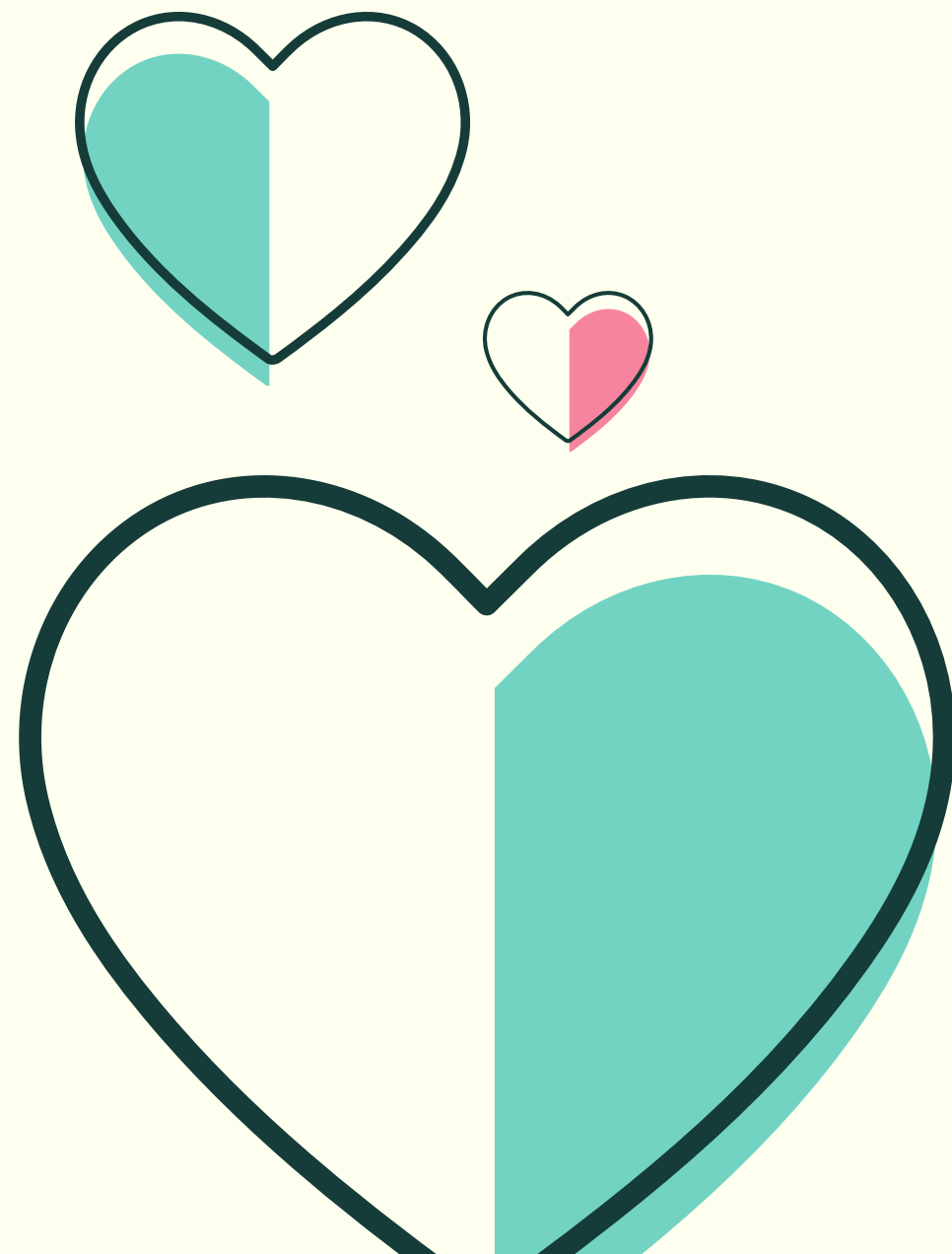


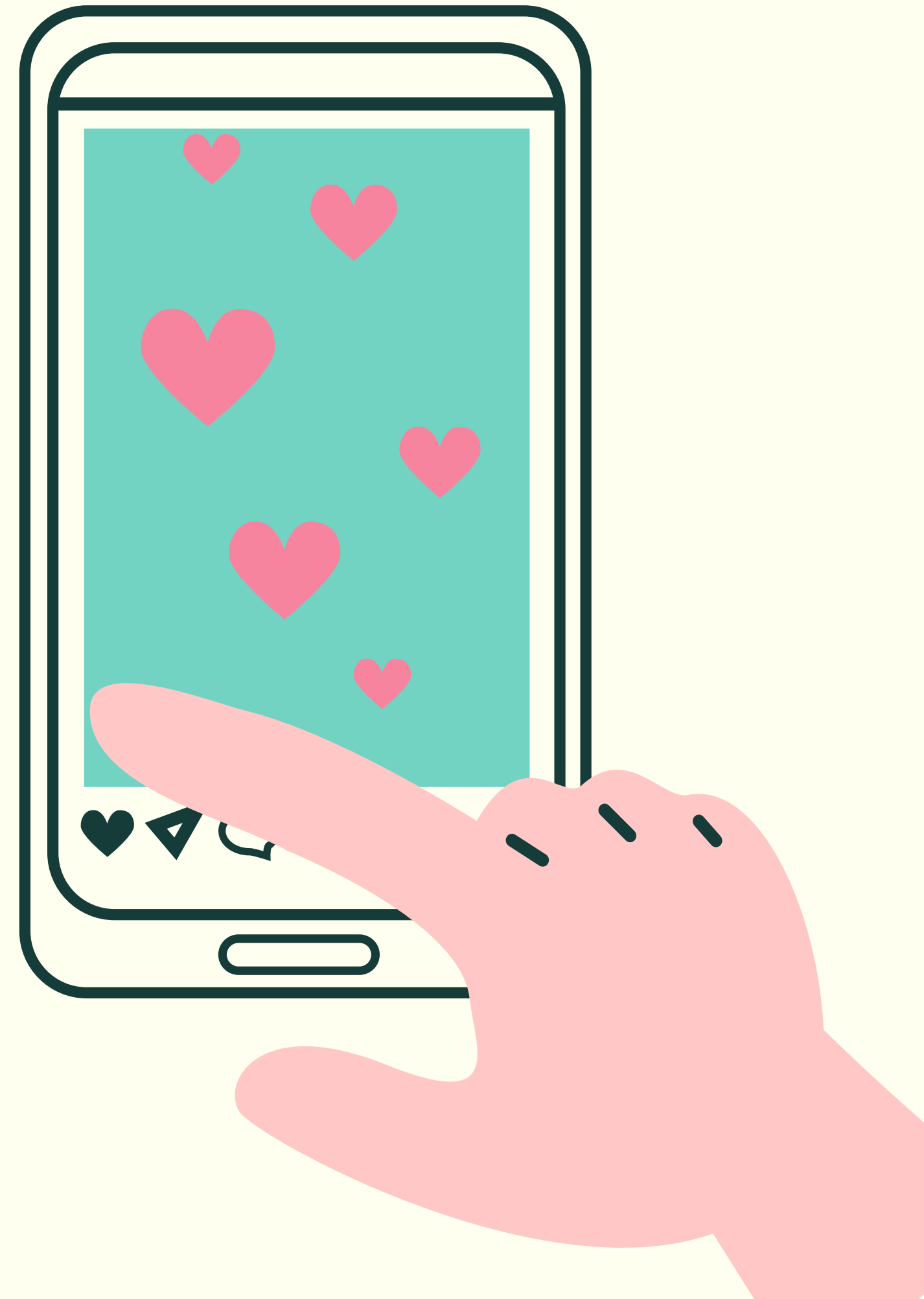
# Social Media


## Ad Click prediction



A curriculum based  
group project.

