SPEND ANALYTICS

KPMG Capstone project -1

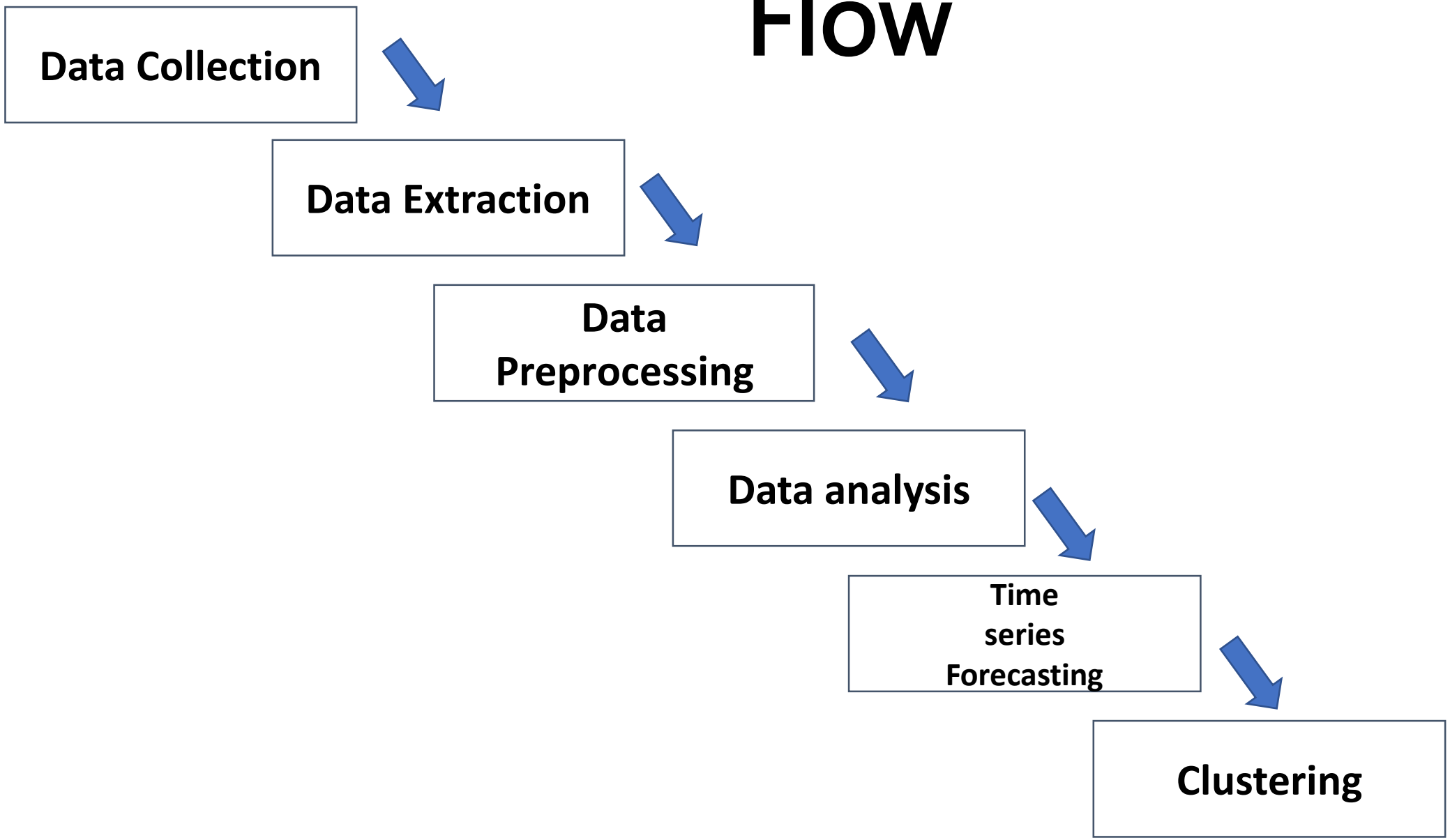
Done by,

Ishwarya M

Objectives:

1. Analyze the data and identify purchasing trends and patterns
2. Identify the cost saving opportunities by using the data of procurement
3. Cluster items that have similar purchasing patterns
4. Create dashboards using any Visualization tool; e.g. Tableau
5. Make a report describing all the findings.

Work Flow

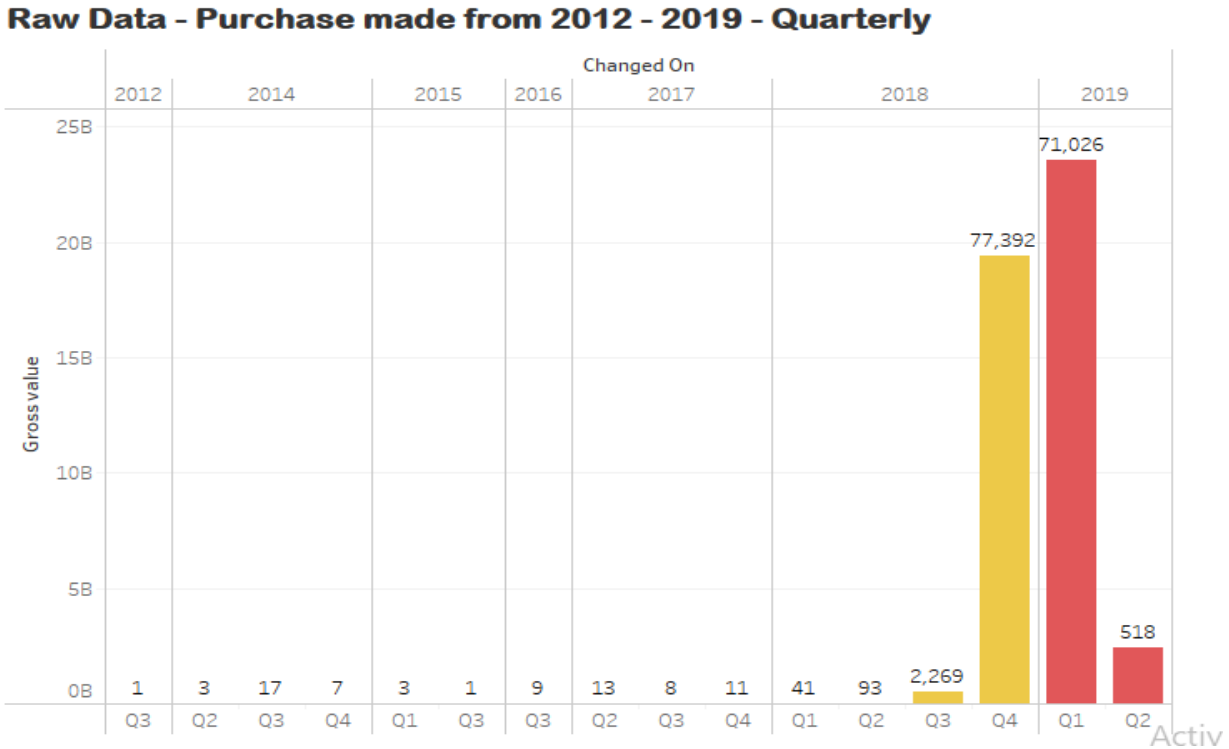

Data Collection
Data Extraction
Data Preprocessing
Data analysis
Time series Forecasting
Clustering

Data Collection

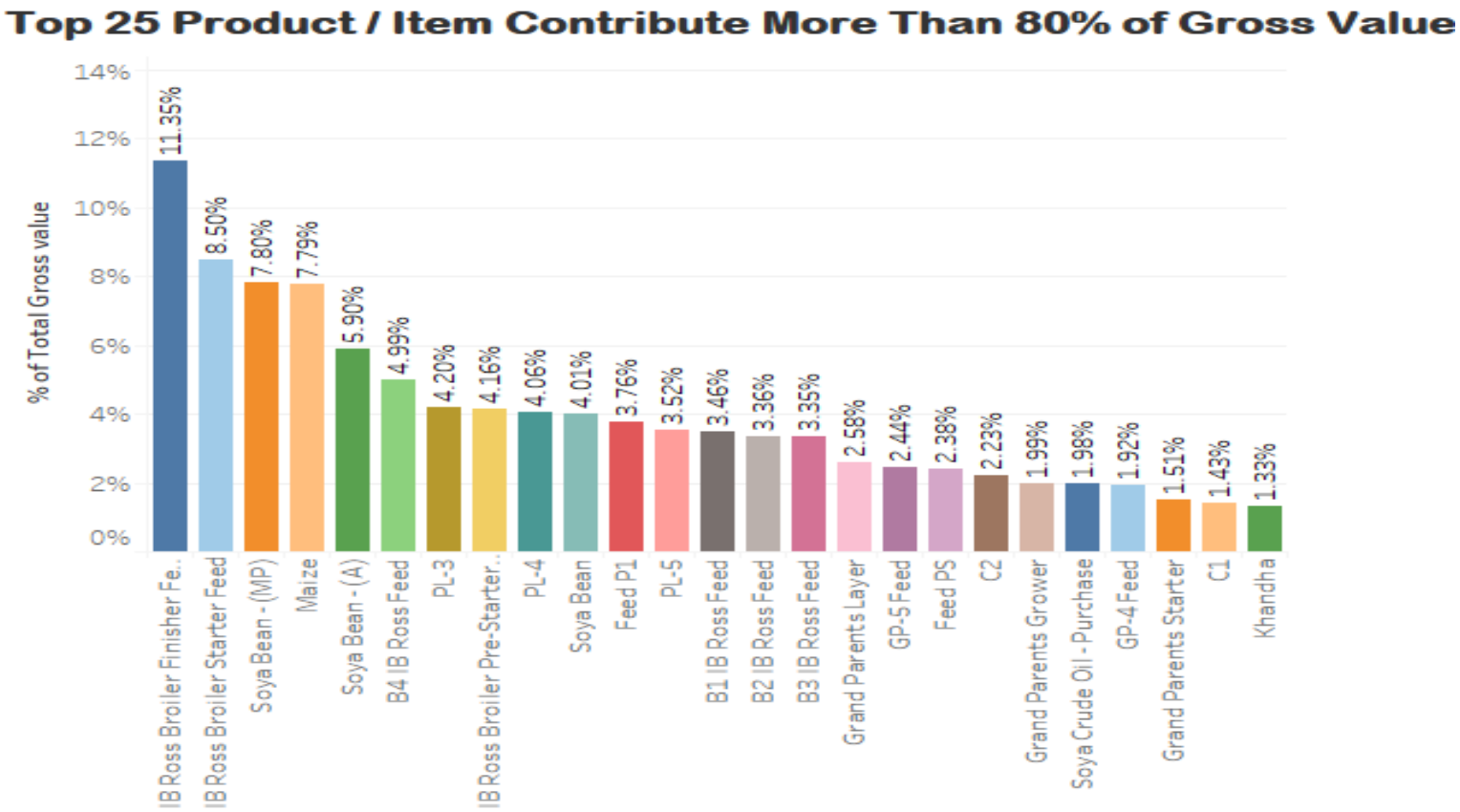
- The data we use here is collected from SAP and exported into Excel file(.xlsx format)
- We have been provided with a single file which contains data related to procurement transactions for the organization, with several features for analysis.
- It has totally
    - 65 Columns
    - 151412 Rows

Data Extraction

- In this process we extracted the columns which are significant for our analysis purpose.
- Data's are available since 2012 till 2019 however there is no connectivity till 2016 and 2017 has very less volume of purchase. Hence, 2018 and 2019 were considered / extracted for the purpose of Analysis.



