

# Customer Segmentation

## Introduction :

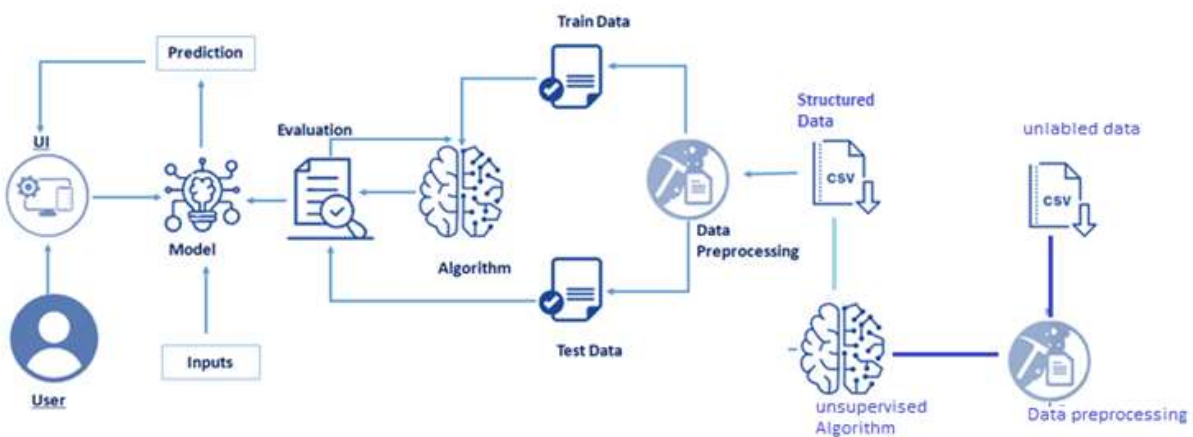
In today's highly competitive world, the primal aim of any business is to grab potential customers who can generate profits for the organization. With increasing the number of organizations in the market, companies want to gain a competitive advantage over others.

The primal task of Management is to identify potential customers from the rest. This will be simplified with the help of Machine Learning models to classify the customers into segments based on various attributes.

The intervention of Data Science and AI helps the business to build such models to analyze the customers and their products in better decision making, to improve the business process, to formulate better strategies, and to improve the revenue.

This project deals with understanding and segmenting the customers based on the data. The Model we built will be able to classify the customer's potentiality in purchasing power. We will be using classification algorithms such as H-clustering, k-means clustering Decision tree, Random forest, KNN, and xgboost. We will train and test the data with these algorithms. From this best model is selected and saved in pkl format. Once the model is saved, we integrate it with the flask application.

## Technical Architecture:



## Pre Requisites

To complete this project, you must require the following software's, concepts, and packages

- o Anaconda navigator:

Refer to the link below to download anaconda navigator

## Prior Knowledge

Customer segmentation use case

<https://www.youtube.com/embed/zPjtDohab-g>

Customer segmentation model using machine learning

[https://www.youtube.com/embed/Liff\\_GA74EI](https://www.youtube.com/embed/Liff_GA74EI)

Open anaconda prompt as administrator.

- Type “**pip install numpy**” and click enter.
- Type “**pip install pandas**” and click enter.
- Type “**pip install matplotlib**” and click enter.
- Type “**pip install scikit-learn**” and click enter.
- Type “**pip install Flask**” and click enter.

The above steps allow you to install the packages in the anaconda environment

## Project Objectives

By the end of this project:

1. This project enables the learner to understand the business use case of how and why to segment the customers.
2. You'll able to understand the unsupervised learning methods such as H-clustering and k-means clustering

3. You'll be able to understand the problem to classify if it is a regression or a classification kind of problem.
4. You will be able to know how to pre-process/clean the data using different data pre-processing techniques.