**2024**





# Project MEMBERS

Ravula.Sandhya

Ande.Sravan Kumar

Chinta.Sai Krishna

Desharaju.Sai Anupam

Battu.Ajay Kumar

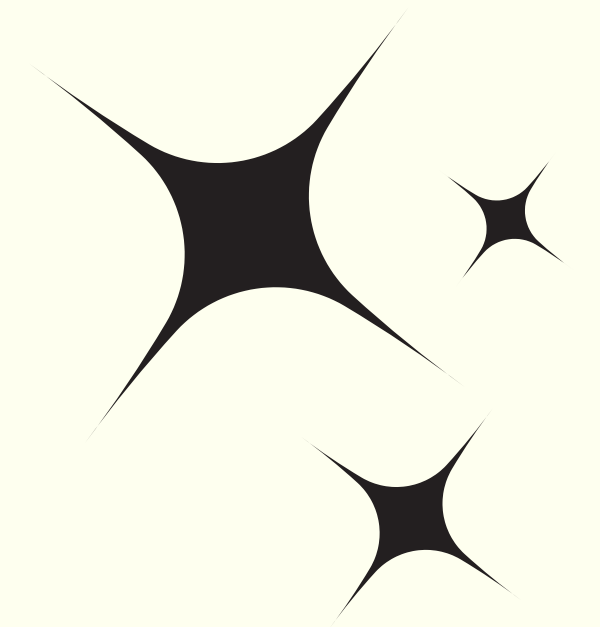
Saggurthi.Yashwanth

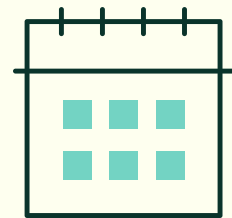
Barathala.Karthik

Samudrala.Sowmya Rani

Dakuri.Sairam Reddy

Azhar.Ahmed Ansari





## Dataset

This is a collection of some thematically related datasets that are suitable for different types of regression analysis. Each set of datasets requires a different technique



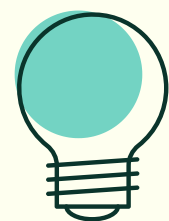
## Data Analysis

Regression analysis is a form of predictive modelling technique which investigates the relationship between a dependent and independent variable



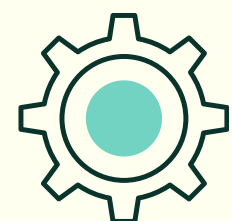
## Data Cleaning

It involves identifying and removing any missing, duplicate, or irrelevant data. The goal of data cleaning is to ensure that the data is accurate, consistent, and free of errors



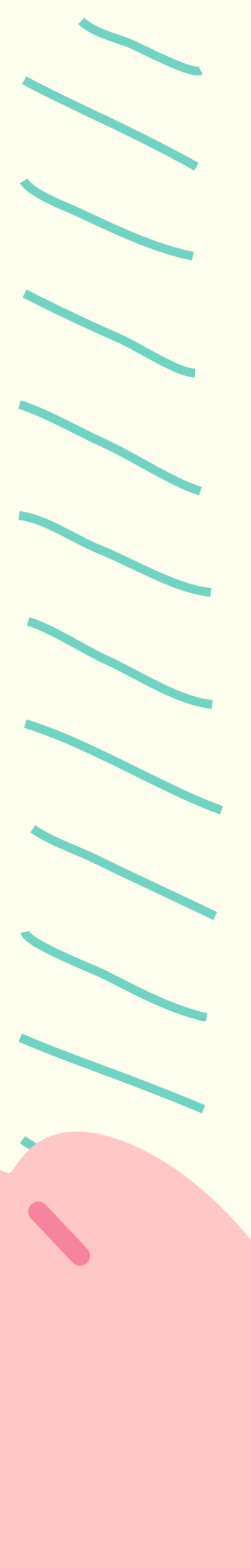
## Implementation

Implementing different type of regression models in order to find the accurate value for the given dataset.



