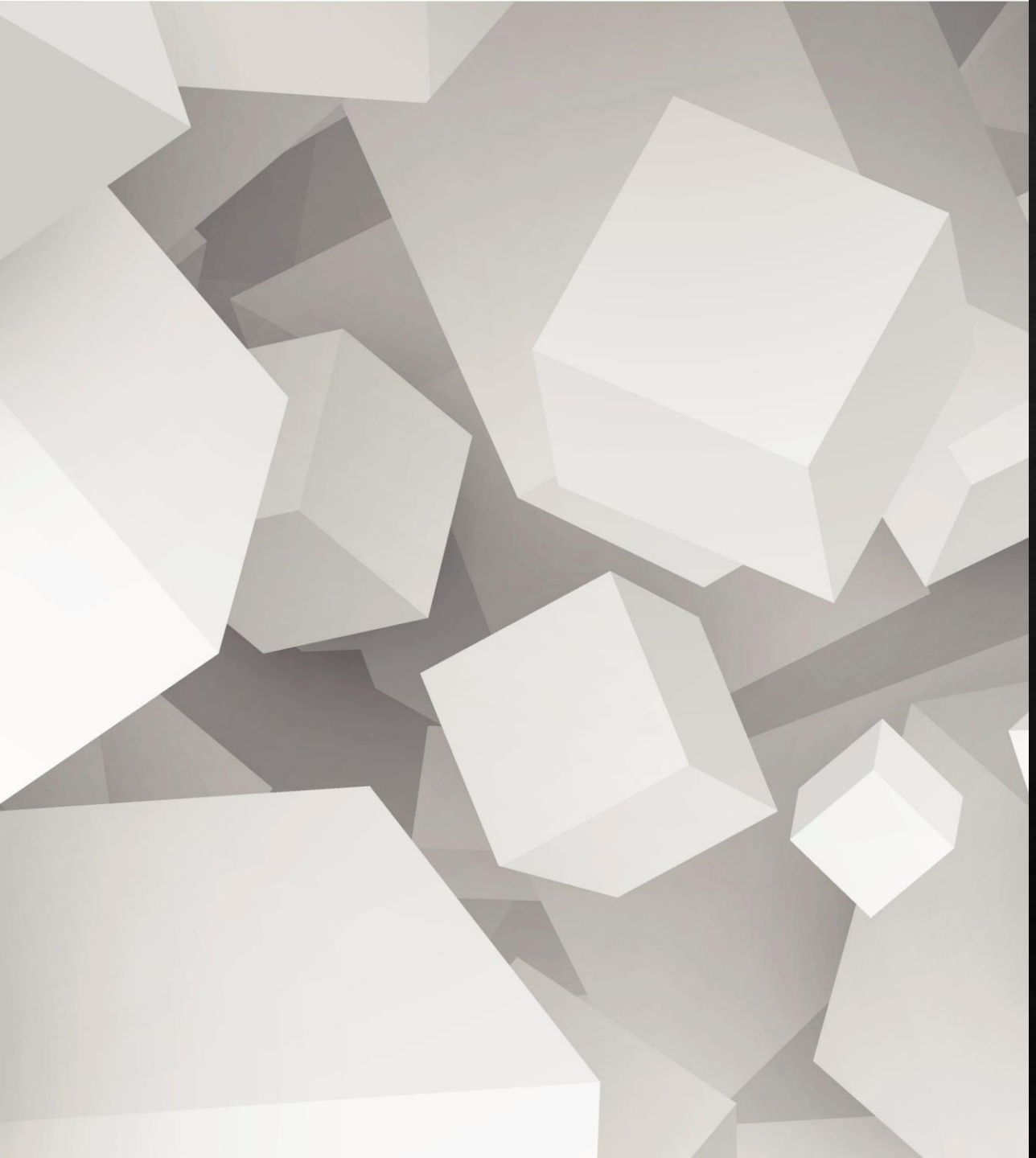


Customer Segmentation

Supervised Machine Learning

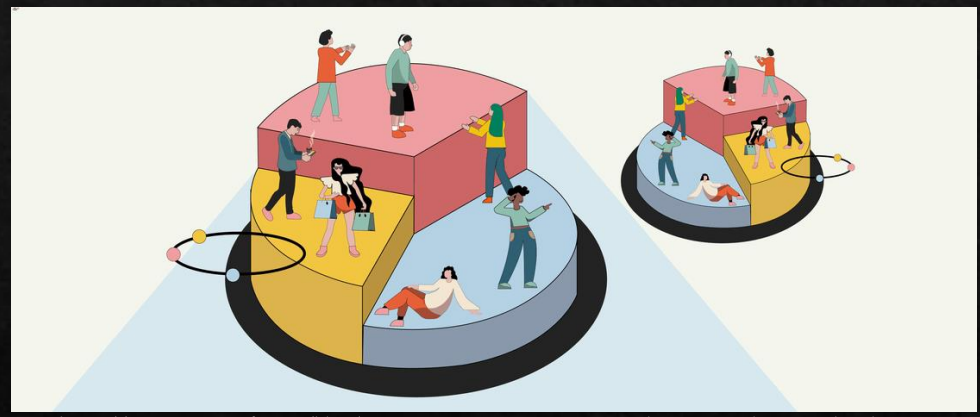
By: Mohsin Mohammed

What is customer segmentation?