- Data was further drilled down and Top 25 Products / Item (Short Text) that contributes to 80% and above of Total Gross Value were extracted using MS Excel.
- There were Missing Values in the Material column however during the above extraction process the rows containing missing values were removed based on the criteria given.
- Finally we gotten a data with 68523 Rows and 11 Columns.

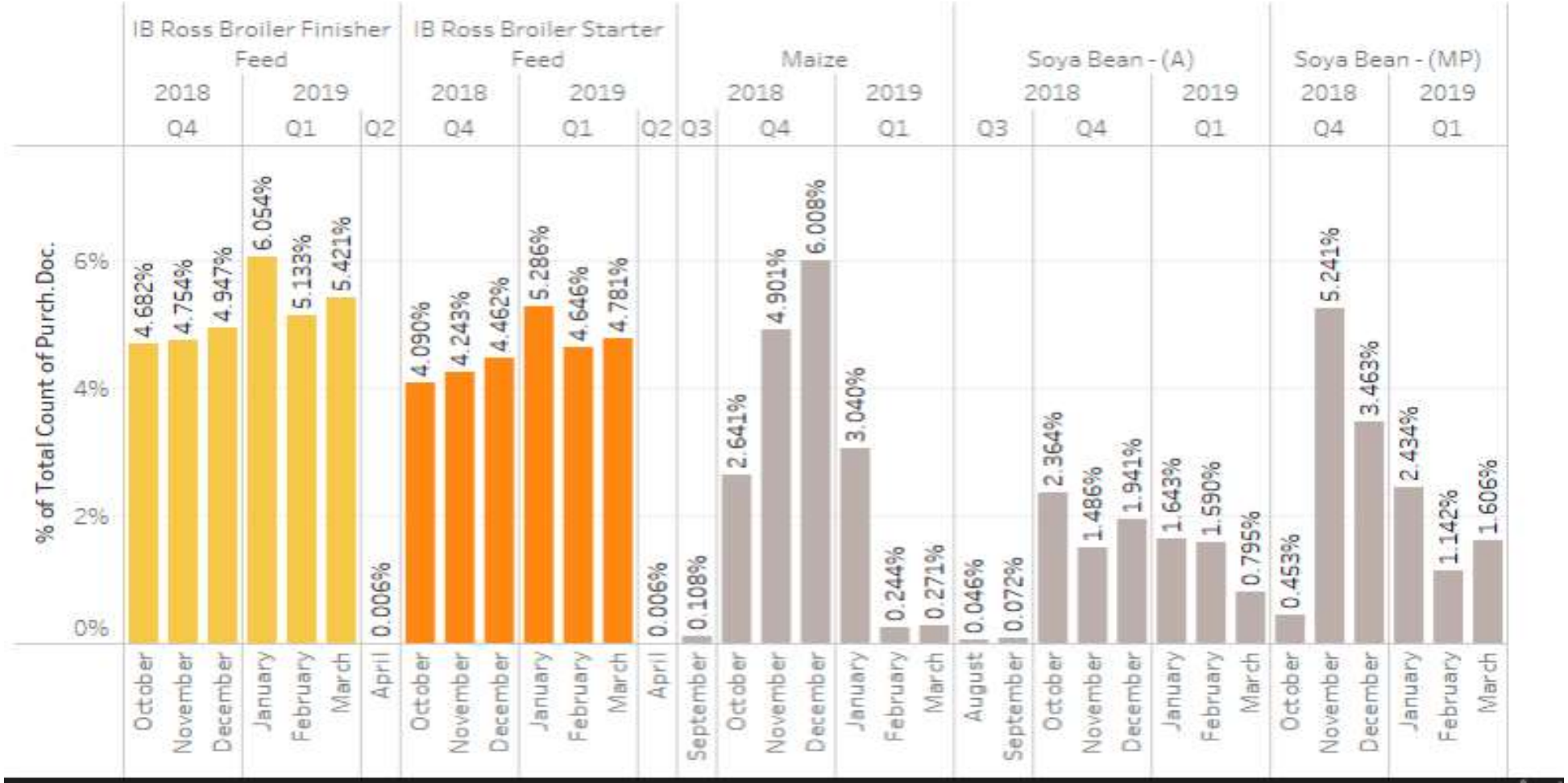


Data Preprocessing

- This entire Data preprocessing is done by using Pandas library in Python.
- Initially done Type casting the columns to its respective Data types
- Checked for missing values. The data has no missing values
- Then we add new column called **Net price** by dividing **Gross Value** with the **PO quantity** of our original data.

Data Analysis

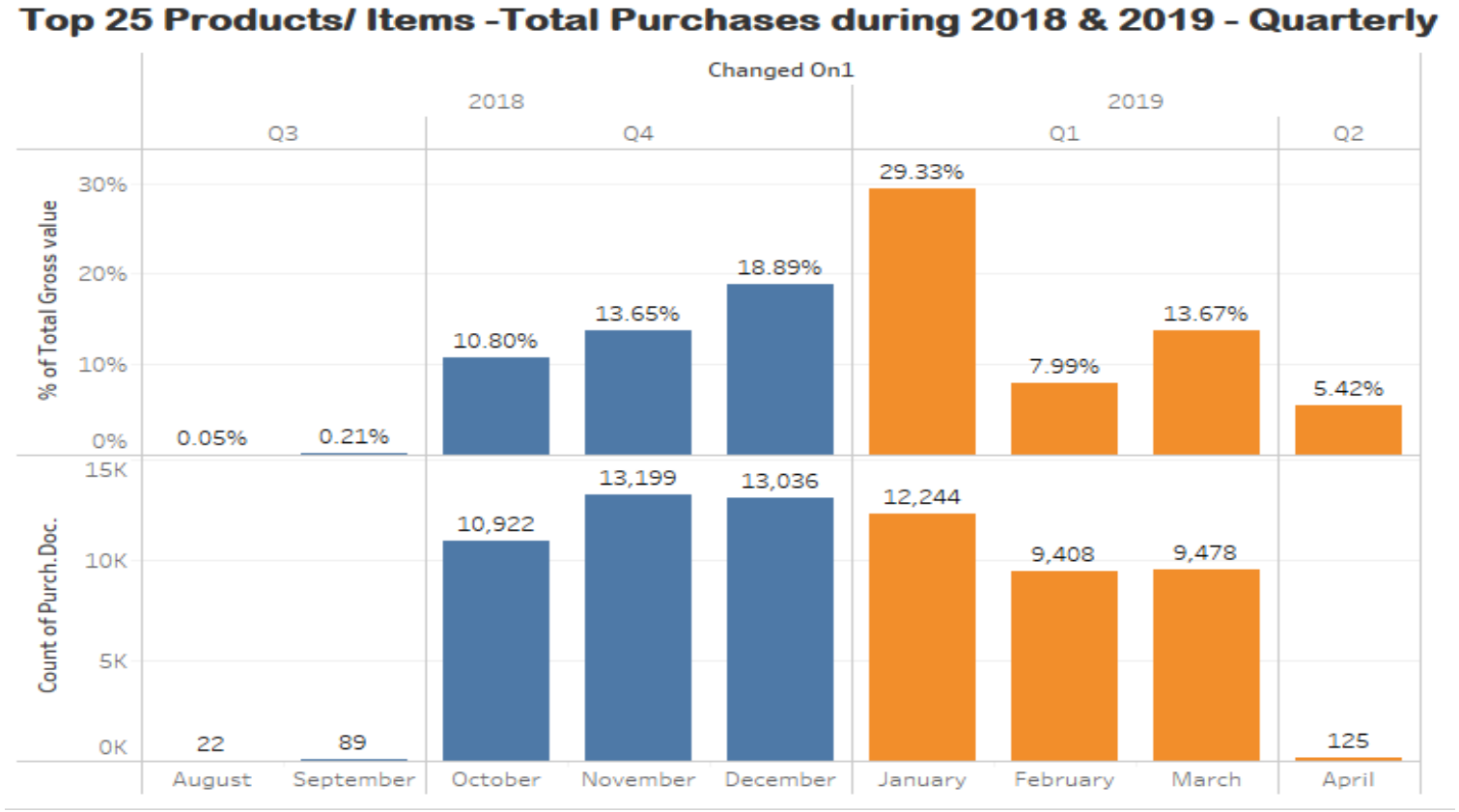
Similar purchase pattern were visualized – IB Ross Broiler Finisher Feed and IB Ross Broiler Starter Feed has similar purchasing pattern during Q4 2018 & Q1 2019 and the same was also proven using clustering techniques.



Quarter wise Analysis

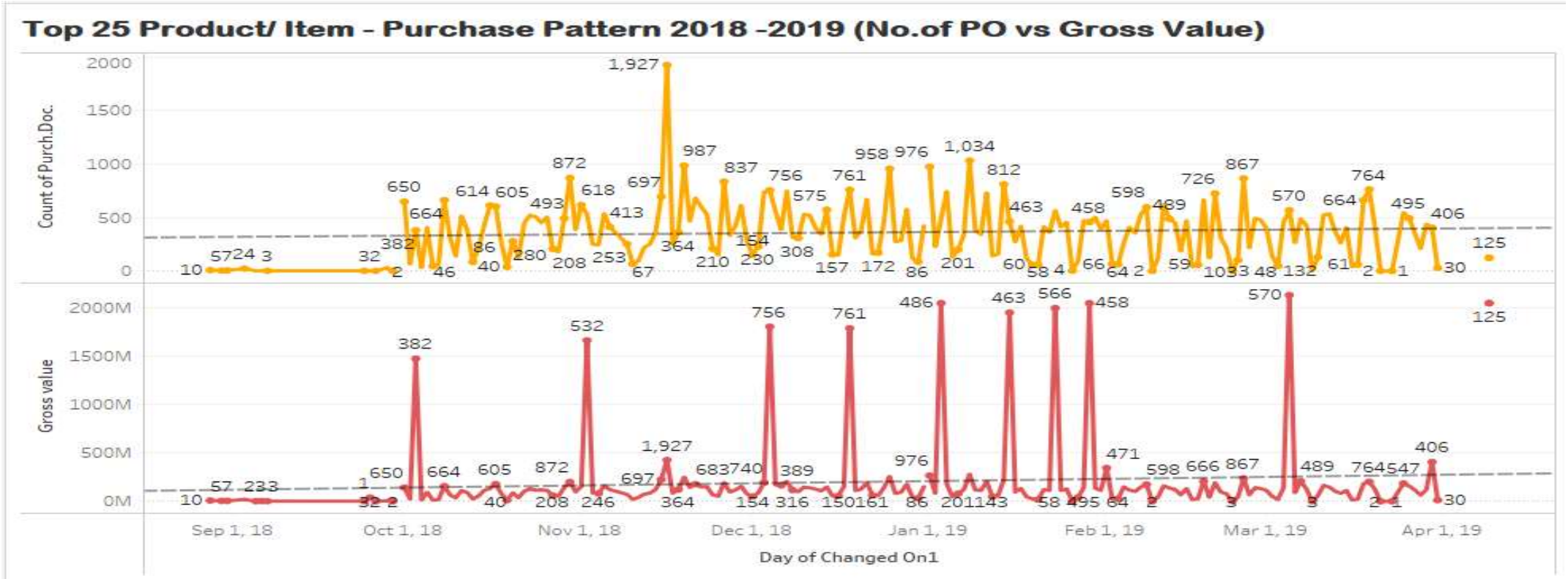
Analysis were performed on the data extracted (Rows – 68523 & 12 column (includes 1 new column added during preprocessing)) and the observations are as below.