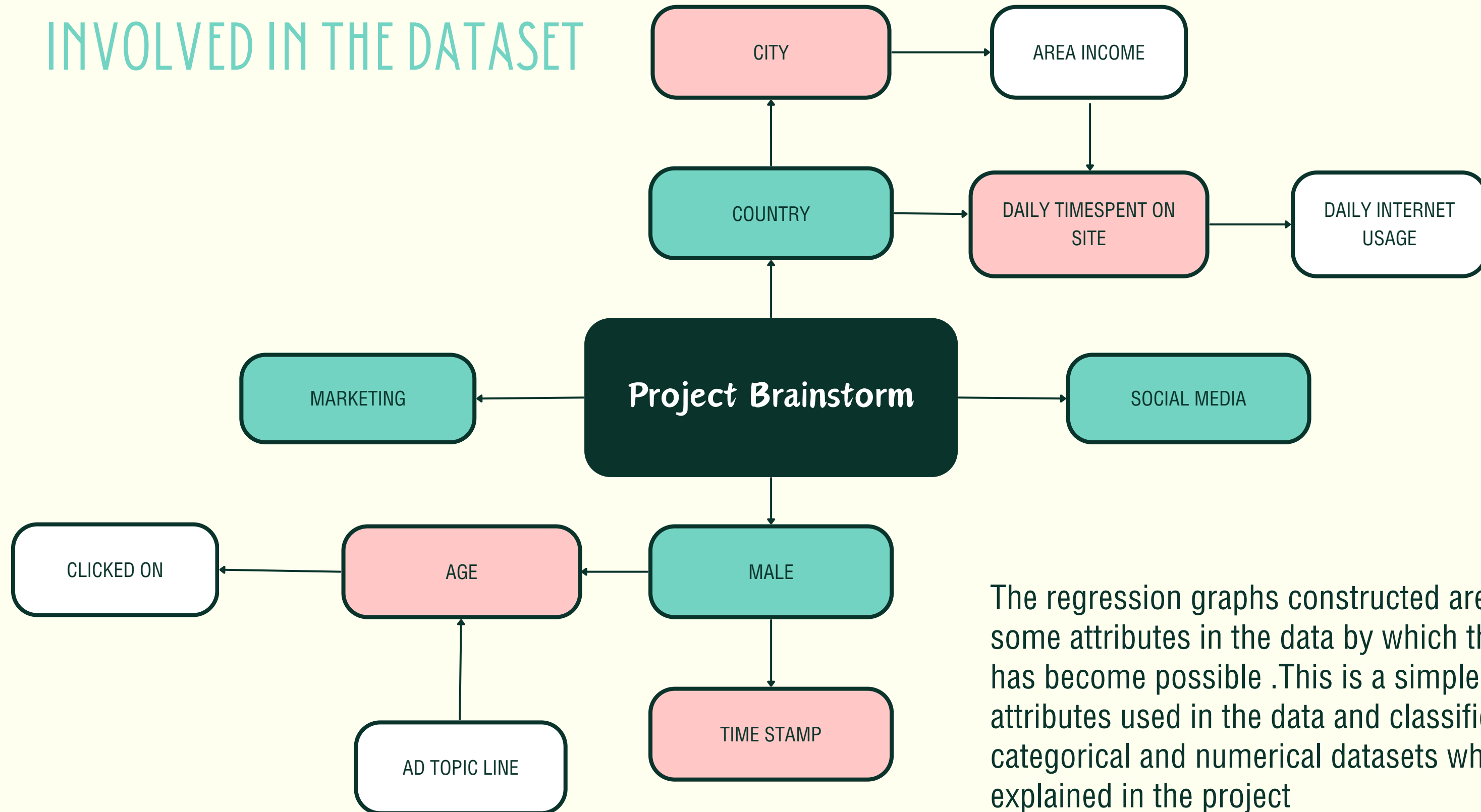
## Accuracy using different models

Different models give different outputs so the accuracy of those models are compared to find which method gives the best accuracy.



