Growth of Over the top(OTT) and Factors Affecting It

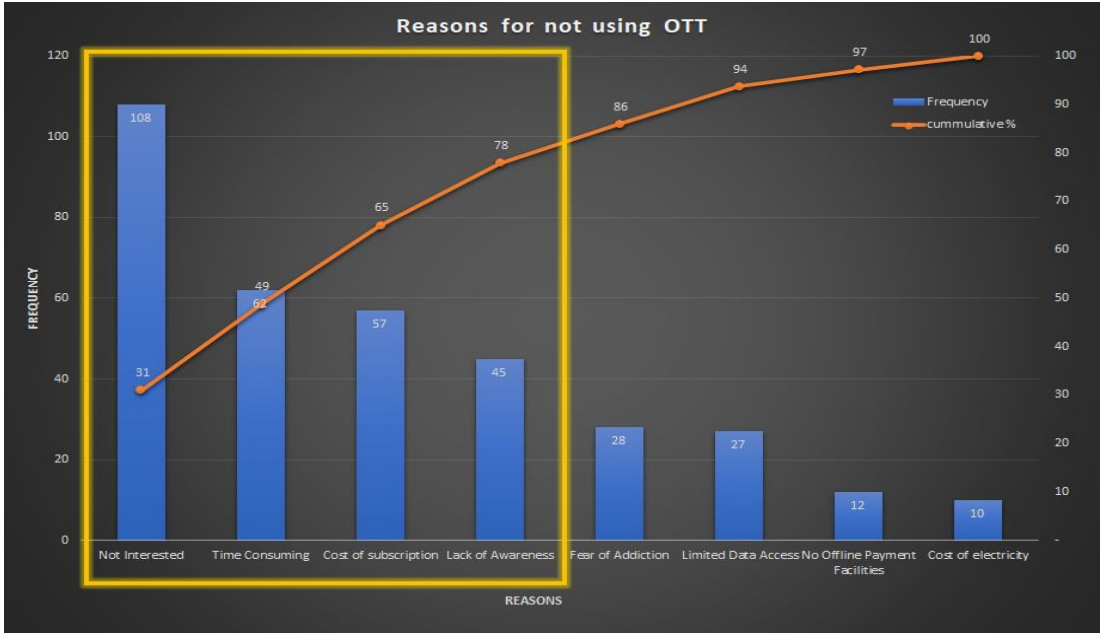
PARETO ANALYSIS:

Pareto analysis is based on the idea that 80% of a project's benefit can be achieved by doing 20% of the work or conversely 80% of problems are traced to 20% of the causes.

Objective: To Identify reasons for not using OTT

The variables used in the analysis are:

Not Interested
Time Consuming
Cost of subscription
Lack of Awareness
Fear of Addiction
Limited Data Access
No Offline Payment Facilities
Cost of electricity



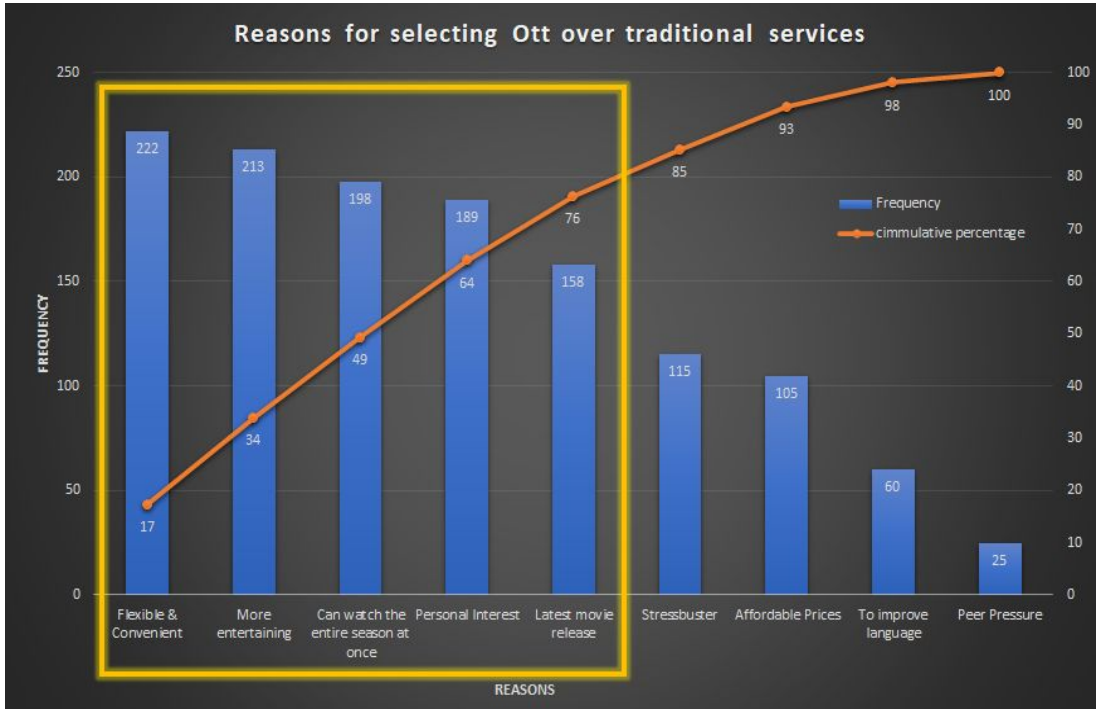
<u>Interpretation</u>:

From Pareto Analysis, it is conclude that the main reasons for not using Ott are ‘Not Interested’, ‘Time consuming’, ‘Cost of subscription’, ‘Lack of Awareness’. So to overcome these reasons providers should focus on awareness of OTT platforms and content also provide different subscription plans.

Objective: Reasons behind switching over to Online Video Streaming(OTT) from Traditional services.

The variables used in the analysis are:

Flexible & Convenient
More entertaining
Can watch the entire season at once
Personal Interest
Latest movie release
Stressbuster
Affordable Prices
To improve language
Peer Pressure



Interpretation:

From Pareto Analysis, it is concluded that the main reasons behind the growth of OTT over traditional services are its flexibility, convenience, more entertaining, the entire season can be watched at once, more interest in OTT content and availability of the latest movies.

K-Means Cluster Analysis

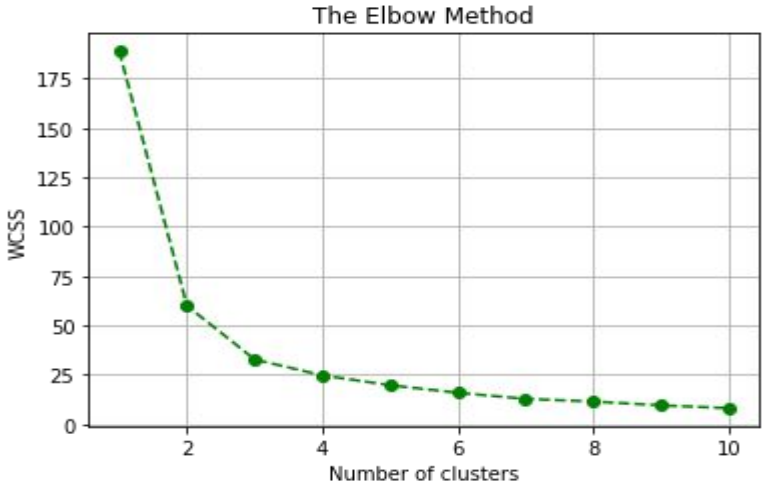
- **Objective**: To compare and classify the viewers based on the user's preference.
- **K-means cluster analysis** is an algorithm that groups similar objects into groups called clusters. The endpoint of cluster analysis is a set of clusters, where each cluster is distinct from each other cluster.
- **Assumptions:** the clusters are spherical and second that the clusters are of resemble in size.

Variables used in analysis are:

1	Gender	15	Yearly spend on OTT
2	Age	16	Regular watching
3	Marital Status	17	Internet Source
4	Education	18	Language
5	Employment	19	Device used for streaming
6	Area	20	Ideal Time for streaming
7	family members	21	OTT platforms in use
8	Income	22	OTT Subscriptions
9	Awarness about OTT	23	Content
10	OTT user	24	Genre
11	Platform for entertainment	25	Duration of OTT use
12	How did you know about OTT	26	Time spent before lockdown
13	OTT over Traditional	27	Time spent after lockdown
14	Subscription BL		

Calculating optimum number of clusters using elbow method:

In this method, the number of clusters are varies within a certain range.For each number, within-cluster sum of square (wcss) value is calculated and stored in a list These value are then plotted against the range of number of clusters used before. The location of bend in the plot indicates the appropriate number of clusters.



From this graph we choose the number of clusters as '3'.