- Huge volume of Purchases were made during the year 2018 & 2019, specifically in Q4 2018 & Q1 2019.
- In Q2 2019 very few purchases were made on 9th April 2019.



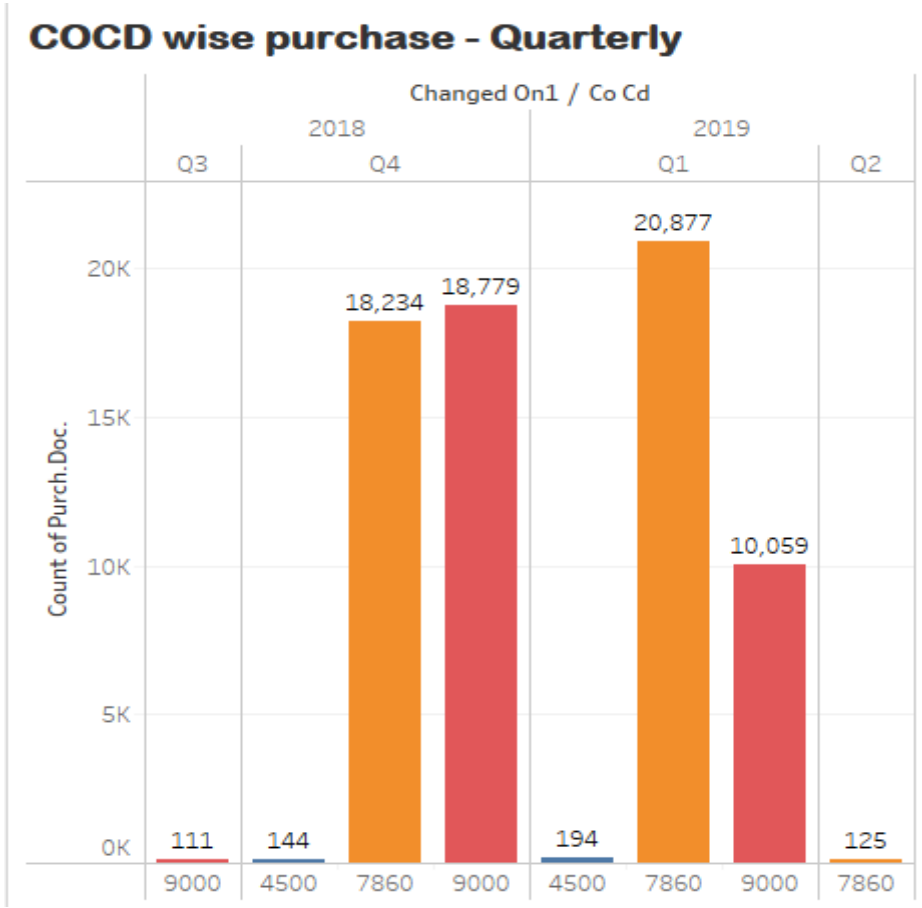
Day wise Analysis

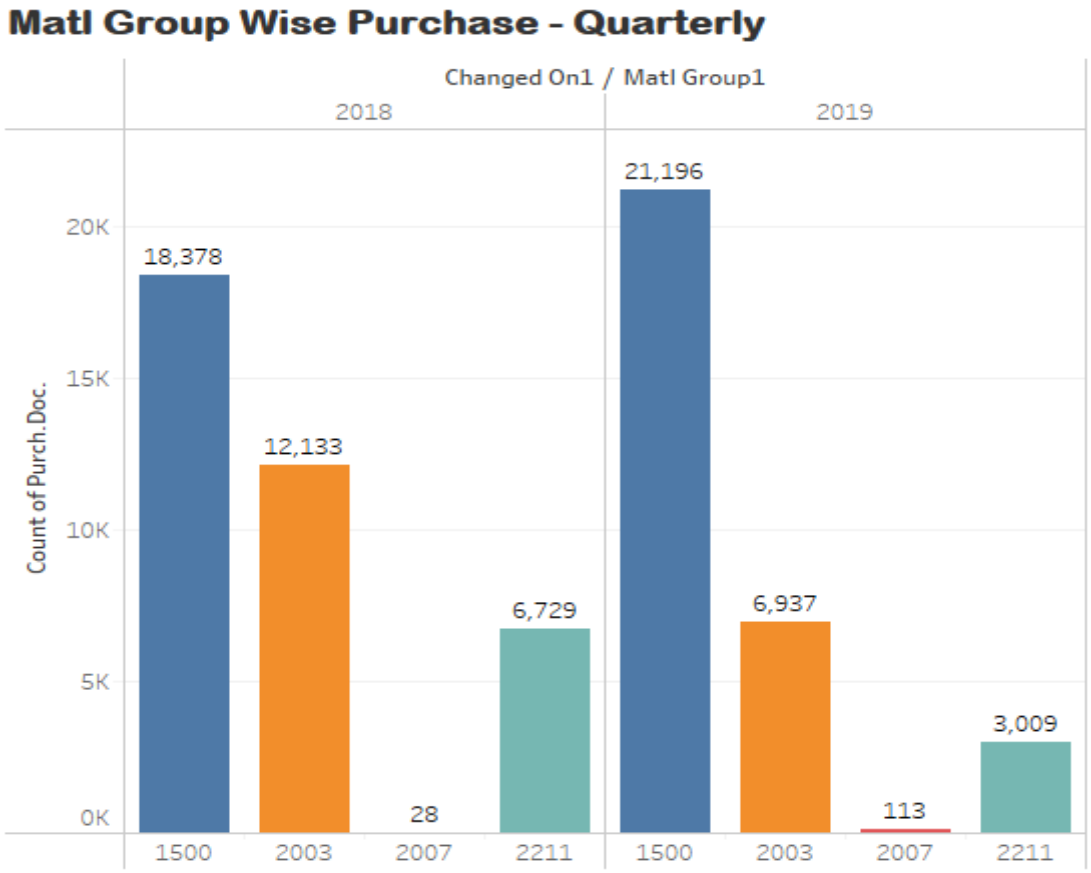
- Further drill down day wise helps to view the spike in purchase since 1st October 2018 till 31st March 2019 and the highest number of Purchase Order were raised on 4th November 2018 with a count of 1927.
- Net price of few products like Soya Bean, Soya Bean (A), Soya Bean(MP), Soya Crude Oil are on the higher side, though the number of Purchase order raised was less the Gross Value has seen a spike.



Purchase location and Material group Analysis

- Purchases were made by different sub locations of COCD -4500,7860,9000.
- Products /Items purchased falls under 4 Material Group (1500,2003,2007 & 2211)

