```
from google.colab import drive
drive.mount('/content/gdrive')  #Mounting google drive
```

```
Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=947318989803-6bn6qk8qdgf4n4g3pfee6491hc0brc4i.a

Enter your authorization code:
··········
Mounted at /content/gdrive
```

```
import numpy as np
import pandas as pd  #importing libraries
```

Creating dataframe from Phase2 Dataset

```
Data = pd.read_excel("gdrive/My Drive/ml_hack/IndiaMart/Hackathon Price data3169da (3).xlsx", sheet_name=0)#reading excel file sheet0
```

```
#dropping rows having null values
data=Data.dropna()
```

```
data
```

Method Applied for Outlier Detection

Boosting method with the data passed through **model 1** and the refined data from model 1 is refined further using **model 2**.

Models used for refining:

1. **IQR(Inter-Quartile Range)** Method
2. **Isolation Forest** Method

Model 1: Outlier Detection using **IQR**(Inter-Quartile Range) Method

```
import numpy as np

def outliers_iqr(ys):
    quartile_1, quartile_3 = np.percentile(ys, [25, 75])
    iqr = quartile_3 - quartile_1
    lower_bound = quartile_1 - (iqr * 1.5)
    upper_bound = quartile_3 + (iqr * 1.5)
    result=[]
    for i in ys:
      if(i<=upper_bound and i>=lower_bound):
        result.append(i)
    return result
```

Model 2: Isolation Forest Method

```
#Isolation forest outlier detection
from sklearn.ensemble import IsolationForest
import pandas as pd

def isolationForest(ys):
  clf = IsolationForest(max_samples=100, random_state=42)

  clf.fit(ys)
  output_table =(clf.predict(ys))
  c=0
  result=[]
  for i in output_table:
    if(i!=-1):
      result.append(ys[c,0])
    c=c+1
  return result
#Predicting outliers using isolation forest
```

MCAT-UNIT wise MIN-MAX price and **MCAT - Unit - ISQ - ISQ Option** wise --- Min Max price

---

```
mcat=list(set(data["Mcat Name"]))
print(len(mcat))
```

```
100
```

```
mcat_isq=[]
unit_isq=[]
isq_name=[]
isq_option=[]
```

```
prices=[]


isq_prices=[]


for i in (mcat):
  for k in list(set(data[data["Mcat Name"]==i]["PC_ITEM_MOQ_UNIT_TYPE"])):
    for l in (list(set(data[data["Mcat Name"]==i][data["PC_ITEM_MOQ_UNIT_TYPE"]==k]["FK_IM_SPEC_MASTER_DESC"]))):
      for m in(list(set(data[data["Mcat Name"]==i][data["PC_ITEM_MOQ_UNIT_TYPE"]==k][data["FK_IM_SPEC_MASTER_DESC"]==l]["FK_IM_SPEC_OPTIONS_DES
        isq_prices.append(list((data[data["Mcat Name"]==i][data["PC_ITEM_MOQ_UNIT_TYPE"]==k][data["FK_IM_SPEC_MASTER_DESC"]==l][data["FK_IM_SPE
        mcat_isq.append(i)
        unit_isq.append(k)
        isq_name.append(l)
        isq_option.append(m)
        prices.append(list(data[data["Mcat Name"]==i][data["PC_ITEM_MOQ_UNIT_TYPE"]==k]["PC_ITEM_FOB_PRICE"]))
```

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:3: UserWarning: Boolean Series key will be reindexed to match DataF
  This is separate from the ipykernel package so we can avoid doing imports until
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:4: UserWarning: Boolean Series key will be reindexed to match DataF
  after removing the cwd from sys.path.
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:5: UserWarning: Boolean Series key will be reindexed to match DataF
  """
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:10: UserWarning: Boolean Series key will be reindexed to match Data
  # Remove the CWD from sys.path while we load stuff.
```

```
len(isq_prices)
```

```
15665
```

```
len(prices)
```

```
15665
```

```
prices_refined1=[]
prices_refined2=[]


isq_prices_refined1=[]
isq_prices_refined2=[]
```

(Refined prices 1 ,Refined prices 2) and (Refined Isq_prices 1 ,Refined Isq_prices 2) after passing the prices through model 1 and model 2 of Boosting

```
for i in prices:
  prices_refined1.append(outliers_iqr((i)))



for i in isq_prices:

  isq_prices_refined1.append(outliers_iqr((i)))



for i in prices_refined1:
  prices_refined2.append(isolationForest((np.asarray(i)).reshape(-1,1)))
```

```
len(prices_refined2)
```

```
15665
```

```
for i in isq_prices_refined1:
  isq_prices_refined2.append(isolationForest((np.asarray(i)).reshape(-1,1)))


len(isq_prices_refined2)
```

```
15665
```

```
price_max=[]
price_min=[]


isq_price_max=[]
isq_price_min=[]


for i in prices_refined2:
  price_max.append(max(i))
  price_min.append(min(i))


for i in isq_prices_refined2:
  isq_price_max.append(max(i))
  isq_price_min.append(min(i))
```

MCAT Unit wise ---- Three most common prices

```
len(prices_refined2)
```

```
15665
```

```
import collections
def mostcommon(a):
  counter=collections.Counter(a)
  l=counter.most_common(3)
  result=[]
  for i in range(len(l)):
    result.append(l[i][0])
  for j in range(3-len(l)):
    result.append("NULL")
  return result
```

```
most_common1=[]
most_common2=[]
most_common3=[]
```

```
for i in prices_refined2:
  most_common1.append(mostcommon(i)[0])
  most_common2.append(mostcommon(i)[1])
  most_common3.append(mostcommon(i)[2])
```

Final result Dataframe(containing all the required prices) is stored in csv format

```
final_result = pd.DataFrame({'MCAT Name':mcat_isq, 'Unit':unit_isq,'Unit wise Min Price':price_min,'Unit wise Max Price':price_max,'Most Common
```

```
final_result.to_csv('gdrive/My Drive/ml_hack/IndiaMart/final_result.csv', index=False)
```