# Attributes

INVOLVED IN THE DATASET

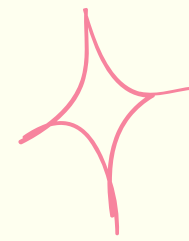


The regression graphs constructed are on the basis of some attributes in the data by which the ad prediction has become possible. This is a simple overview of the attributes used in the data and classified into categorical and numerical datasets which are further explained in the project

# Dataset

10 ATTRIBUTES & VALUES RANGING FROM 1-1000

	Daily Time Spent on Site	Age	Area Income	Daily Internet Usage	Ad Topic Line	City	Male	Country	Timestamp	Clicked on Ad
0	68.95	35	61833.90	256.09	Cloned 5thgeneration orchestration	Wrightburgh	0	Tunisia	2016-03-27 00:53:11	0
1	80.23	31	68441.85	193.77	Monitored national standardization	West Jodi	1	Nauru	2016-04-04 01:39:02	0
2	69.47	26	59785.94	236.50	Organic bottom-line service-desk	Davidton	0	San Marino	2016-03-13 20:35:42	0
3	74.15	29	54806.18	245.89	Triple-buffered reciprocal time-frame	West Terrifurt	1	Italy	2016-01-10 02:31:19	0
4	68.37	35	73889.99	225.58	Robust logistical utilization	South Manuel	0	Iceland	2016-06-03 03:36:18	0
...	...	...	...	...	...	...	...	...	...	...
995	72.97	30	71384.57	208.58	Fundamental modular algorithm	Duffystad	1	Lebanon	2016-02-11 21:49:00	1
996	51.30	45	67782.17	134.42	Grass-roots cohesive monitoring	New Darlene	1	Bosnia and Herzegovina	2016-04-22 02:07:01	1
997	51.63	51	42415.72	120.37	Expanded intangible solution	South Jessica	1	Mongolia	2016-02-01 17:24:57	1
998	55.55	19	41920.79	187.95	Proactive bandwidth-monitored policy	West Steven	0	Guatemala	2016-03-24 02:35:54	0
999	45.01	26	29875.80	178.35	Virtual 5thgeneration emulation	Ronniemouth	0	Brazil	2016-06-03 21:43:21	1