## Project Flow

User interacts with the UI (User Interface) to enter the input values

- Entered input values are analyzed by the model which is integrated
- Once the model analyses the input, the prediction is showcased on the UI

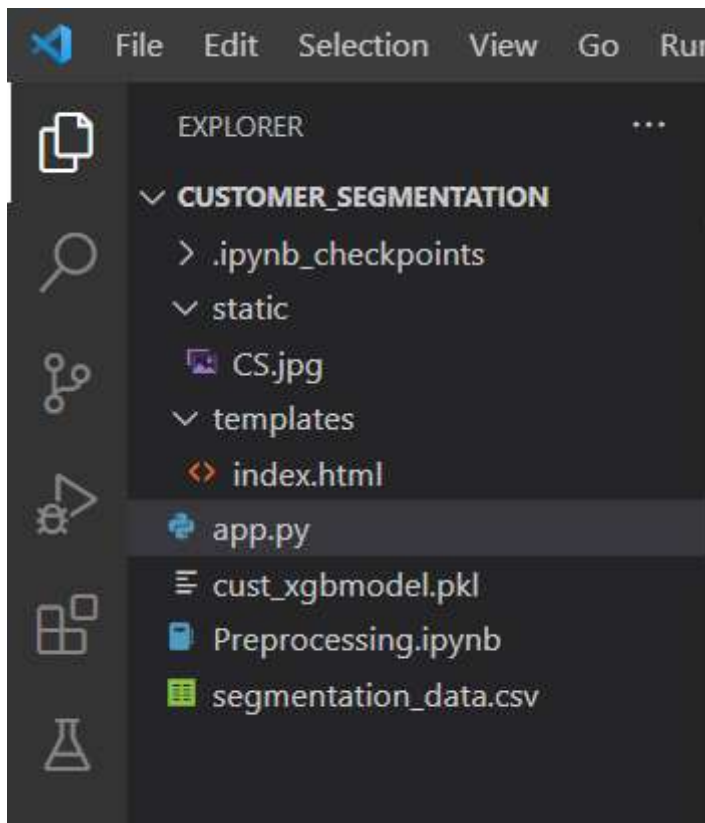
To accomplish this, we have to complete all the activities and tasks listed below

- Data Collection.
  - o Collect the dataset or Create the dataset
- Data Pre-processing.
  - o Import the Libraries.
  - o Importing the dataset.
  - o Checking for Null Values.
  - o Data Visualization.
  - o Taking care of Missing Data.
  - o Feature Scaling.
- Unsupervised Model Building
  - o Import the model building Libraries
  - o Initializing the model
  - o Fit and predict the clusters
  - o Add the classes to the main data set and save the dataset
  - o Splitting x and y
  - o Splitting train and test data
- Supervised Model Building
  - o Import the model building Libraries

- o Initializing the model
- o Model Training
- o Evaluating the Model
- o Save the Model
- Application Building
  - o Create an HTML file
  - o Build a Python Code

## Project Structure

Create a Project folder that contains files as shown below



## Data Collection

ML depends heavily on data, without data, it is impossible for an “AI” to learn. It is the most crucial aspect that makes algorithm training possible. In Machine Learning projects, we need a training

data set. It is the actual data set used to train the model for performing various actions.

[https://docs.google.com/spreadsheets/d/1NnUMX3sjJgRRerkJTAXemIfdyo2GiUhgE\\_m4w-fAhvs/edit#gid=1219451115](https://docs.google.com/spreadsheets/d/1NnUMX3sjJgRRerkJTAXemIfdyo2GiUhgE_m4w-fAhvs/edit#gid=1219451115)

## Data Pre-Processing

Data Pre-processing includes the following main tasks

- o Import the Libraries.

It is important to import all the necessary libraries such as pandas, NumPy, matplotlib.

- **Numpy**- It is an open-source numerical Python library. It contains a multi-dimensional array and matrix data structures. It can be used to perform mathematical operations on arrays such as trigonometric, statistical, and algebraic routines.
- **Pandas**- It is a fast, powerful, flexible, and easy-to-use open-source data analysis and manipulation tool, built on top of the Python programming language.
- **Seaborn**- Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.
- **Matplotlib**- Visualisation with python. It is a comprehensive library for creating static, animated, and interactive visualizations in Python
- **Sklearn** – which contains all the modules required for model building
- **Scipy** – which contains all the modules required for scientific and computing functions.

```
In [1]: import pandas as pd
import seaborn as sns
import matplotlib
import scipy
import sklearn
import os
```

- o Importing the dataset.