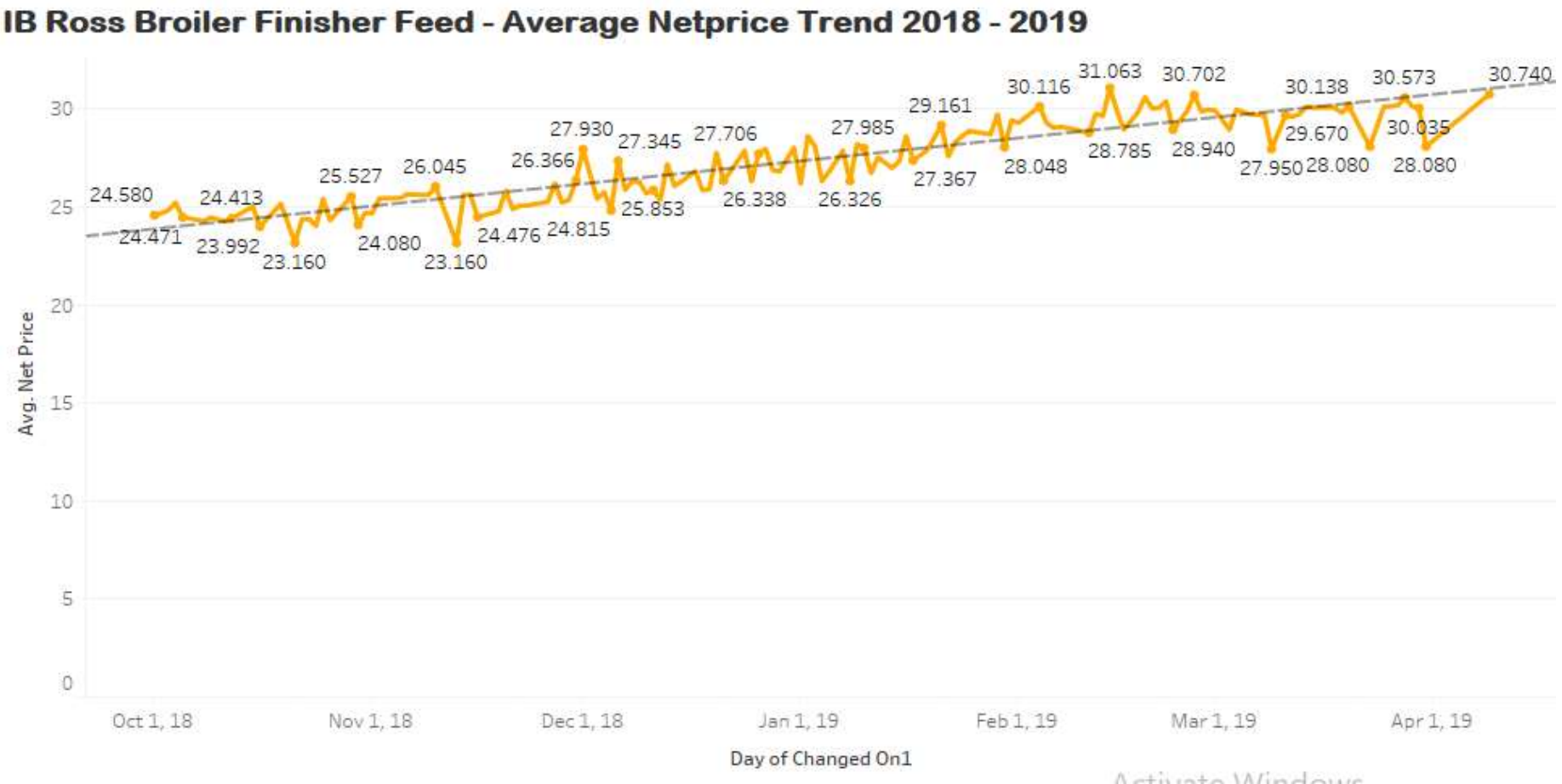




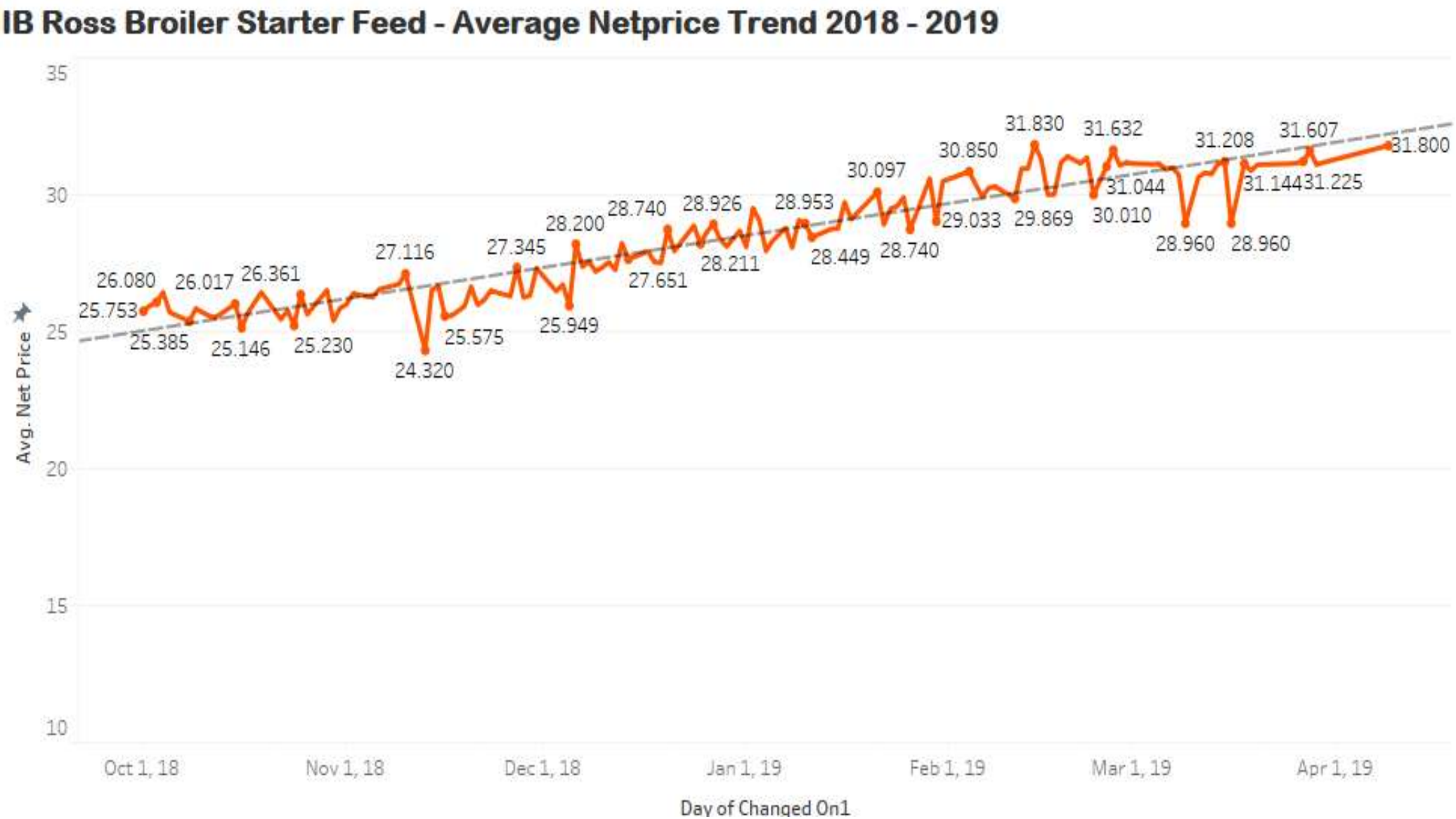
Net Price Trend for Top 5 Products

Net price were calculated for Top 25 product / item, to see the trend of the net price. However due to volume of data, only Top 5 product / Item were chosen to visualize net price pattern and forecasting since those items give highest contribution to the Gross value..

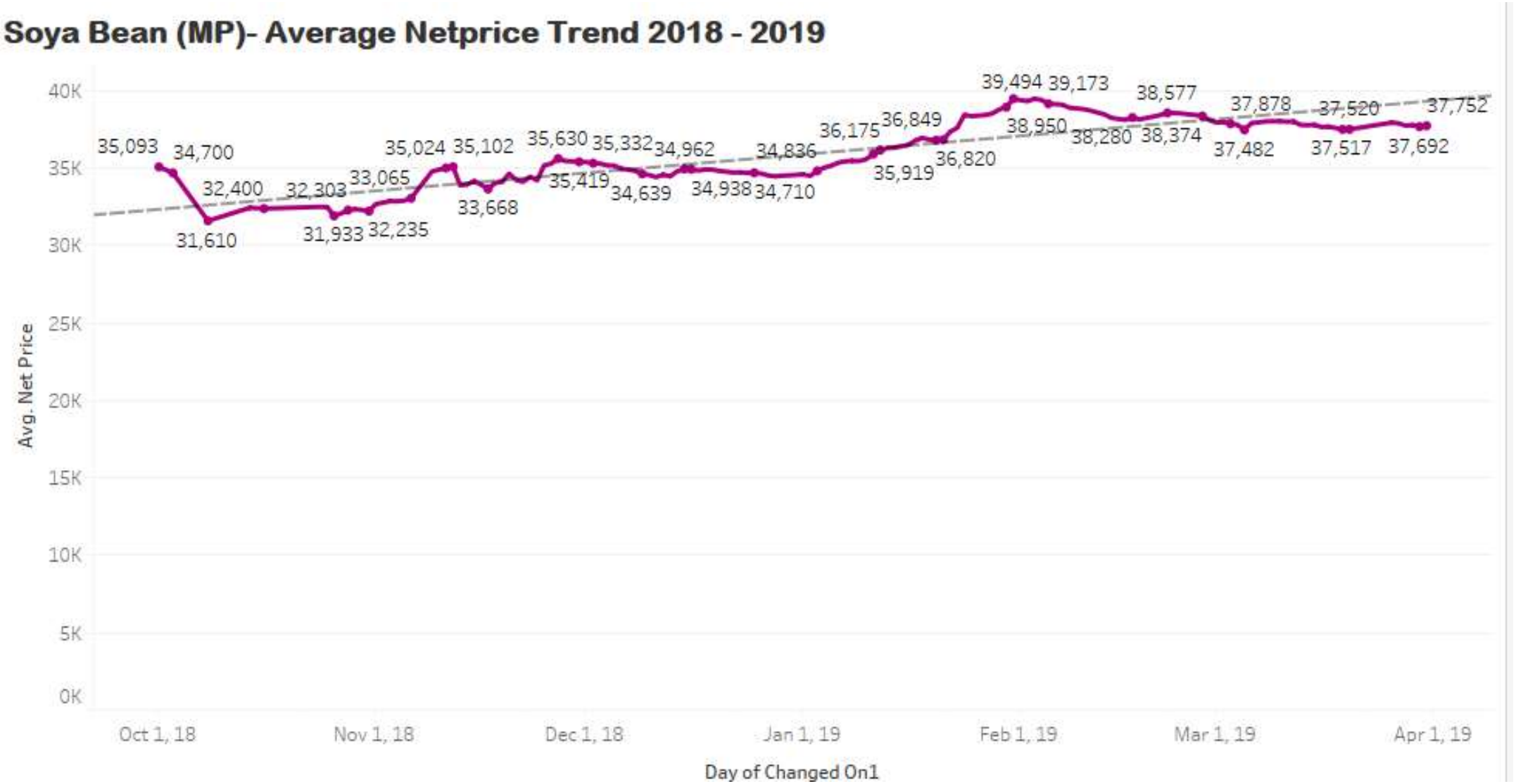
1.



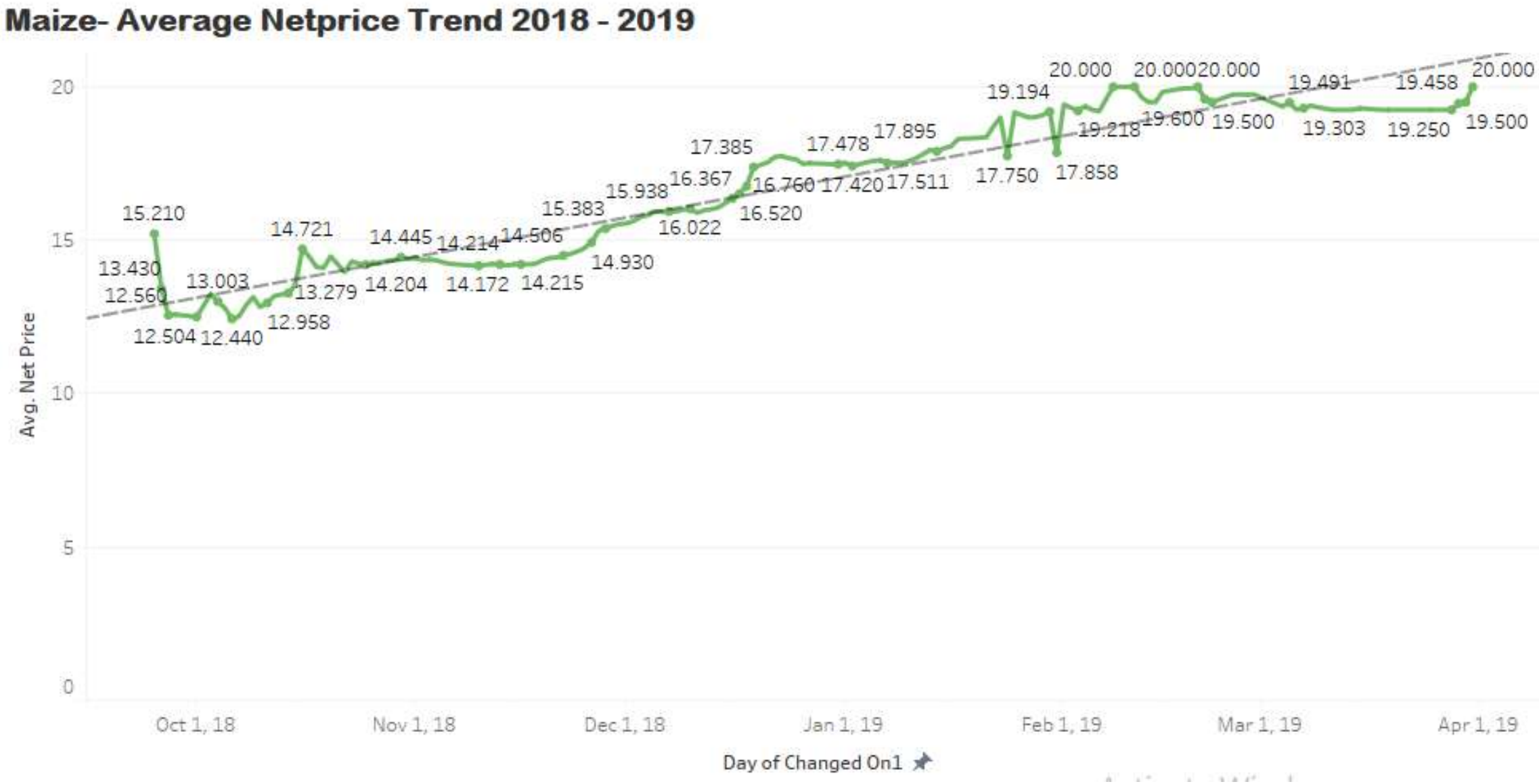
2.