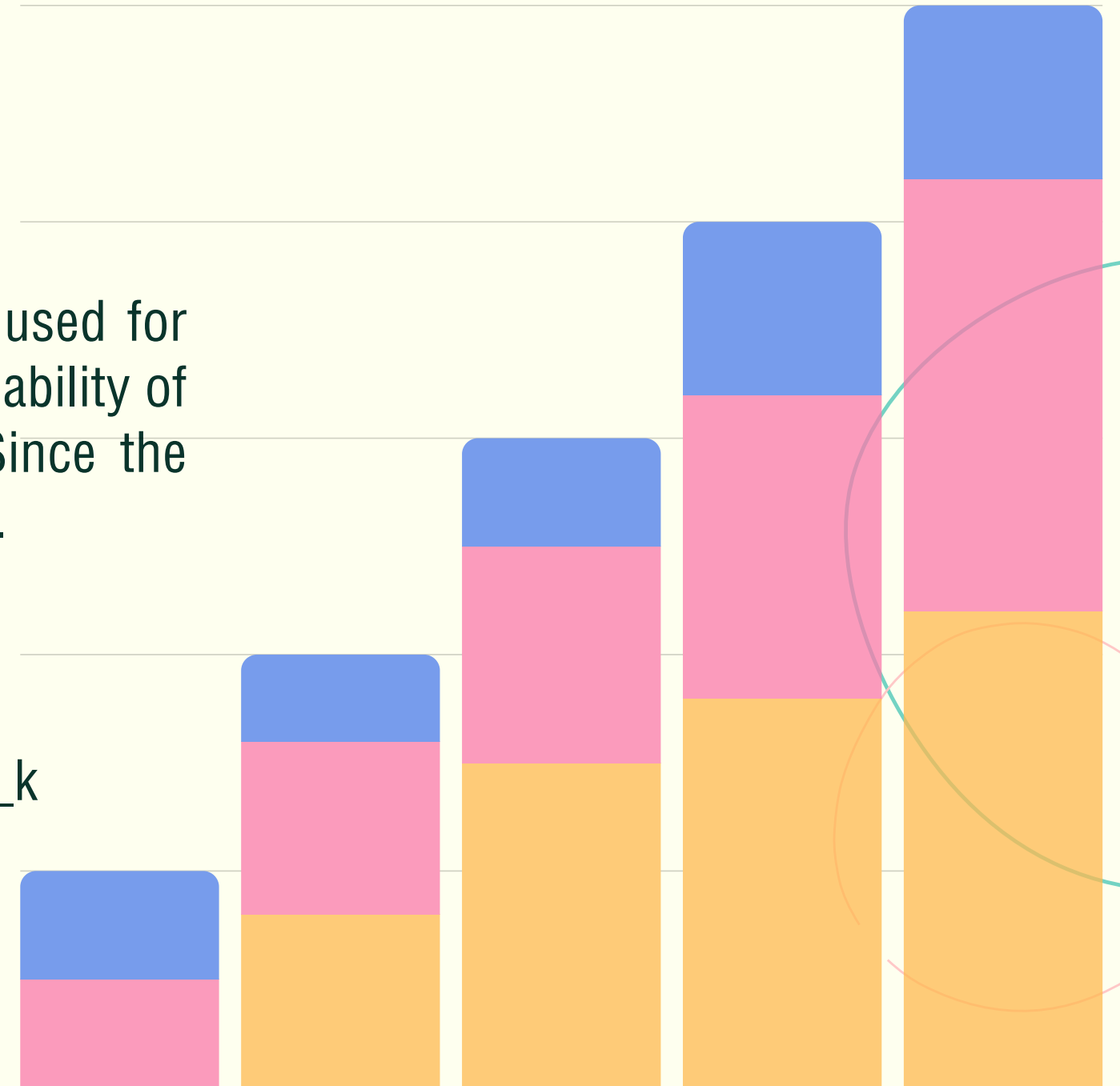
# Introduction

## Logistic Regression

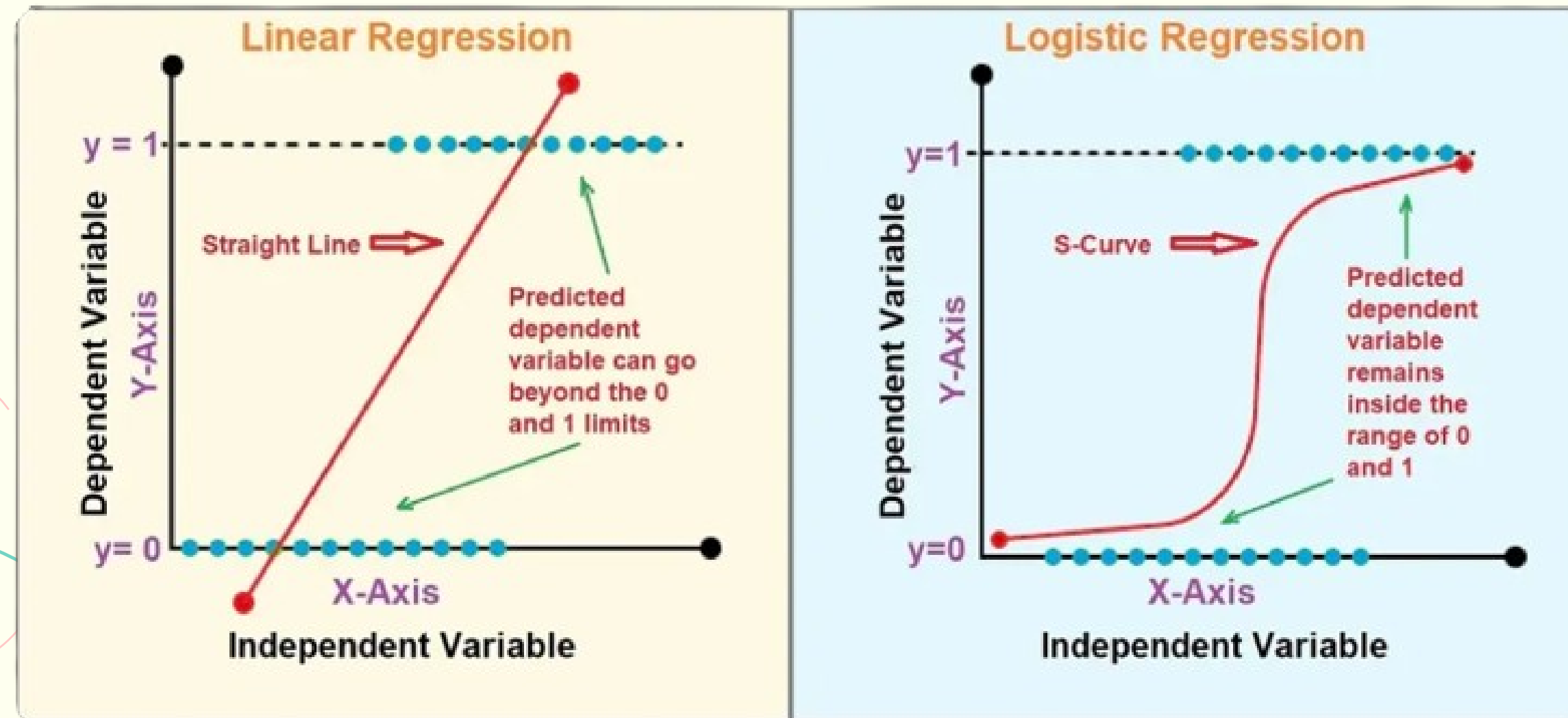
This type of statistical model is also known as logit model which is often used for classification and predictive analytics. Logistic regression estimates the probability of an event occurring, based on a given dataset of independent variables. Since the outcome is a probability, the dependent variable is bounded between 0 and 1.

logistic function is represented by the following formulas:

$$\text{Logit}(\pi) = \ln(\pi/(1-\pi)) = \text{Beta}_0 + \text{Beta}_1 * X_1 + \dots + B_k * K_k$$



# Linear & Logistic Regression



# Implementing

## Logistic Regression Model

### Splitting Dataset

```
In [0]: from sklearn.model_selection import train_test_split
```

```
In [0]: X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.3, random_state = 42)
```

### Implementing Logistic Regression Model

```
In [0]: from sklearn.linear_model import LogisticRegression
```

```
In [0]: log_reg_model = LogisticRegression()
```

```
In [0]: log_reg_model.fit(X_train, Y_train)
```

```
Out[0]: LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
    intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
    penalty='l2', random_state=None, solver='liblinear', tol=0.0001,
    verbose=0, warm_start=False)
```

```
In [0]: log_reg_pred = log_reg_model.predict(X_test)
```