In [15]:
```
from sklearn.cluster import KMeans
wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters = i, init = 'k-means++', random_state = 42)
    kmeans.fit(feature_scaled)
    wcss.append(kmeans.inertia_)
plt.plot(range(1, 11), wcss,'go--', color='green')
plt.title('The Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.grid()
plt.show()
```

Using the optimum no of clusters k-means model is trained :

In [16]:
```
km = KMeans(n_clusters=3)
y_predicted = km.fit_predict(feature_scaled)
y_predicted
```

Calculation of cluster centroids and addition of cluster columns to original data:

The entire data is divided into 3 clusters and the clusters are named as **Minimal viewer, Normal viewer, Extreme viewer**

```
In [24]: kmeans.cluster_centers_

Out[24]: array([[0.79487179, 0.30246914, 0.69230769, ..., 0.48717949, 0.1025641 ,
                 0.25641026],
                [0.375     , 0.20216049, 0.95833333, ..., 0.70833333, 0.75      ,
                 0.84895833],
                [0.43478261, 0.1763285 , 0.86956522, ..., 0.93478261, 0.93478261,
                 0.91576087],
                ...,
                [0.46808511, 0.19858156, 0.87234043, ..., 0.29787234, 0.44680851,
                 0.43085106],
                [0.6097561 , 0.22583559, 0.82926829, ..., 0.12195122, 0.56097561,
                 0.42682927],
                [0.93333333, 0.21440329, 0.84444444, ..., 0.8       , 0.46666667,
                 0.50555556]])

In [25]: #adding cluster column to data
         feature_scaled['cluster']=y_predicted

In [17]: #to check no of observations in each cluster.
         feature_scaled['cluster'].value_counts()

Out[17]: 2    145
         1    141
         0    121
         Name: cluster, dtype: int64
```

The distribution of each cluster is as follows:

Table 1

	Minimal viewer	Normal viewer	Extreme viewer
No_of_Device	2	2	3
No_of_Purchased	1	1	2
No_of_content	4	4	7
Total_Genre	4	5	7
Total_Platform	3	4	5
Total_Subscription	1	2	3
Yearly spend on OTT	514	2210	2758

The table 1 shows that extreme viewers spend more money on the subscriptions of different types of OTT platforms, also they prefer to watch a variety of content, genre and they use more devices.

Duration of OTT use	Minimal viewer	Normal viewer	Extreme viewer
Less than 3 months	67%	21%	13%
3-5 months	59%	23%	18%
5-12 months	42%	33%	24%
2 years	31%	43%	26%
3 years	28%	35%	37%
more than 3 years	5%	39%	56%
Regular watching			
no	50%	29%	20%
yes	16%	44%	40%
Income			
below 1L	53%	15%	33%
1-4L	43%	34%	24%
4-8L	32%	32%	37%
8-12L	33%	28%	39%
above 12L	10%	39%	51%

Table 2

Table 2 shows that minimal viewers have been using OTT platforms for less than three months and Extreme viewers have been using it for the past three years.Number of Minimal viewers are more in the below 1 lakh income category whereas Extreme viewers are more in 8-12 lakhs income category. Also Extreme viewers prefer to watch regularly.

Table 1

	Minimal viewer	Normal viewer	Extreme viewer
No_of_Device	2	2	3
No_of_Purchased	1	1	2
No_of_content	4	4	7
Total_Genre	4	5	7
Total_Platform	3	4	5
Total_Subscription	1	2	3
Yearly spend on OTT	514	2210	2758

The table 1 shows that extreme viewers spend more money on the subscriptions of different types of OTT platforms, also they prefer to watch a variety of content, genre and they use more devices

Duration of OTT use	Minimal viewer	Normal viewer	Extreme viewer
Less than 3 months	67%	21%	13%
3-5 months	59%	23%	18%
5-12 months	42%	33%	24%
2 years	31%	43%	26%
3 years	28%	35%	37%
more than 3 years	5%	39%	56%
Regular watching			
no	50%	29%	20%
yes	16%	44%	40%
Income			
below 1L	53%	15%	33%
1-4L	43%	34%	24%
4-8L	32%	32%	37%
8-12L	33%	28%	39%
above 12L	10%	39%	51%

Table 2

Table 2 shows that minimal viewers have been using OTT platforms for less than three months and Extreme viewers have been using it for the past three years.Number of Minimal viewers are more in the below 1 lakh income category whereas Extreme viewers are more in 8-12 lakhs income category. Also Extreme viewers prefer to watch regularly.

XG Boost Analysis

- **Objective**: To predict which type of subscribers will prefer the leading OTT platform(Amazon Prime).

- **XGBoost** is a scalable ensemble technique based on gradient boosting that has demonstrated to be a reliable and efficient machine learning challenge solver. This work proposes a practical analysis of how this novel technique works in terms of training speed, generalization performance and parameter setup.

- The target variable is stored in 'Y' and features in 'X'.
  Y : User having subscription of Amazon Prime
  Y = 1 ; if yes
  = 0 ; otherwise.

