:

```
dot_data = tree.export_graphviz(clf, out_file=None)
graph = graphviz.Source(dot_data)
graph.render("iris")
```

Out[40]:

```
'iris.pdf'
```

In [47]:

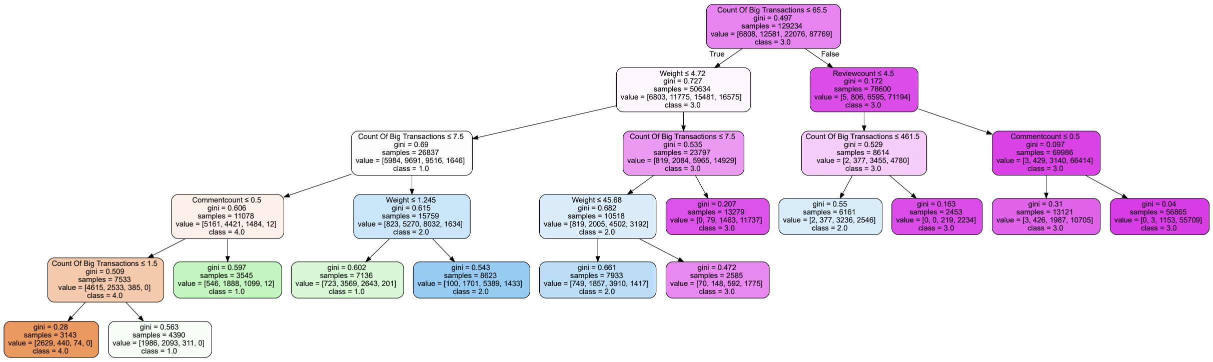
```
target = df1['Quartile'].unique()
for i in target: i = str(i)
```

Out[47]:

```
numpy.float64
```

In [60]:
```
dot_data = tree.export_graphviz(clf, out_file=None,
feature_names=X_train.columns,
class_names=target.astype(str),
filled=True, rounded=True,
special_characters=True)
graph = graphviz.Source(dot_data)
graph
```

Out[60]:


Count Of Big Transactions ≤ 65.5
gini = 0.497
samples = 129234
value = [6808, 12581, 22076, 87769]
class = 3.0
True
False
Weight ≤ 4.72
gini = 0.727
samples = 50634
value = [6803, 11775, 15481, 16575]
class = 3.0
Reviewcount ≤ 4.5
gini = 0.172
samples = 78600
value = [5, 806, 6595, 71194]
class = 3.0
Count Of Big Transactions ≤ 7.5
gini = 0.69
samples = 26837
value = [5984, 9691, 9516, 1646]
class = 1.0
Count Of Big Transactions ≤ 7.5
gini = 0.535
samples = 23797
value = [819, 2084, 5965, 14929]
class = 3.0
Count Of Big Transactions ≤ 461.5
gini = 0.529
samples = 8614
value = [2, 377, 3455, 4780]
class = 3.0
Commentcount ≤ 0.5
gini = 0.097
samples = 69986
value = [3, 429, 3140, 66414]
class = 3.0
Commentcount ≤ 0.5
gini = 0.606
samples = 11078
value = [5161, 4421, 1484, 12]
class = 4.0
Weight ≤ 1.245
gini = 0.615
samples = 15759
value = [823, 5270, 8032, 1634]
class = 2.0
Weight ≤ 45.68
gini = 0.682
samples = 10518
value = [819, 2005, 4502, 3192]
class = 2.0
gini = 0.207
samples = 13279
value = [0, 79, 1463, 11737]
class = 3.0
gini = 0.55
samples = 6161
value = [2, 377, 3236, 2546]
class = 2.0
gini = 0.163
samples = 2453
value = [0, 0, 219, 2234]
class = 3.0
gini = 0.31
samples = 13121
value = [3, 426, 1987, 10705]
class = 3.0
gini = 0.04
samples = 56865
value = [0, 3, 1153, 55709]
class = 3.0
Count Of Big Transactions ≤ 1.5
gini = 0.509
samples = 7533
value = [4615, 2533, 385, 0]
class = 4.0
gini = 0.597
samples = 3545
value = [546, 1888, 1099, 12]
class = 1.0
gini = 0.602
samples = 7136
value = [723, 3569, 2643, 201]
class = 1.0
gini = 0.543
samples = 8623
value = [100, 1701, 5389, 1433]
class = 2.0
gini = 0.661
samples = 7933
value = [749, 1857, 3910, 1417]
class = 2.0
gini = 0.472
samples = 2585
value = [70, 148, 592, 1775]
class = 3.0
gini = 0.28
samples = 3143
value = [2629, 440, 74, 0]
class = 4.0
gini = 0.563
samples = 4390
value = [1986, 2093, 311, 0]
class = 1.0