Grouping customers into segments that share common characteristics

Customer Segment examples can be based on:

- Age
- Gender
- Profession
- Family size



https://www.segmentify.com/blog/top-customer-segmentation-examples-every-marketer-needs-to-know

Benefits of Customer Segmentation

- Customer satisfaction
- Customer retention
- Efficient use of Resources
- Increased Revenue

Research Problem :

- Help a company classify new customers into four groups .
- Use the existing labeled customer base with unique IDs to train a machine learning algorithm
- Predict on unlabeled instances

Performance Metric :

- Accuracy score

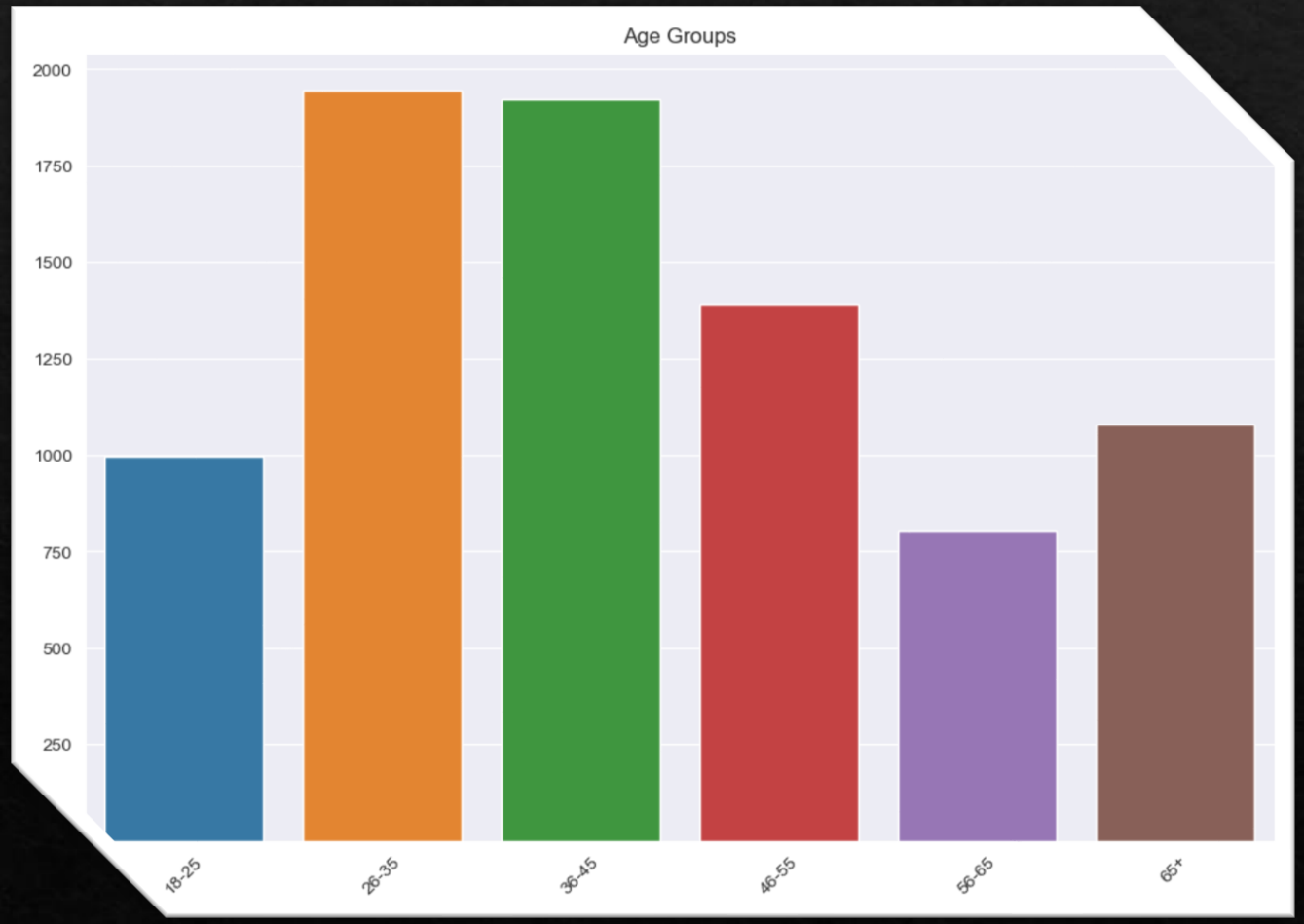
Solution :