Variables used in analysis are:

1	Gender	15	Yearly spend on OTT
2	Age	16	Regular watching
3	Marital Status	17	Internet Source
4	Education	18	Language
5	Employment	19	Device used for streaming
6	Area	20	Ideal Time for streaming
7	family members	21	OTT platforms in use
8	Income	22	OTT Subscriptions
9	Awarness about OTT	23	Content
10	OTT user	24	Genre
11	Platform for entertainment	25	Duration of OTT use
12	How did you know about OTT	26	Time spent before lockdown
13	OTT over Traditional	27	Time spent after lockdown
14	Subscription BL		

Splitting data into training and testing data sets.

Training the XG Boost model n
train dataset.

```
#training xg-boost model
xg_clf = xgb.XGBClassifier(colsample_bytree=0.6, gamma=1, learning_rate=0.01, max_depth=8,
                           n_estimators=250, objective='binary:logistic',
                           subsample=0.7, reg_alpha=0.1, reg_lambda=2,
                           seed=123, max_delta_step=0, verbosity=3, n_jobs=3,
                           scale_pos_weight=0.5*np.sum(training_df[target_col]==0)/np.sum(training_df[target_col]==1))

    xg_clf.fit(training_df[feature_list], training_df[target_col])

    preds = xg_clf.predict(training_df[feature_list])


# training_df["pred_prob"] = xg_clf.predict_proba(training_df[feature_list])[:,1]
    pred_prob = xg_clf.predict_proba(training_df[feature_list])

    prob_name_list = ["pred_prob_"+str(xg_clf.classes_[0]), "pred_prob_" + str(xg_clf.classes_[1])]

    training_df["pred_prob_"+str(xg_clf.classes_[0])] = pred_prob[:,0]
    training_df["pred_prob_" + str(xg_clf.classes_[1])] = pred_prob[:, 1]

    training_df["y_pred"] = preds

    # training_df.ix[0, "max_prob"] =  training_df["pred_prob"].max()
    # training_df.ix[0, "min_prob"] =  training_df["pred_prob"].min()
    # training_df.ix[0, "avg_prob"] =  training_df["pred_prob"].mean()
    # , "max_prob", "min_prob", "avg_prob"
    training_df[["LeadID", target_col,"y_pred"]+prob_name_list]\
        .round(2).to_csv(r"training_data_pred.csv", index=False)
```

Calculation of Correlation matrix,VIF and Confusion matrix.
Removal of features which were highly correlated and which are having high VIF then again running the model using remaining features.

```
#calculating confusion matrix
def confusion_matrix(df, target_col, top_per=0.0):
    op_df = deepcopy(df)

    if top_per !=0:
        op_df.sort_values(["pred_prob_1"], ascending=False, inplace=True)
        op_df.reset_index(drop=True, inplace=True)
        top_len = int(top_per * op_df.shape[0])
        op_df["y_pred"] = 0
        op_df.loc[0:top_len, "y_pred"]=1

    pivot_df = pd.pivot_table(op_df, index=target_col, columns="y_pred", values="LeadID", aggfunc="count")
    pivot_df["Total"] = pivot_df.sum(axis=1)
    pivot_df.loc["Total", :] = pivot_df.sum(axis=0)
    pivot_df.loc["Precision", :]=[np.nan, np.nan, pivot_df.loc[1, 1]*100/pivot_df.loc["Total", 1]]
    pivot_df.loc["Recall", :]=[np.nan, np.nan, pivot_df.loc[1, 1]*100/pivot_df.loc[1, "Total"]]
    pivot_df.loc["Per_Base_Pred", :]=[np.nan, np.nan, pivot_df.loc["Total", 1]*100/pivot_df.loc["Total", "Total"]]

    pivot_df.reset_index(inplace=True)
    pivot_df["OTT_Subscription_PRIME"] = ["Actual_0", "Actual_1" ,"Total", "Precision", "Recall", "Per_Base_Pred"]
    pivot_df.rename(columns={0:"Pred_0", 1:"Pred_1"}, inplace=True)
    return pivot_df
print("number of features", len(feature_list))
```

```
#calculating correlation matrix

    t_corr_df = training_df[feature_list].corr()
    t_corr_df.round(2).to_csv(r"t_corr.csv")
#calculating VIF
    t_vif = pd.DataFrame()
    t_vif["VIF Factor"] = [variance_inflation_factor(training_df[feature_list].values, i) for i in range(training_df[feature_lis
    t_vif["features"] = feature_list
    t_vif.to_csv(r"training_data_vif.csv", index=False)
```

```
# feature importance df
    feat_imp_df = pd.DataFrame()
    feat_imp_df["Feature_Name"] = feature_list
    feat_imp_df["Imp"] = xg_clf.feature_importances_

    feat_imp_df.sort_values(["Imp"], ascending=False, inplace=True)
    feat_imp_df.reset_index(drop=True, inplace=True)
    # feat_imp_df.round(2).to_csv("D:/Analytics/Wayne/PMS_Redemption/xgboost_model_building/new_y
    #         index=False)

    feat_imp_df.to_csv(r"training_data_feat_imp.csv",
            index=False)
```

Top 15 features are selected using feature important function

Below table is the final correlation matrix table.