- You might have your data in .csv files, .excel files
- Let's load a .csv data file into pandas using read\_csv() function. We will need to locate the directory of the CSV file at first (it's more efficient to keep the dataset in the same directory as your program).

If your dataset is in some other location, then `df= pd.read_csv(r"File_location/datasetname..csv`  
Note: `r` stands for "raw and will cause backslashes rather than special characters.

If the dataset is in the same directory of your program, you can directly read it, without giving raw as `r`.

Our dataset `segmentation_data.csv` contains the following columns.

ID - Unique id of the customer

Sex - Gender of the customer

Marital status - whether the person is married or not

Age - Age of the person

Education - Education of the person

Income - income of the person

Occupation - indicates the profession of a person, employed or unemployed or business

Settlement size - Represents the no. of persons in a family

`Out[3]:`

	Sex	MaritalStatus	Age	Education	Income	Occupation	SettlementSize
ID							
100000001	0	0	67	2	124670	1	2
100000002	1	1	22	1	150773	1	2
100000003	0	0	49	1	89210	0	0
100000004	0	0	45	1	171565	1	1
100000005	0	0	53	1	149031	1	1
100000006	0	0	35	1	144848	0	0
100000007	0	0	53	1	156495	1	1
100000008	0	0	35	1	193621	2	1
100000009	0	1	61	2	151591	0	0
100000010	0	1	28	1	174646	2	0

- o Checking for Null Values.

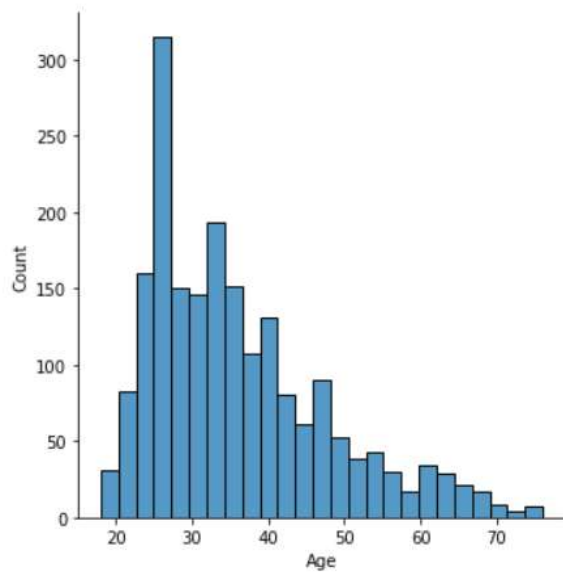
```
In [8]: df.isnull().sum()
```

```
Out[8]: Sex                0  
MaritalStatus            0  
Age                      0  
Education                0  
Income                   0  
Occupation               0  
SettlementSize          0  
dtype: int64
```

- o Data Visualization.

```
In [14]: sns.displot(df['Age'])
```

```
Out[14]: <seaborn.axisgrid.FacetGrid at 0x284afd34400>
```



o Feature Scaling.

```
In [26]: from sklearn import preprocessing  
df = preprocessing.minmax_scale(df, feature_range = (0,1))
```

```
In [27]: df = pd.DataFrame(df, columns = names)
```

```
In [28]: df.head()
```

```
Out[28]:
```

	Sex	MaritalStatus	Age	Education	Income	Occupation	SettlementSize
0	0.0	0.0	0.844828	0.666667	0.324781	0.5	1.0
1	1.0	1.0	0.068966	0.333333	0.420210	0.5	1.0
2	0.0	0.0	0.534483	0.333333	0.195144	0.0	0.0
3	0.0	0.0	0.465517	0.333333	0.496223	0.5	0.5
4	0.0	0.0	0.603448	0.333333	0.413842	0.5	0.5

## Unsupervised Model Building

o Import the model building Libraries

o Initializing the model

- From sklearn.clusters import Kmeans
- from scipy import spatial

For selecting no of clusters it is essential to plot an elbow curve, from that we can identify how many no. of clusters can be taken