- Use supervised machine learning models

Exploratory Data Analysis

- Majority of the customers between 26 – 45 years old

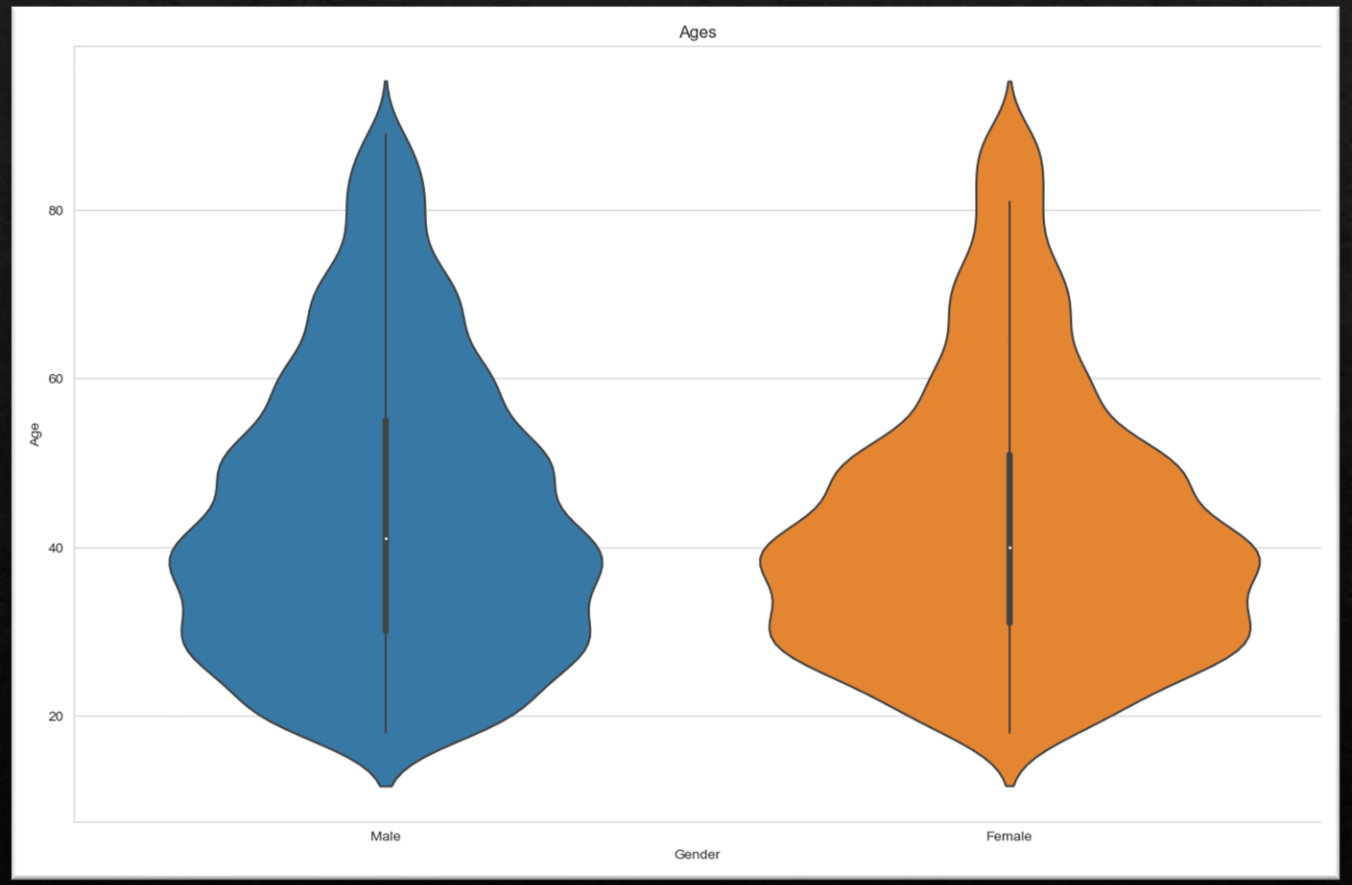
```
def plot_ages(data):

    age_1 = data[(data['Age'] >= 18) & (data['Age'] <=25)]
    age_2 = data[(data['Age'] >= 26) & (data['Age'] <=35)]
    age_3 = data[(data['Age'] >= 36) & (data['Age'] <=45)]
    age_4 = data[(data['Age'] >= 46) & (data['Age'] <=55)]
    age_5 = data[(data['Age'] >= 56) & (data['Age'] <=65)]
    age_6 = data[(data['Age'] >= 65)]


    x = ["18-25", "26-35", "36-45", "46-55", "56-65", "65+"]
    y = [len(age_1), len(age_2), len(age_3), len(age_4), len(age_5), len(age_6)]


    sns.set_style("darkgrid")
    plt.figure(figsize=(12,8))
    plt.title("Age Groups")
    sns.barplot(x=x, y=y)
    plt.xticks(rotation=45)
    plt.savefig('Age_distribution.png', bbox_inches='tight')
    plt.show()

plot_ages(train_data)
```



