# Build a Decision Tree model for Customer Churn

## Business Objective

A Decision Tree is a Supervised learning technique that can be used for both classification and regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules, and each leaf node represents the outcome.

It is a graphical representation of all possible solutions to a problem/decision based on given conditions. It is called a decision tree because similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure. A decision tree asks a question and based on the answer (Yes/No), it further splits the tree into subtrees.

In our case study, we will be working on a churn dataset. Churned Customers are those who have decided to end their relationship with their existing company.
XYZ is a service-providing company that provides customers with a one-year subscription plan for their product. The company wants to know if the customers will renew the subscription for the coming year or not.

We have already seen how the logistics regression model works on this dataset. It is advised to check this project first, Churn Analysis for Streaming App using Logistic Regression
In this project, we will try to fit the decision tree classifier on the given dataset.

## Data Description

The CSV consists of around 2000 rows and 16 columns

**Features:**

1. Year
2. Customer_id - unique id
3. Phone_no - customer phone no
4. Gender -Male/Female
5. Age – age of the customer
6. No of days subscribed - the number of days since the subscription
7. Multi-screen - does the customer have a single/ multiple screen subscription
8. Mail subscription - customer receive emails or not
9. Weekly mins watched - number of minutes watched weekly
10. Minimum daily mins - minimum minutes watched

11. Maximum daily mins - maximum minutes watched
12. Weekly nights max mins - number of minutes watched at night time
13. Videos watched - total number of videos watched
14. Maximum_days_inactive - days since inactive
15. Customer support calls - number of customer support calls
16. Churn -
    - 1- Yes
    - 0 - No

**Aim**

Build a decision tree model on the given dataset to determine whether the customer will churn or not.

**Tech stack**

- Language - Python
- Libraries - NumPy, pandas, matplotlib, sklearn, pickle, imblearn

**Approach**

1. Importing the required libraries and reading the dataset.
2. Feature Engineering
    - Dropping of unwanted columns
3. Model Building
    - Performing train test split
    - Decision tree Model
4. Model Validation (predictions)
    - Accuracy score
    - Confusion matrix
    - ROC and AUC
    - Recall score
    - Precision score
    - F1-score
5. Feature Importance
    - Create a function to find important features
    - Plot the features

**Modular code overview**

```
input
  |_data_regression.csv

src
  |_Engine.py
  |_ML_Pipeline
            |_evaluate_metrics.py
            |_feature_imp.py
            |_ml_model.py
            |_plot_model.py
            |_utils.py

lib
  |_[DEMO]_Decision_Tree.ipynb

output
  |_Decision_Tree_plot.png
  |_Feature_Importance.png
  |_model.pkl
```

Once you unzip the modular_code.zip file you can find the following folders within it.

1. input
2. src
3. output
4. lib

1. Input folder **-** It contains all the data that we have for analysis. There is one csv file in our case,
    - Data_regression

2. Src folder - This is the most important folder of the project. This folder contains all the modularized code for all the above steps in a modularized manner. This folder consists of:
    - Engine.py
    - ML_Pipeline
      The ML_pipeline is a folder that contains all the functions put into different python files, which are appropriately named. These python functions are then called inside the engine.py file.

3. Output folder **–** The output folder contains the best-fitted models that we trained for this data. These models can be easily loaded and used for future use and the user need not have to train all the models from the beginning.

**Note:** This model is built over a chunk of data. One can obtain the model for the entire data by running engine.py by taking the entire data to train the models.

4. Lib folder **-** This is a reference folder. It contains the original ipython notebook that we saw in the videos. There is also a reference ppt attached.

**Project Takeaways**

1. Introduction to decision trees
2. Understanding the measures of impurity
3. Understanding the working of the decision tree algorithm
4. What is Classification and Regression Trees (CART)
5. What is the C5.0 algorithm and CHAID algorithm
6. Comparing the types of decision trees w.r.t measures of impurity
7. Using python libraries such as matplotlib for data interpretation and advanced visualizations.
8. Data inspection and cleaning
9. Using sklearn library to build the decision tree model
10. Splitting Dataset into Train and Test using sklearn.
11. Making predictions using the trained model.
12. Gaining confidence in the model using metrics such as accuracy score, confusion matrix, recall, precision, and f1 score
13. Handling the unbalanced data using SMOTE method.
14. Performing feature importance.