IB Ross Broiler Starter Feed - Average Netprice Trend 2018 - 2019
Avg. Net Price
35
30
25
20
15
10
26.080
26.017
26.361
25.753
25.385
25.146
25.230
27.116
24.320
27.345
25.575
28.200
25.949
28.740
27.651
28.926
28.211
28.953
28.449
30.097
28.740
30.850
29.033
31.830
29.869
31.632
31.044
30.010
31.208
28.960
28.960
31.607
31.144
31.225
31.800
Oct 1, 18
Nov 1, 18
Dec 1, 18
Jan 1, 19
Feb 1, 19
Mar 1, 19
Apr 1, 19
Day of Changed On1

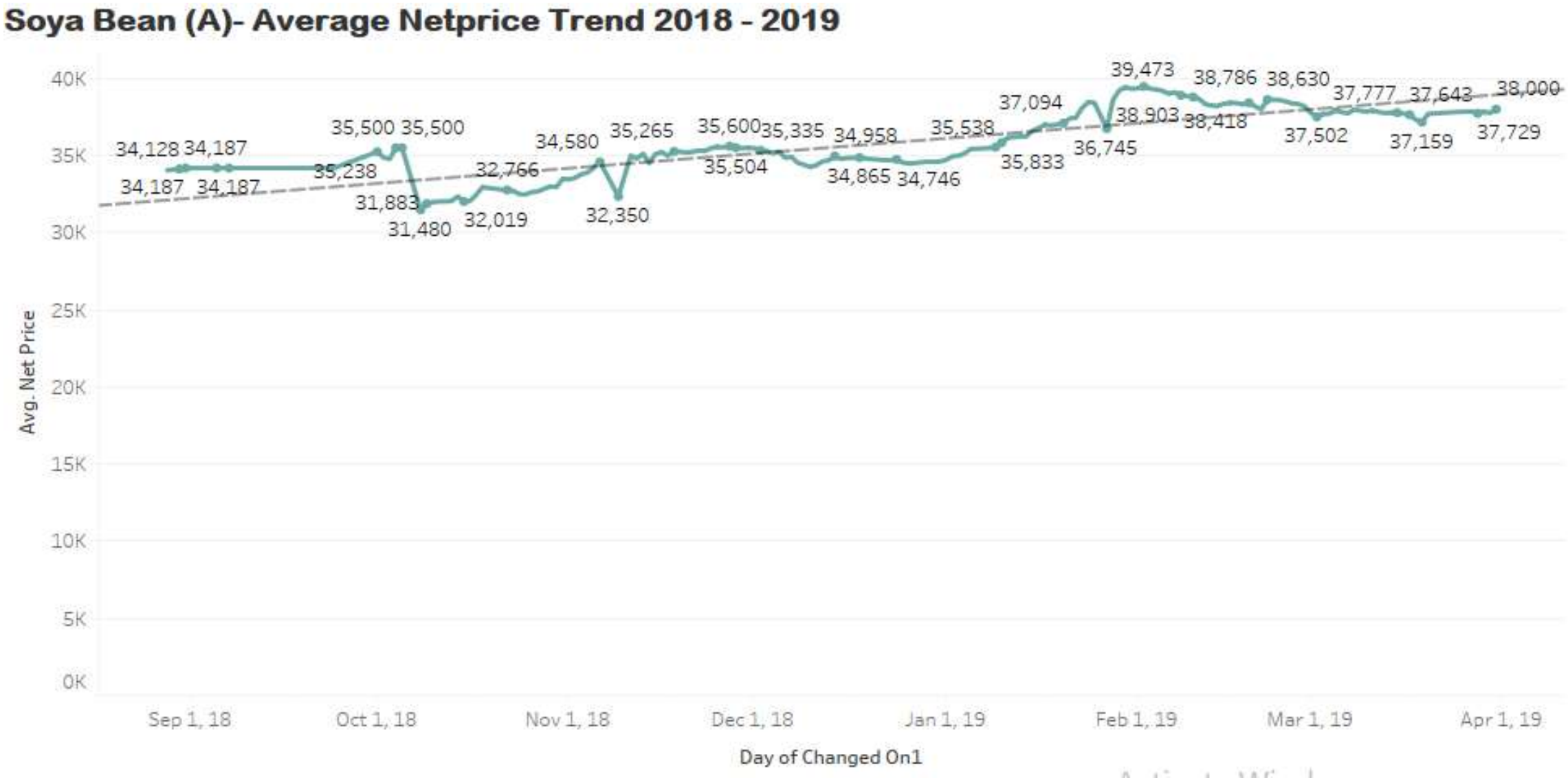
3.


Soya Bean (MP)- Average Netprice Trend 2018 - 2019
Avg. Net Price
40K
35K
30K
25K
20K
15K
10K
5K
0K
35,093
34,700
31,610
32,400
32,303
31,933
33,065
32,235
35,024
33,668
35,102
35,630
35,419
35,332
34,639
34,962
34,938
34,836
34,710
36,175
35,919
36,849
36,820
39,494
38,950
39,173
38,280
38,577
38,374
37,878
37,482
37,520
37,517
37,752
37,692
Oct 1, 18
Nov 1, 18
Dec 1, 18
Jan 1, 19
Feb 1, 19
Mar 1, 19
Apr 1, 19
Day of Changed On1

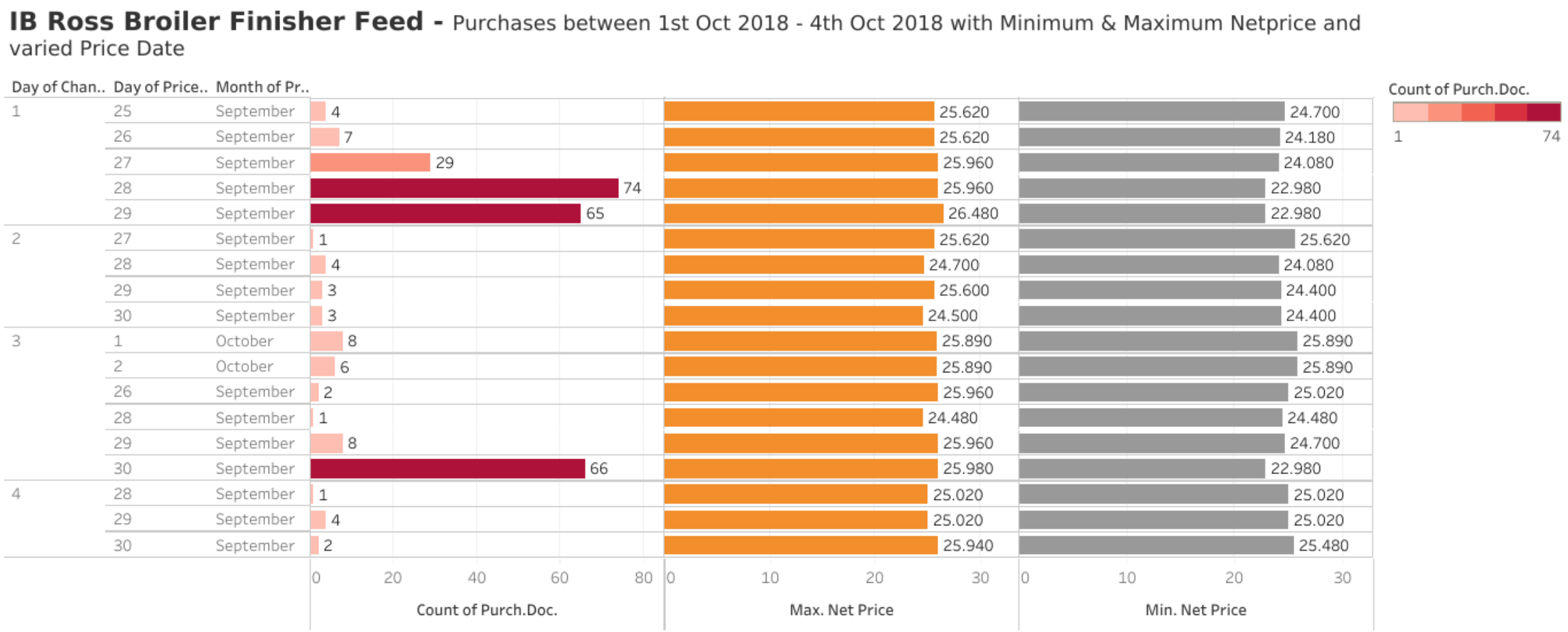
4.


Maize- Average Netprice Trend 2018 - 2019
Avg. Net Price
20
15
10
5
0
15.210
13.430
12.560
12.504
13.003
12.440
14.721
13.279
12.958
14.445
14.204
14.214
14.172
14.506
14.215
15.383
14.930
15.938
16.022
16.367
16.520
17.385
16.760
17.478
17.420
17.895
17.511
19.194
17.750
20.000
17.858
19.218
20.000
19.600
20.000
19.500
19.491
19.303
19.458
19.250
20.000
19.500
Oct 1, 18
Nov 1, 18
Dec 1, 18
Jan 1, 19
Feb 1, 19
Mar 1, 19
Apr 1, 19
Day of Changed On1

5.


Soya Bean (A)- Average Netprice Trend 2018 - 2019
40K
35K
30K
25K
20K
15K
10K
5K
0K
Avg. Net Price
34,128
34,187
34,187
34,187
35,238
35,500
35,500
31,883
31,480
32,019
32,766
34,580
32,350
35,265
35,600
35,504
35,335
34,958
34,865
34,746
35,538
35,833
37,094
36,745
39,473
38,903
38,786
38,418
38,630
37,502
37,777
37,159
37,643
37,729
38,000
Sep 1, 18
Oct 1, 18
Nov 1, 18
Dec 1, 18
Jan 1, 19
Feb 1, 19
Mar 1, 19
Apr 1, 19
Day of Changed On1

Noted Price date and net price were not constant for the purchases made during a single day for an Item. For Example: Purchases during 1/10/2018 to 4/10/2018 of IB Ross Broiler Finisher Feed item/ product were taken. Noted that purchases orders has a price range from 22.980 – 26.480 with varied price date.



Time series Forecasting

- For the further time series forecasting, we take the Top 5 items that contributed more towards Volume(Maximum purchased) and Gross Value(Maximum Expenditure) . For the steps explanation first we will go with forecasting the net price value of Soya Bean (A) and rest 4 items follow the same procedure.
- For doing time series forecasting , we filter the two columns which are Time series element and Net price from the Dataset. Since it is the only required variables for time series analysis.
- Then we perform data preprocessing by assigning the respective data types to the variables.
- Since our data has multiple purchases on a single date, we use group by function to group all the dates with its respective average net price value on a single date.

```
df = df.groupby('Date', as_index=False, sort=False)['Netprice'].mean()
print (df)
```

Since our data have many missing dates , the time series forecasting wont be effective. So we add all the missing dates to a given date range

```
df['Date'] = pd.to_datetime(df['Date'].astype(str), format='%d-%m-%Y')
df2=df[df.duplicated('Date')].set_index('Date')
new_df=df.drop_duplicates('Date').set_index('Date').asfreq('D',fill_value=None)
new_df=new_df.append(df2).sort_index().reset_index()
print(new_df)
```

Then imputed the net price value of the recently added missing date with **ffill method** (Forward Fill) which takes the previous value from the given observation and fills through out

```
df3.fillna(method='ffill',inplace=True)
```

- Then we check for stationarity , since we know that for doing a time series forecasting the data should be stationary.
- For this we use Augmented **Dickey-Fuller test** .