Exploratory Data Analysis

- Current customers included slightly more female than males

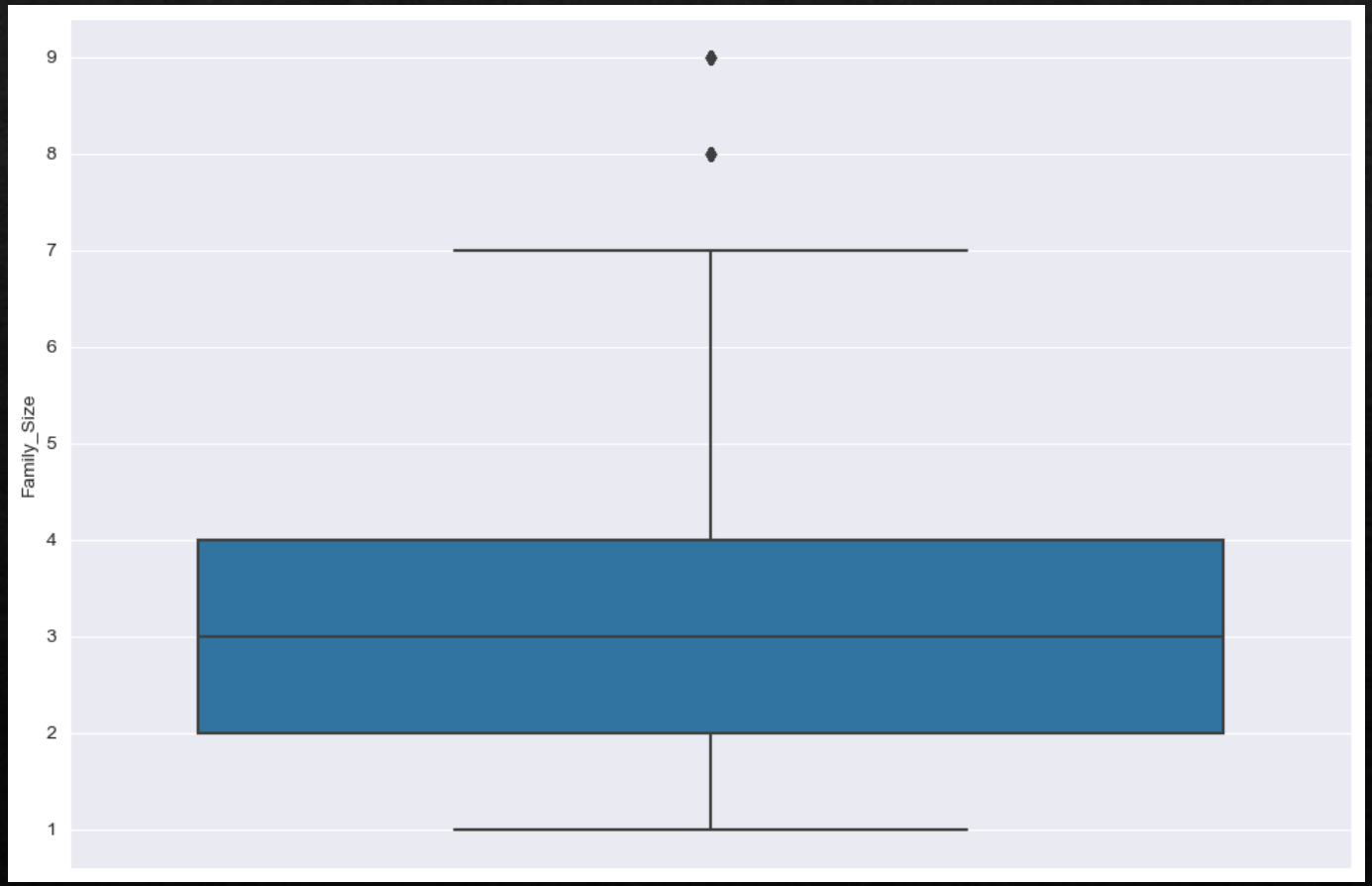
```
sns.set_style('whitegrid')
plt.figure(figsize=(16,10))
plt.title("Ages")
sns.violinplot(x=train_data.Gender, y=train_data.Age)
plt.savefig('violin.png', bbox_inches='tight')
plt.show()
```