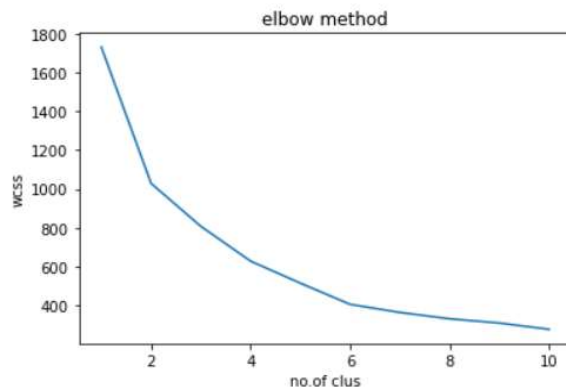


```
In [29]: from sklearn.cluster import KMeans
from scipy import spatial
```

```
In [30]: wssc = []
for i in range(1,11):
    kmeans = KMeans(n_clusters=i, init='k-means++', random_state=0)
    kmeans.fit(df)
    wssc.append(kmeans.inertia_)
```

```
In [31]: import matplotlib as plt
```

```
In [32]: plt.pyplot.plot(range(1,11),wssc)
plt.pyplot.title('elbow method')
plt.pyplot.xlabel('no.of clus')
plt.pyplot.ylabel('wcss')
plt.pyplot.show()
```



```
In [33]: km_model =KMeans(n_clusters = 3, init = 'k-means++', random_state = 0)
```

o Fit and predict the clusters

```
In [36]: ykmeans=km_model.fit_predict(df)
```

o Add the classes to the main data set and save the dataset

o Splitting x and y

In machine learning, the concept of the dependent variable (y) and independent variables(x) is important to understand. Here, the Dependent variable is nothing but output in the dataset and the independent variable is all inputs in the dataset.

· With this in mind, we need to split our dataset into the matrix of independent variables and the vector or dependent variable. Mathematically, Vector is defined as a matrix that has just one column.

To read the columns, we will use iloc of pandas (used to fix the indexes for selection) which takes two parameters — [row selection, column selection].

Let's split our dataset into independent and dependent variables.

```
In [40]: y = df['kclus']  
x = df.drop(columns = ['kclus'], axis = 1)
```

o Splitting train and test data

```
In [42]: from sklearn import model_selection  
x_train, x_test, y_train, y_test = model_selection.train_test_split(x, y, test_size = 0.2, random_state = 0)
```

## Supervised Model Building:

Model building includes the following main tasks

o Import the model building Libraries

```
In [45]: from sklearn.ensemble import RandomForestClassifier  
from sklearn import tree  
import xgboost
```

o Initializing the model

```
In [46]: rand_model = RandomForestClassifier()  
tree_model = tree.DecisionTreeClassifier()  
xgb_model = xgboost.XGBClassifier()
```

o Training and testing the model

```
In [47]: rand_model.fit(x_train,y_train)  
tree_model.fit(x_train,y_train)  
xgb_model.fit(x_train,y_train)  
  
Out[47]: XGBClassifier(base_score=0.5, booster='gbtree', callbacks=None,  
                      colsample_bylevel=1, colsample_bynode=1, colsample_bytree=1,  
                      early_stopping_rounds=None, enable_categorical=False,  
                      eval_metric=None, gamma=0, gpu_id=-1, grow_policy='depthwise',  
                      importance_type=None, interaction_constraints='',  
                      learning_rate=0.300000012, max_bin=256, max_cat_to_onehot=4,  
                      max_delta_step=0, max_depth=6, max_leaves=0, min_child_weight=1,  
                      missing=nan, monotone_constraints=(), n_estimators=100,  
                      n_jobs=0, num_parallel_tree=1, objective='multi:softprob',  
                      predictor='auto', random_state=0, reg_alpha=0, ...)
```

o Evaluation of Model

```
In [48]: pred=rand_model.predict(x_train)
pred1=tree_model.predict(x_train)
pred2=xgb_model.predict(x_train)
```