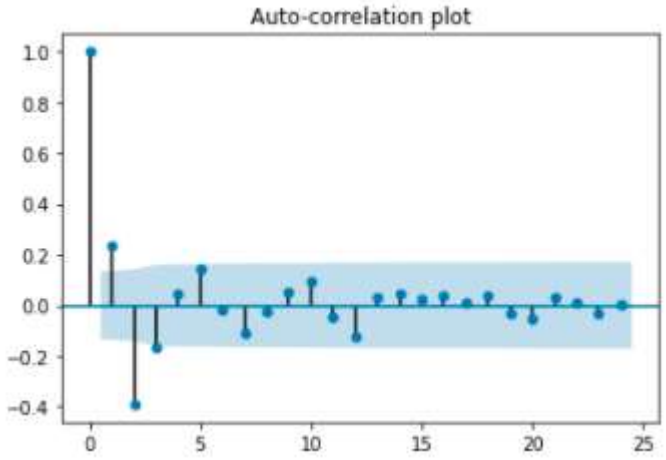
```
from statsmodels.tsa.stattools import adfuller
X = df["Netprice"].values
result = adfuller(X)
print('ADF Statistic: %f' % result[0])
print('p-value: %f' % result[1])
print('Critical Values:')
for key, value in result[4].items():
    print('\t%s: %.3f' % (key, value))

if result[0] < result[4]["5%"]:
    print ("Reject Ho - Time Series is Stationary")
else:
    print ("Failed to Reject Ho - Time Series is Non-Stationary")
```

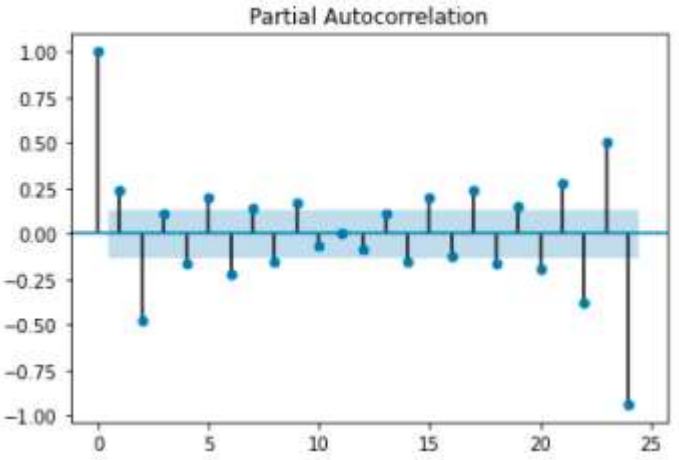
```
/usr/local/lib/python3.7/dist-packages/statsmodels/tools/_testing.py:19:
  import pandas.util.testing as tm
ADF Statistic: -1.156668
p-value: 0.691953
Critical Values:
        1%: -3.466
        5%: -2.877
        10%: -2.575
Failed to Reject Ho - Time Series is Non-Stationary
```

- According to the Augmented **Dickey-Fuller test** , its clear that our data is not stationary. So we perform differencing to make it Stationary

```
df["Netprice_diff"] = df["Netprice"] - df["Netprice"].shift(2)
```

Then for analyzing and forecasting **time series** data we use the statistical models named **ARIMA** (AutoRegressive Integrated Moving Average). To fit the Arima model **ARIMA(p,d,q)**" we need these three terms **p**, d, and q.So for finding those we use Acf and Pacf plot





From these plots it is clear that , p=2 and q=2. Now these parameters can be used inside the Arima model

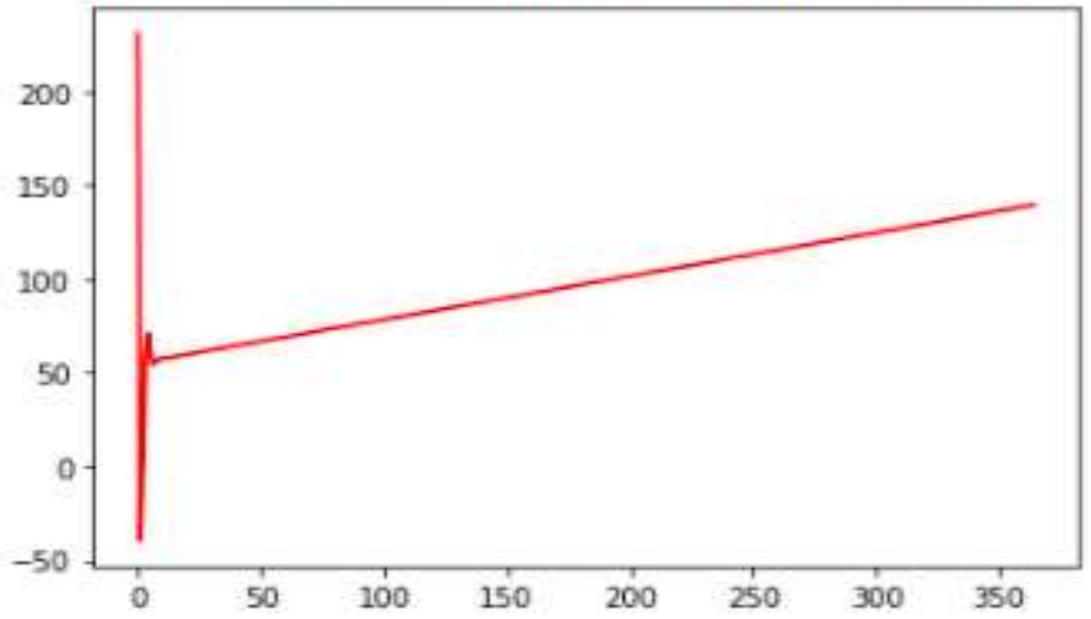
```
from statsmodels.tsa.seasonal import seasonal_decompose
from statsmodels.tsa.arima_model import ARIMA
decomposition = seasonal_decompose(df, freq=100)
model = ARIMA(df, order=(2,0,2))
results = model.fit(disp=-1)
print(results.summary())
```

After fitting the model, then further forecasting for next 365 days

```
predictions= results.forecast(steps=365)[0]
predictions
```

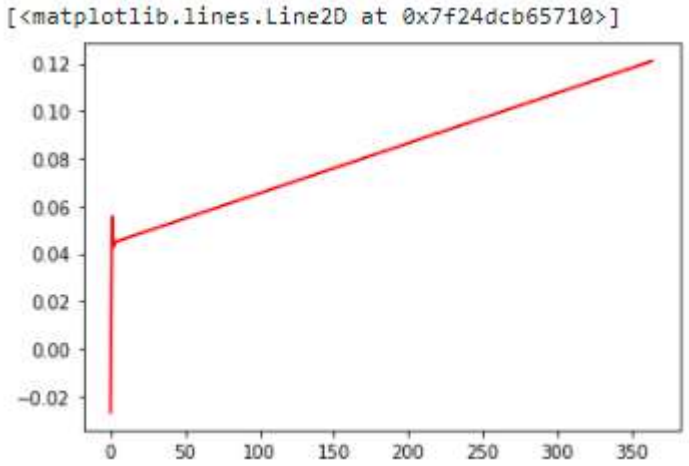
```
plt.plot(predictions,color='red')
```

```
[<matplotlib.lines.Line2D at 0x7f2da62040d0>]
```

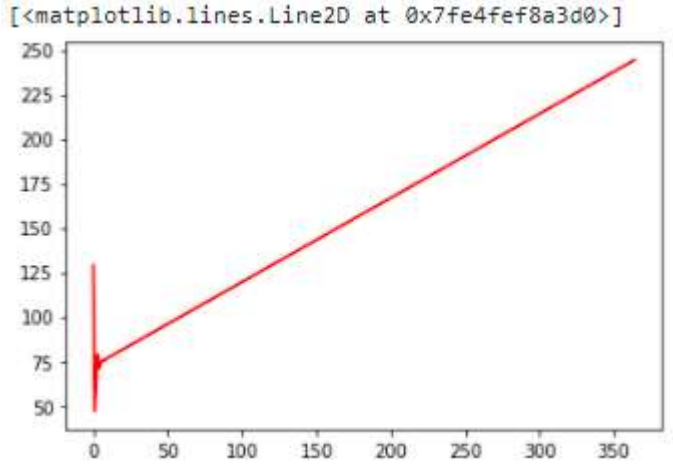


The forecasted values are in increasing trend which clearly shows that, for the material called Soya Bean (A) , the purchase will be in more in next one year.

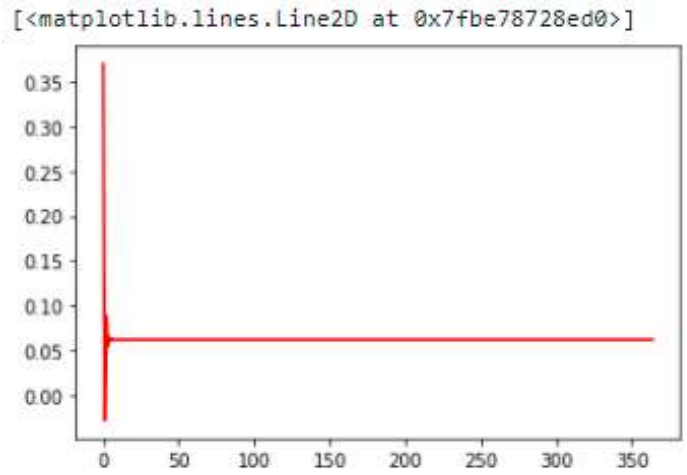
Maize



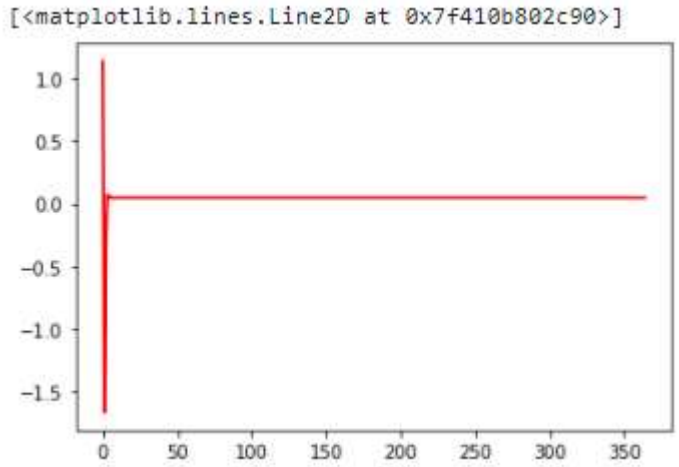
Soya Bean (MP)



IB Ross Broiler Finisher Feed



IB Ross Broiler Starter Feed



Clustering

Two methods were used to cluster items that have similar purchasing patterns – KModes & KMeans.

KModes:

- For that first we will extract the significant columns that are used for clustering. Columns such as short text,cocd,matl group,Po quantity and gross value were taken for further clustering.
- After that we will install K-Prototypes library which will do the clustering process which can handle both the categorical and Numeric variables.
- Performed clustering with the range of 2 to 4 number of clusters and look for the cost function. The cost is somewhat higher.
- Then performed clustering with the range of 4 to 6 number of clusters and now the cost function is lower than the previous one, hence we assume the number of clusters as 5 .