Exploratory Data Analysis

- 75 % of the customers have 4 people per household
- 25 % of the customers have 2 people per household

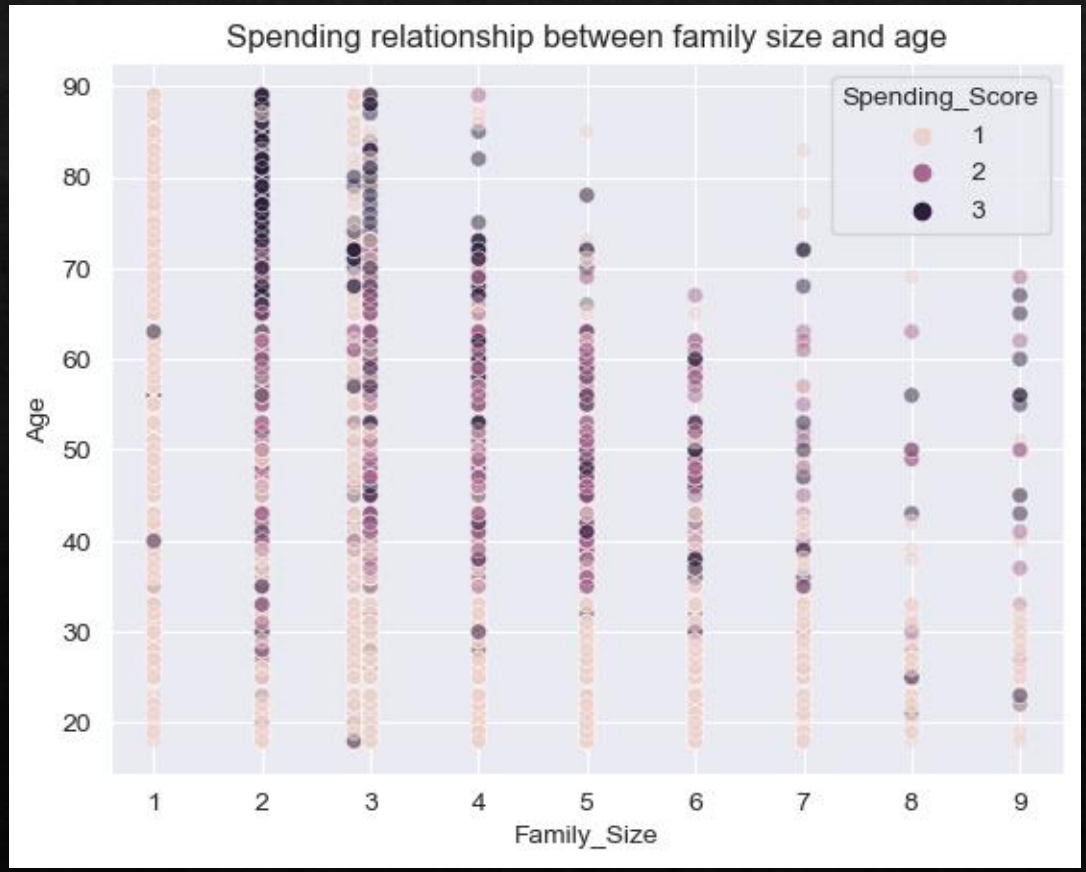
```
plt.figure(figsize=(12,8))
sns.boxplot(y=train_data["Family_Size"])
plt.savefig('family_size.png', bbox_inches='tight')
plt.show()
```



Exploratory Data Analysis

- Customers that have a family size of 2 and between age 65-89 are big spenders

```
sns.scatterplot(x=data_encoded['Family_Size'], y=data_encoded['Age'], alpha=0.5, hue=data_encoded.Spending_Score)
plt.title("Spending relationship between family size and age")
plt.savefig("age_fam_spending_score.png", bbox_inches='tight')
plt.show()
```