Accuracy testing using the train data

```
In [50]: print(metrics.accuracy_score(pred,y_train))
print(metrics.accuracy_score(pred1,y_train))
print(metrics.accuracy_score(pred2,y_train))
```

```
1.0
1.0
1.0
```

```
In [51]: pred=rand_model.predict(x_test)
pred1=tree_model.predict(x_test)
pred2=xgb_model.predict(x_test)
```

```
In [51]: pred=rand_model.predict(x_test)
pred1=tree_model.predict(x_test)
pred2=xgb_model.predict(x_test)
```

```
In [52]: print(metrics.accuracy_score(pred,y_test))
print(metrics.accuracy_score(pred1,y_test))
print(metrics.accuracy_score(pred2,y_test))
```

```
0.9975
0.9975
0.9975
```

- o Save the Model

After building the model we have to save the model.

Pickle in Python is primarily used in serializing and deserializing a Python object structure. In other words, it's the process of converting a Python object into a byte stream to store it in a file/database, maintain program state across sessions or transport data over the network. wb indicates write method and rd indicates read method.

- This is done by the below code

```
In [53]: import pickle
```

```
In [54]: pickle.dump(xgb_model,open("cust_xgbmodel.pkl",'wb'))
```

## Application Building

In this section, we will be building a web application that is integrated to the model we built. A UI is provided for the users where he has to enter the values for predictions. The entered values are given to the saved model and prediction is showcased on the UI.

This section has the following tasks

### 1. Building HTML Pages

- In this HTML page, we will create the front end part of the web page. In this page we will accept input from the user and Predict the values.

For more information regarding HTML link

- In our project we have HTML files, they are

#### 1. index.html



## 2. Building server-side script

app.py

```

File Edit Selection View Go Run Terminal Help
app.py - Customer Segmentation - Visual Studio Code


EXPLORER
CUSTOMER_SEGMENTATION
  .ipynb_checkpoints
  static
  templates
    index.html
  app.py
  cust_xgbmodel.pkl
  Preprocessing.ipynb
  segmentation_data.csv

app.py
1 import numpy as np
2 import pickle
3 import pandas
4 import os
5 from flask import Flask, request, jsonify, render_template
6 import joblib
7
8 app = Flask(__name__)
9 model = pickle.load(open('c:/Users/hp/Desktop/Customer_Segmentation/cust_xgbmodel.pkl', 'rb'))
10
11 @app.route('/')
12 def home():
13     return render_template("index.html")
14
15 @app.route('/predict', methods = ["POST", "GET"])
16 def predict():
17     input_feature = [float(x) for x in request.form.values()]
18     features_values = [np.array(input_feature)]
19
20     names = ['Sex', 'MaritalStatus', 'Age', 'Education', 'Income', 'Occupation', 'SettlementSize']
21
22     data = pandas.DataFrame(features_values, columns = names)
23
24     prediction = model.predict(data)
25     print(prediction)
26
27     if(prediction == 0):
28         return render_template("index.html", prediction_text = "Not a potential customer")
29     elif(prediction == 1):
30         return render_template("index.html", prediction_text = "Potential customer")
31     else:
32         return render_template("index.html", prediction_text = "Highly Potential customer")
33
34 if __name__ == "__main__":
35     port = int(os.environ.get('PORT', 5000))
36     app.run(port = port, debug = True, use_reloader = False)

```

```
(deployment) C:\Users\hp\Desktop\Customer_Segmentation>python app.py
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
127.0.0.1 - - [24/Sep/2022 17:51:11] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [24/Sep/2022 17:51:14] "GET /static/CS.jpg HTTP/1.1" 304 -
```

Customer Segmentation with Machine Learning



Customer Details

Sex:

Marital status:

Age:

Education:

Income:

Occupation:

Settlement size:

Potentiality

Kushal Yadav

Customer Segmentation with Machine Learning



Customer Details

Sex:

Marital status:

Age:

Education:

Income:

Occupation:

Settlement size:

Potentiality

Potential customer

Kushal Yadav