	Yearly spend	OTT_Subscript	Subscription	OTT_Subscript	No_of_Devic	Internet_Sou	Genre_Actio	Genre_Sci-I	time spent A	Content_Mo	OTT_Subscrip	OTT_Subscrip
Yearly spend on OTT	1	0.38	0.28	0.25	0.13	0.17	0	0.06	0.18	0.13	0.32	0.09
OTT_Subscription_NETFLIX	0.38	1	0.38	0.26	0.26	0.25	0.04	0.2	0.09	0.11	0.33	0.07
Subscription BL	0.28	0.38	1	0.26	0.24	0.23	0.15	0.19	0.14	0.21	0.23	0.11
OTT_Subscription_HOTSTAR	0.25	0.26	0.26	1	0.23	0.13	0.02	0.02	0.13	0.06	0.39	0.23
No_of_Device	0.13	0.26	0.24	0.23	1	0.28	0.06	0.19	0.11	0.09	0.23	0.09
Internet_Source_Wi-Fi	0.17	0.25	0.23	0.13	0.28	1	-0.04	0.11	0.12	0.05	0.06	-0.03
Genre_Action	0	0.04	0.15	0.02	0.06	-0.04	1	0.16	0.05	0.13	0.07	0.09
Genre_Sci-Fi	0.06	0.2	0.19	0.02	0.19	0.11	0.16	1	0.1	0.16	0.02	0.16
time spent AL	0.18	0.09	0.14	0.13	0.11	0.12	0.05	0.1	1	0.14	0.18	0.12
Content_Movies	0.13	0.11	0.21	0.06	0.09	0.05	0.13	0.16	0.14	1	0.11	0.06
OTT_Subscription_ZEE5	0.32	0.33	0.23	0.39	0.23	0.06	0.07	0.02	0.18	0.11	1	0.24
OTT_Subscription_XSTREAM	0.09	0.07	0.11	0.23	0.09	-0.03	0.09	0.16	0.12	0.06	0.24	1

Table 1

2nd table is indicating final VIF matrix,VIF for all the variables is under 8 hence there is no multicollinearity

VIF Factor	features
1.631915698	Yearly spend on OTT
2.801459977	OTT_Subscription_NETFLIX
4.104388923	Subscription BL
2.208447032	OTT_Subscription_HOTSTAR
6.100228036	No_of_Device
4.405732579	Internet_Source_Wi-Fi
2.538700975	Genre_Action
2.250108685	Genre_Sci-Fi
2.76209053	time spent AL
4.778500154	Content_Movies
1.58695512	OTT_Subscription_ZEE5
1.217599688	OTT_Subscription_XSTREAM

Table 1 is the confusion matrix for training data set:

OTT_Subscription_PRIME	Pred_0	Pred_1	Total
Actual_0	116	15	131
Actual_1	45	108	153
Total	161	123	284
Precision			87.8
Recall			70.59
Accuracy of model			0.788732

Table 1

Table 2 is the confusion matrix for testing data set:

OTT_Subscription_PRIME	Pred_0	Pred_1	Total
Actual_0	51	6	57
Actual_1	24	42	66
Total	75	48	123
Precision			87.5
Recall			63.64
Accuracy of model			0.756098

Table 2

The Accuracy of the training set is 78.87% & the testing set is 75.61% .
Hence the results of the testing set are very close to the results of the training set hence we can say that our model is a good fit and ready for future prediction.

Significant features for this model are given in the table 3 along with their significance.

Feature_Name	Imp
Yearly spend on OTT	0.229117
OTT_Subscription_NETFLIX	0.19697
Subscription BL	0.120353
time spent AL	0.072684
OTT_Subscription_HOTSTAR	0.07069
No_of_Device	0.067124
OTT_Subscription_ZEE5	0.059562
Internet_Source_Wi-Fi	0.055217
Genre_Action	0.045168
Content_Movies	0.042665
Genre_Sci-Fi	0.040451

Table 3