Data Preprocessing

- Analyze the train set

```
train_data.head()
```

	ID	Gender	Ever_Married	Age	Graduated	Profession	Work_Experience	Spending_Score	Family_Size	Var_1	Segmentation
0	462809	Male	No	22	No	Healthcare	1.0	Low	4.0	Cat_4	D
1	462643	Female	Yes	38	Yes	Engineer	NaN	Average	3.0	Cat_4	A
2	466315	Female	Yes	67	Yes	Engineer	1.0	Low	1.0	Cat_6	B
3	461735	Male	Yes	67	Yes	Lawyer	0.0	High	2.0	Cat_6	B
4	462669	Female	Yes	40	Yes	Entertainment	NaN	High	6.0	Cat_6	A

- Statistical Analysis

	ID	Age	Work_Experience	Family_Size
count	8068.000000	8068.000000	7239.000000	7733.000000
mean	463479.214551	43.466906	2.641663	2.850123
std	2595.381232	16.711696	3.406763	1.531413
min	458982.000000	18.000000	0.000000	1.000000
25%	461240.750000	30.000000	0.000000	2.000000
50%	463472.500000	40.000000	1.000000	3.000000
75%	465744.250000	53.000000	4.000000	4.000000
max	467974.000000	89.000000	14.000000	9.000000

Data Preprocessing

- Impute missing values

```
def replace_num_missing_values(dataframe):
    numerical_data_cols = dataframe.select_dtypes(include=[np.number]).columns
    print(f"Replacing the missing values from these numerical column attributes: {numerical_data_cols.tolist()}")
    for col in numerical_data_cols:
        dataframe[col].fillna(dataframe[col].mean(), inplace=True)
    return dataframe
```

```
train_data.isnull().sum()   # Checking the missing values
```

```
ID                 0
Gender             0
Ever_Married       0
Age                0
Graduated          0
Profession         0
Work_Experience    0
Spending_Score     0
Family_Size        0
Var_1              0
Segmentation       0
dtype: int64
```

Data Preprocessing

- Merged the train and test set for analyzing data leak
- 2332 Customer IDs were found in the test set
- Delete the duplicates

Checking data leak based on unique customer id

```
In [29]: combined_data.ID.duplicated().sum()
Out[29]: 2332
```

Data Preprocessing

- New shape of the test set dropped to 295 from 2332
- Training a supervised learning model on labeled 8068 instances
- Predicting on 295 unlabeled unique ID instances

```
In [43]: test_data_v1.shape
Out[43]: (295, 17)

In [44]: new_train.shape
Out[44]: (8068, 18)
```

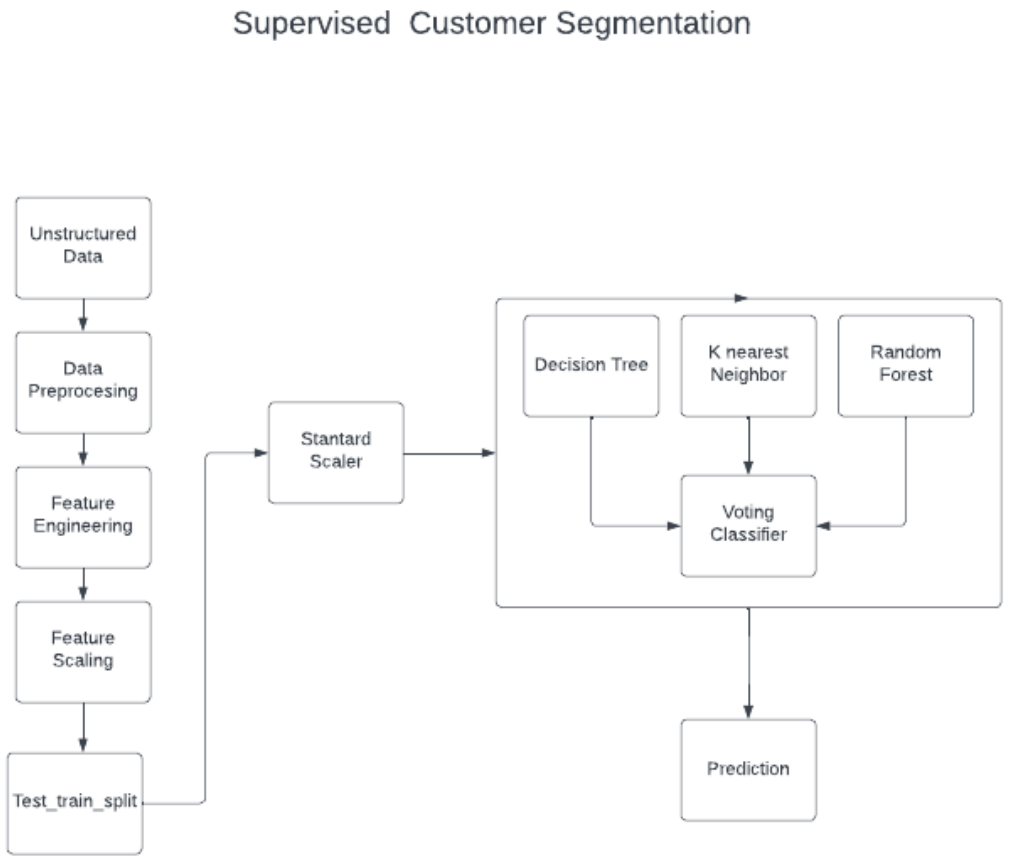
Feature Scaling

- Standardize your train and test sets

Feature Scaling:

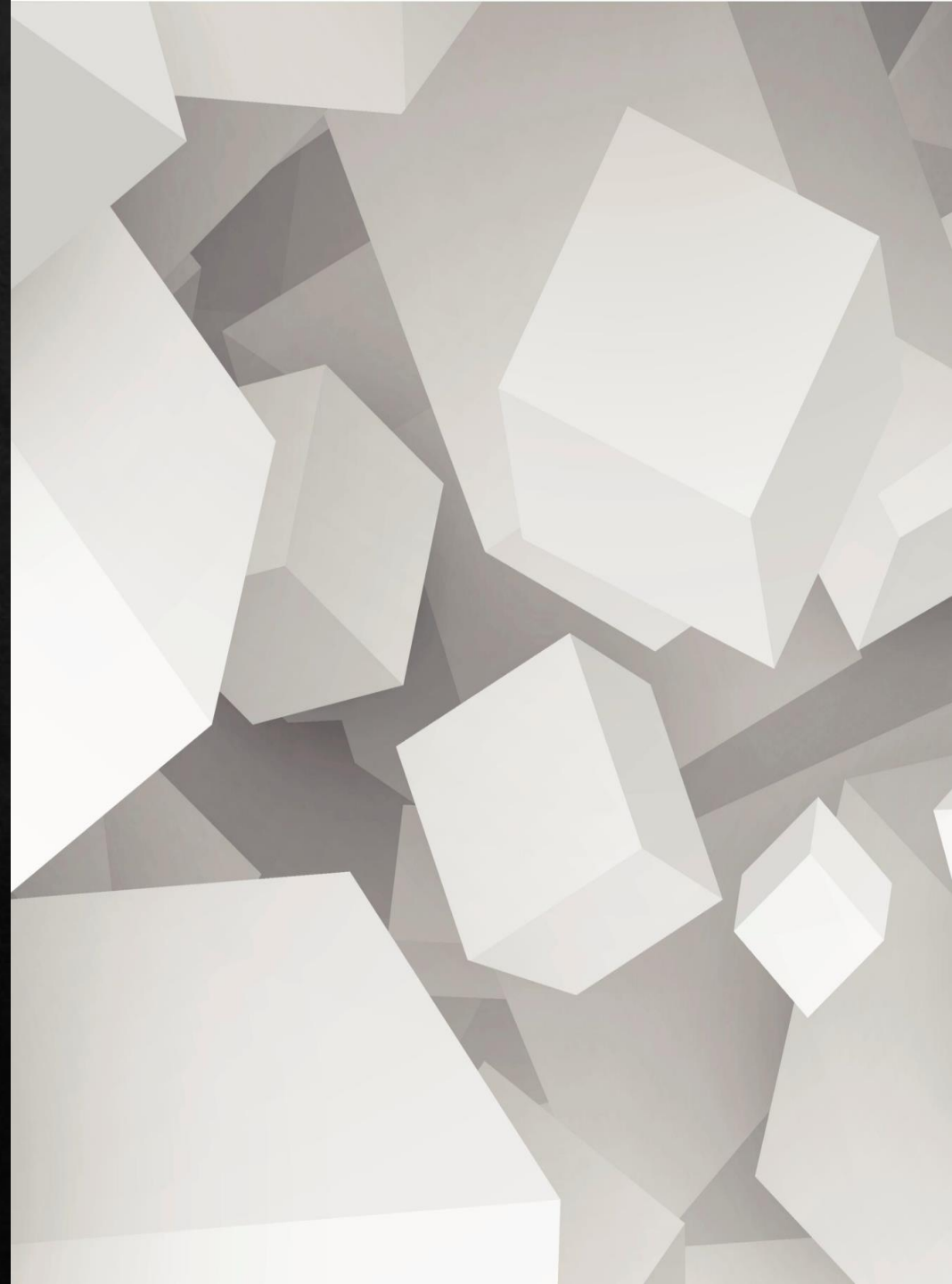
```
scaler = StandardScaler()
X = new_train.drop("Segmentation", axis=1)
X_scaled = pd.DataFrame(scaler.fit_transform(X), columns=X.columns)
```