Finding out the best possible number of clusters

```
cost=[]
for i in range(2,4):
    model=KPrototypes(n_jobs=-1,n_clusters=i,verbose=2,random_state=0)
    model.fit_predict(dfMatrix,categorical=[0,1,2])
    cost.append(model.cost_)
```

```
Best run was number 6
```

```
cost
```

```
[3.807667777016448e+16, 2.6789596610757284e+16]
```

```
cost=[]
for i in range(4,6):
    model=KPrototypes(n_jobs=-1,n_clusters=i,verbose=2,random_state=0)
    model.fit_predict(dfMatrix,categorical=[0,1,2])
    cost.append(model.cost_)
```

```
Best run was number 5
```

```
cost
```

```
[2.3076052702807224e+16]
```

- Then Run the K prototype algorithm with the no of clusters as 5 .
- We got the model labels as [0,1,2,3,4], and append those model labels in the data frame as the separate column.
- Cluster of materials with the similar purchase trend was identified as follows:

Clusters and its value counts:

```
df_clus['cluster'].value_counts()
```

```
1    59439
2     8168
0      640
4      211
3       73
```

Materials with similar purchasing patterns

CLUSTER - 0

```
cluster_1['Short Text'].value_counts()
```

```
B4 IB Ross Feed              64
PL-4                         59
PL-3                         58
Feed P1                      52
PL-5                         49
Grand Parents Layer          39
C2                           37
GP-5 Feed                    37
B1 IB Ross Feed              37
B2 IB Ross Feed              36
B3 IB Ross Feed              34
Feed PS                      33
GP-4 Feed                    29
Grand Parents Grower         28
C1                           22
Grand Parents Starter        21
Maize                         4
Soya Crude Oil - Purchase     1
Name: Short Text, dtype: int64
```

CLUSTER - 1

```
cluster_2['Short Text'].value_counts()
```

```
IB Ross Broiler Finisher Feed       13662
IB Ross Broiler Starter Feed        12621
IB Ross Broiler Pre-Starter Feed     9965
Maize                                8265
Soya Bean                            6972
Soya Bean - (MP)                     3888
Soya Bean - (A)                      2608
Khandha                              1346
Soya Crude Oil - Purchase              17
B1 IB Ross Feed                        12
Grand Parents Grower                   12
GP-5 Feed                              10
Feed P1                                 9
B2 IB Ross Feed                         9
Grand Parents Starter                   8
C1                                      7
B3 IB Ross Feed                         6
GP-4 Feed                               5
PL-4                                    4
PL-3                                    4
Grand Parents Layer                     4
PL-5                                    4
C2                                      1
Name: Short Text, dtype: int64
```

CLUSTER - 2

```
cluster_3['Short Text'].value_counts()
```

```
Soya Bean - (MP)                    3040
Soya Bean - (A)                     2193
IB Ross Broiler Finisher Feed       1311
IB Ross Broiler Starter Feed         671
Soya Bean                            369
IB Ross Broiler Pre-Starter Feed     223
Khandha                               75
Maize                                 47
B1 IB Ross Feed                       43
Feed P1                               29
Grand Parents Grower                  25
PL-3                                  19
B2 IB Ross Feed                       17
Grand Parents Layer                   17
C1                                    13
Soya Crude Oil - Purchase             12
GP-5 Feed                             12
PL-5                                  12
Feed PS                               10
B3 IB Ross Feed                        9
PL-4                                   9
GP-4 Feed                              6
C2                                     5
Grand Parents Starter                  1
Name: Short Text, dtype: int64
```

CLUSTER - 3

```
cluster_4['Short Text'].value_counts()
```

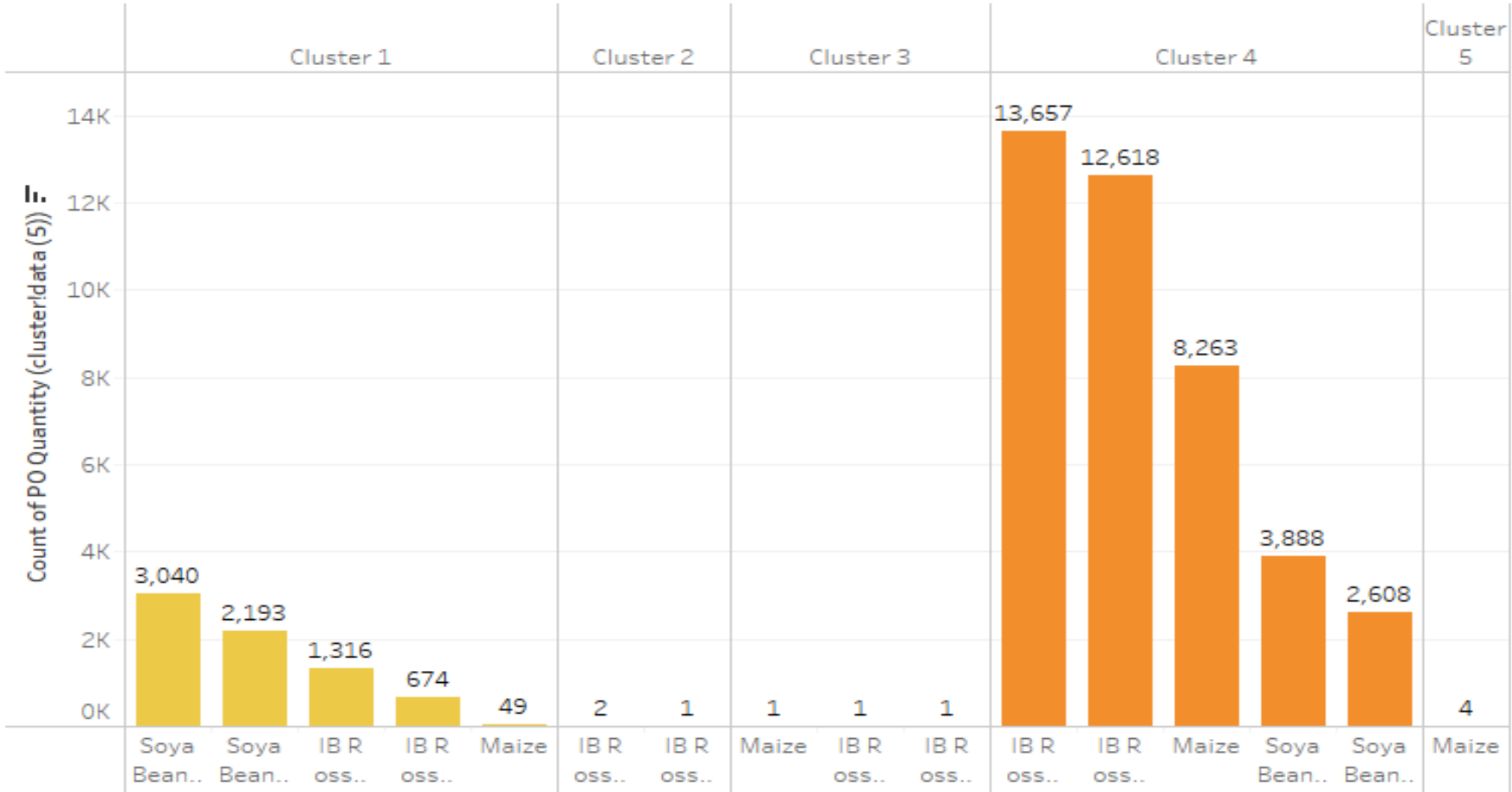
```
B3 IB Ross Feed                 15
B2 IB Ross Feed                 12
Soya Crude Oil - Purchase        9
Feed P1                          9
PL-3                             8
PL-5                             6
Grand Parents Grower             6
PL-4                             4
B1 IB Ross Feed                  1
IB Ross Broiler Starter Feed     1
Maize                            1
IB Ross Broiler Finisher Feed    1
Name: Short Text, dtype: int64
```

CLUSTER - 4

```
cluster_5['Short Text'].value_counts()
```

```
Soya Crude Oil - Purchase          102
B1 IB Ross Feed                     21
B2 IB Ross Feed                     16
B3 IB Ross Feed                     16
PL-5                                15
PL-4                                11
Feed P1                             10
PL-3                                 9
Feed PS                              4
B4 IB Ross Feed                      2
IB Ross Broiler Finisher Feed        2
C1                                   1
IB Ross Broiler Starter Feed         1
IB Ross Broiler Pre-Starter Feed     1
Name: Short Text, dtype: int64
```

Clustering KMode



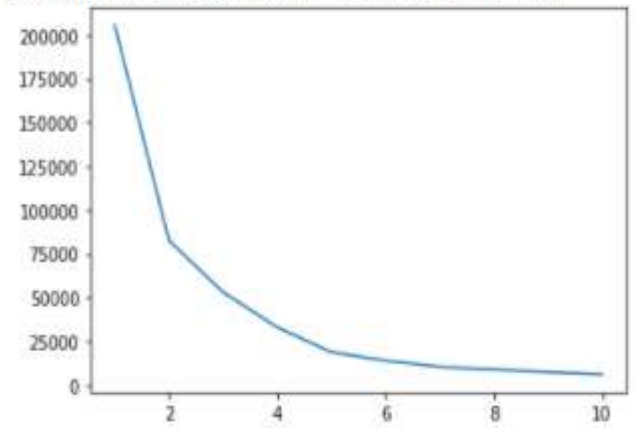
Clustering - KMeans

Clustering was performed on the data extracted with Top 25 Products/ Item that contributes to 80% > the Gross Value.

- There were only 3 numerical data (PO Quantity, Net price & Gross Value) with which KMeans technique was performed.
- PO Quantity & Gross Value was highly skewed (10>)and in order to normalize the skewness top 3 Method (Log, Sqrt & Boxcox) used to identify the best suit, as such Boxcox Method was used to normalize the skewness.
- With the help of KMeans technique 3 cluster were created and the Silhouette Score was 0.6974452046084545.
- Model labels were created for 3 clusters and the same were appended as column to the data frame.
- Cluster 1, found to have more no.of highest count with similar purchase pattern.
- Output was then extracted to CSV file.

```
[ ] plt.plot(range(1,11),WSS)
```