Experiment Design



Build machine learning models

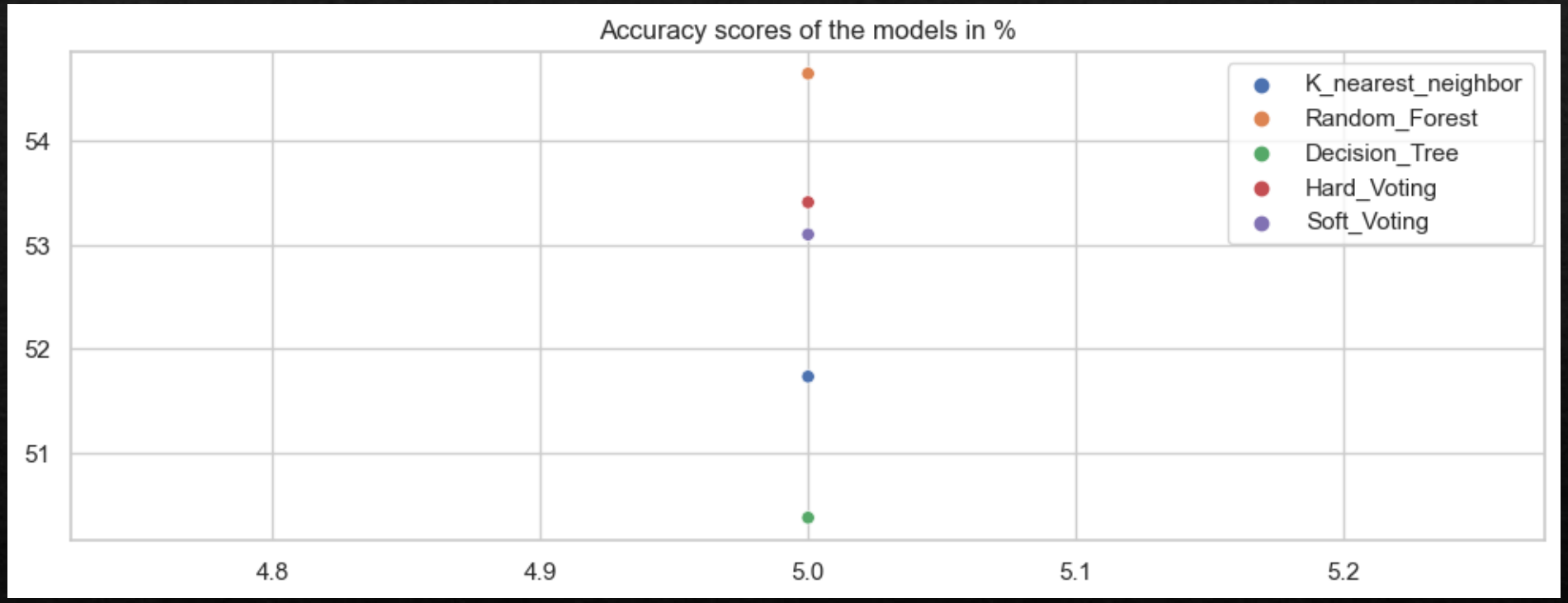
- Support vector machines
- K nearest neighbor
- Random Forest Classifier
- Decision Tree Classifier
- Voting Ensemble Classifier



Accuracy Scores

- SVM : 48.57%
- K nearest neighbor : 51.73%
- Random Forest Classifier : 54.65%
- Decision Tree Classifier : 50.37%
- Voting Ensemble Classifier : 53.41%

Accuracy Scores



Predictions on the unlabeled test set

- The orange column represents the random forest classifier with the highest accuracy score of 54.65%
- The runner up is the voting ensemble (grey column) which scored lower at 53.41%

	KNN_predictions	rdm_predictions	tree_predictions	hard_vot_pred
0	2	3	0	0
1	0	3	0	3
2	0	3	1	0
3	2	0	0	0
4	2	0	0	0
5	2	3	1	0
6	2	0	0	0
7	0	0	0	0
8	2	3	0	0
9	2	0	0	0
10	2	0	0	0
11	2	3	0	0
12	3	3	3	3
13	2	0	0	0
14	3	3	0	3
15	2	0	0	0
16	0	3	0	0
17	3	3	0	3
18	2	0	0	0
19	3	3	0	3
20	2	0	0	0
21	0	3	0	0
22	2	0	0	0

Conclusion

- We used the accuracy score of SVM 48.57% as our baseline
- Subsequent models trained beat the baseline
- The voting ensemble scored 53.41% on the test and will be dropped
- Best performance was by Random Forest classifier with 54.65% accuracy

Thankyou!

- I'll be happy to answer your questions.