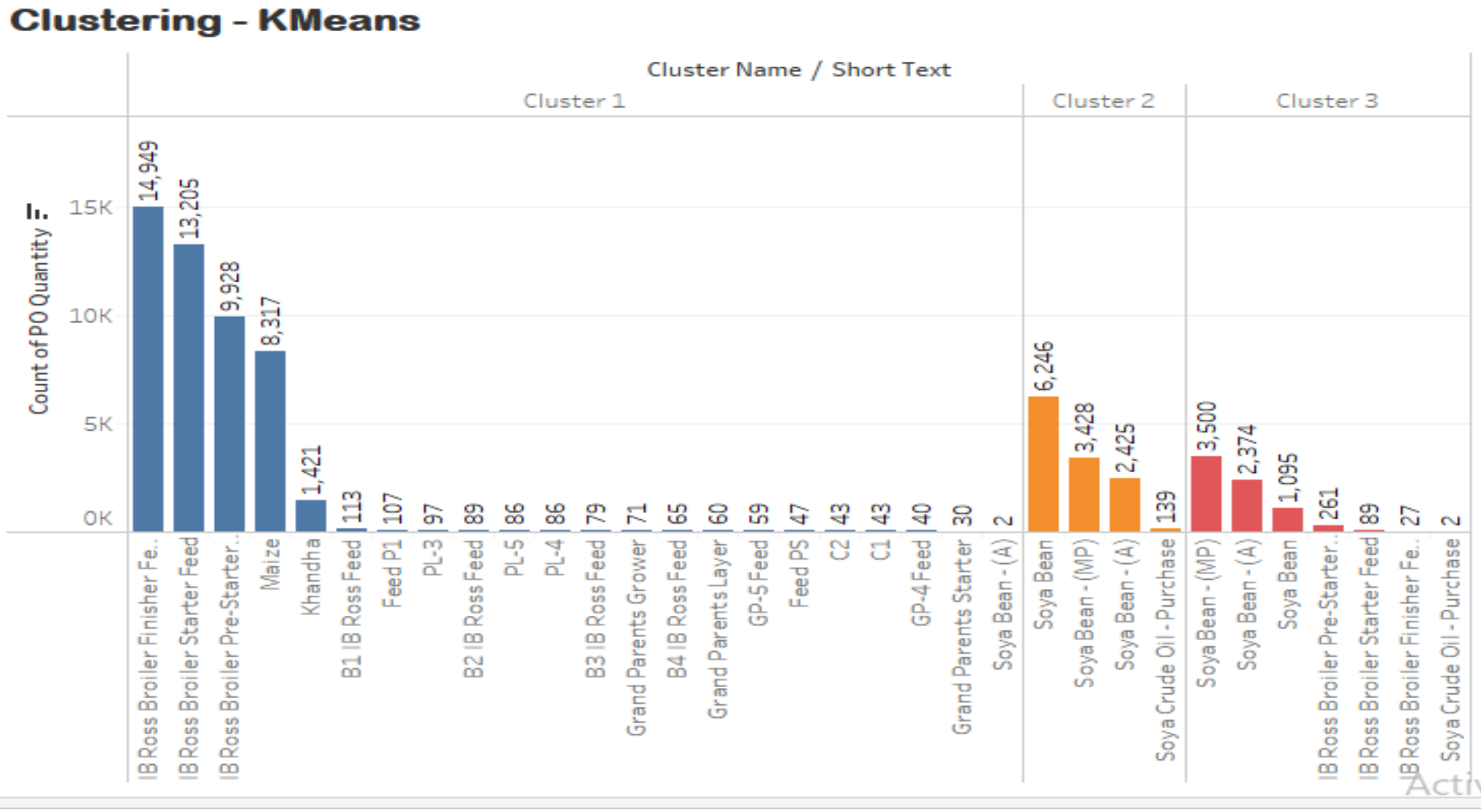
```
[<matplotlib.lines.Line2D at 0x7fec93346f90>]
```




```
data_df['Cluster_Score'].value_counts().sort_index()
```

```
0    48937
1    12238
2     7348
Name: Cluster_Score, dtype: int64
```

Clustering - KMeans



Cost Saving Opportunity

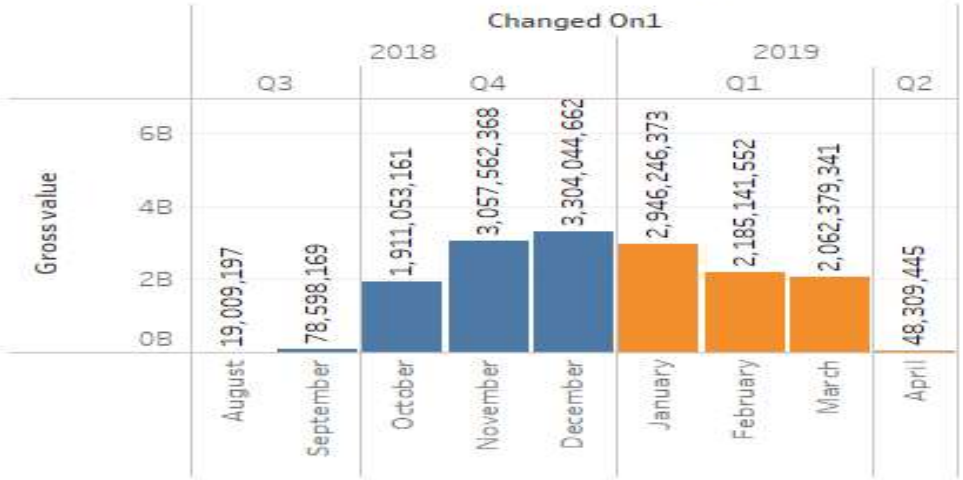
- In order to find the saving opportunity, Minimum value of Net price for a specific month was taken and calculated with Purchase Order Quantity for all 25 product / items. Of which Top 5 Product/ Item has significant saving opportunity table as below. As such placing bulk purchases by Monthly (depending on the requirement) could help to reduce the cost significantly.
- Soya Bean-(A), has 2 transaction with a very low net price value (6413.03) on 27th Jan 2019 however the cost of this product is above 35000 and hence this low value was not considered for evaluation.

Product / Item	Gross Value	Possible Savings	Possible Savings in %
Maize	2,942,718,145.02	511,504,100.40	17.38%
IB Ross Broiler Finisher Feed	4,285,259,626.41	451,160,023.20	10.53%
IB Ross Broiler Starter Feed	3,209,562,870.62	319,258,431.05	9.95%
Soya Bean - (A)	2,227,880,477.51	167,786,465.14	7.53%
Soya Bean - (MP)	2,946,923,148.36	209,936,313.06	7.12%

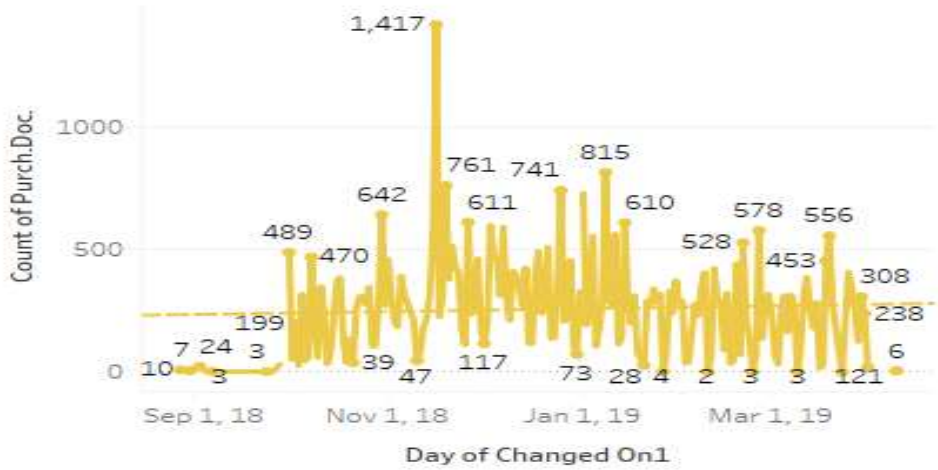
Dashboard for Top 5 materials that contributes more towards Purchase

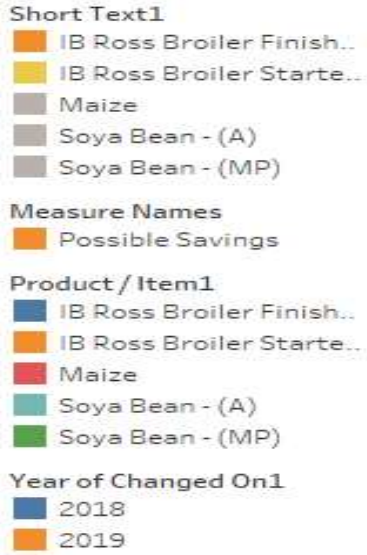
Performance & Cost Saving Opportunity of Top 5 Product / Item

Top 5 Products/ Items -Total Purchases during 2018 & 2019 - Quarterly

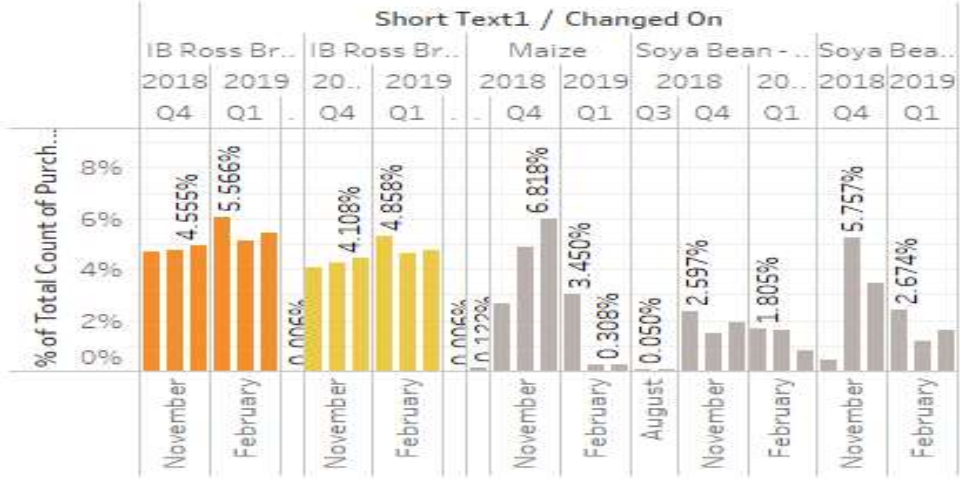


Top 5 Product/ Item - Purchase Trend 2018 -2019

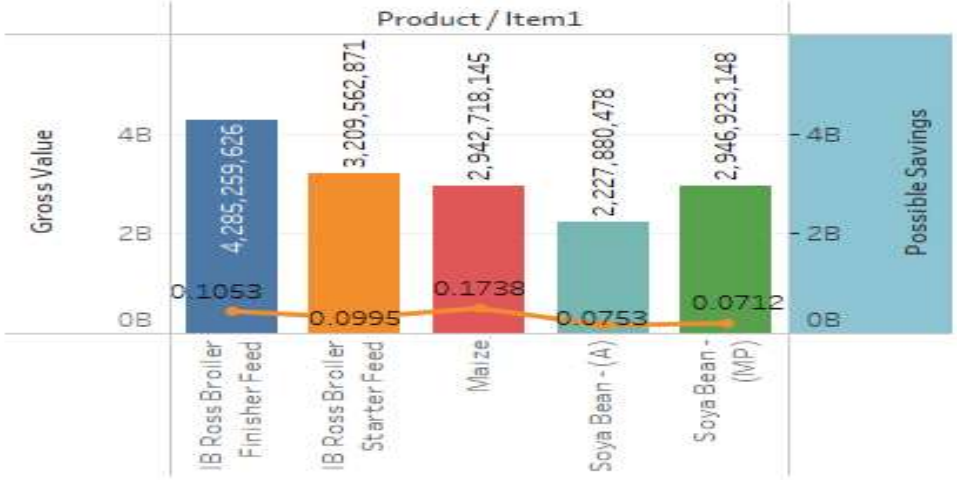




Purchase Pattern for - Top 5 Product/ Item



Possible Cost Saving Opportunities - Based Top5 Product / Item



Recommendations:

- Consolidation of Purchase Order Request and placing in single order to could help reduce the amount of work and cost involved.
- Placing bulk order could help to negotiate with vendor for better discounts.
- Not sure whether the Transportation cost was included in the Gross Value / Net Price per unit. No clear information available with regards to the storage facility or capacity. Further more information required to look for better saving opportunity in terms of storage / transportation when purchased in bulk either Quarterly or Monthly.