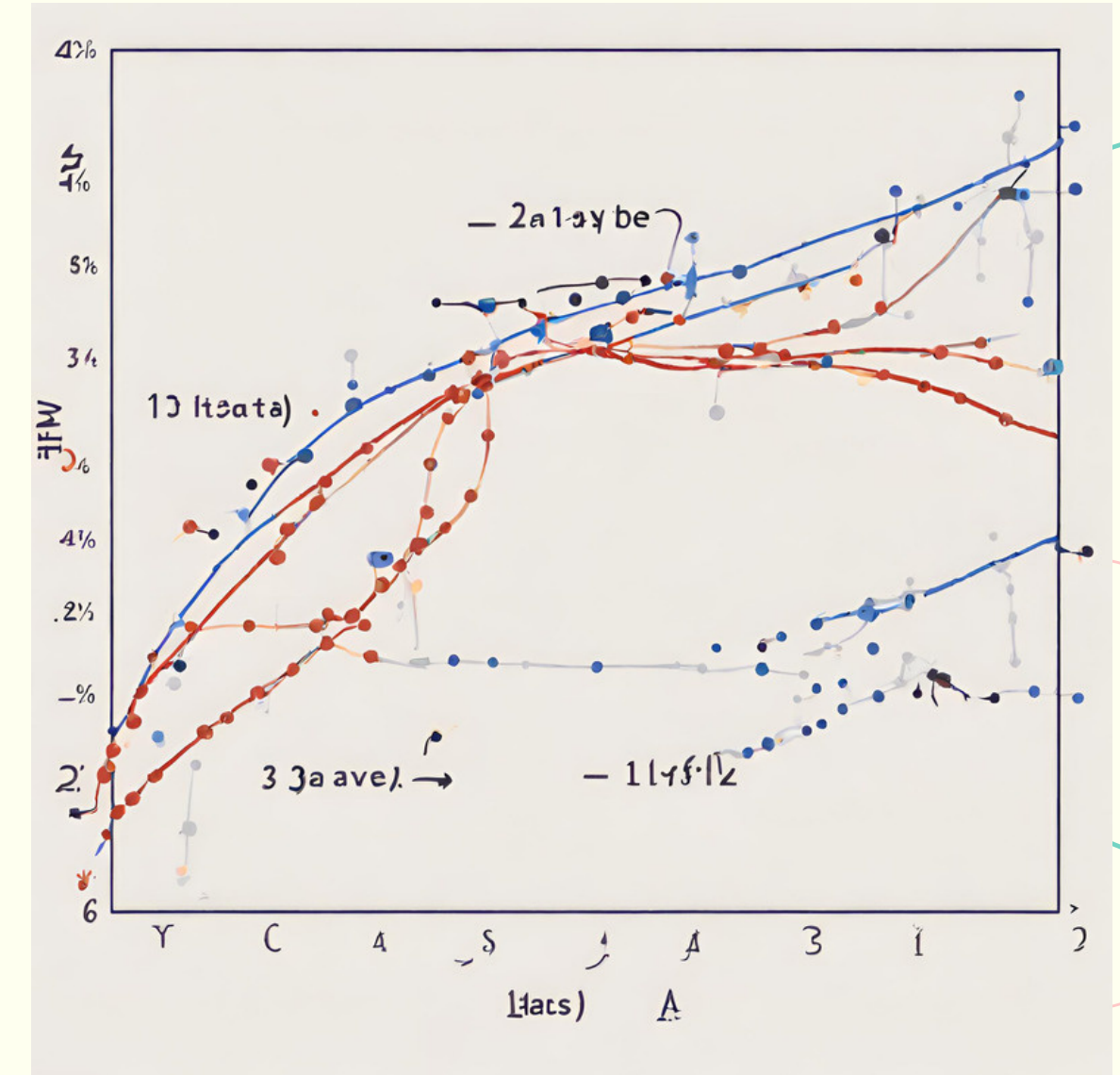


# Naive Bayes

The Naive Bayes classifier is a supervised machine learning algorithm, which is used for classification tasks. It is also part of a family of generative learning algorithms. It also uses Bayes' theorem to learn the probability of objects, their features, and which groups they belong to. It is also known as a probabilistic classifier.

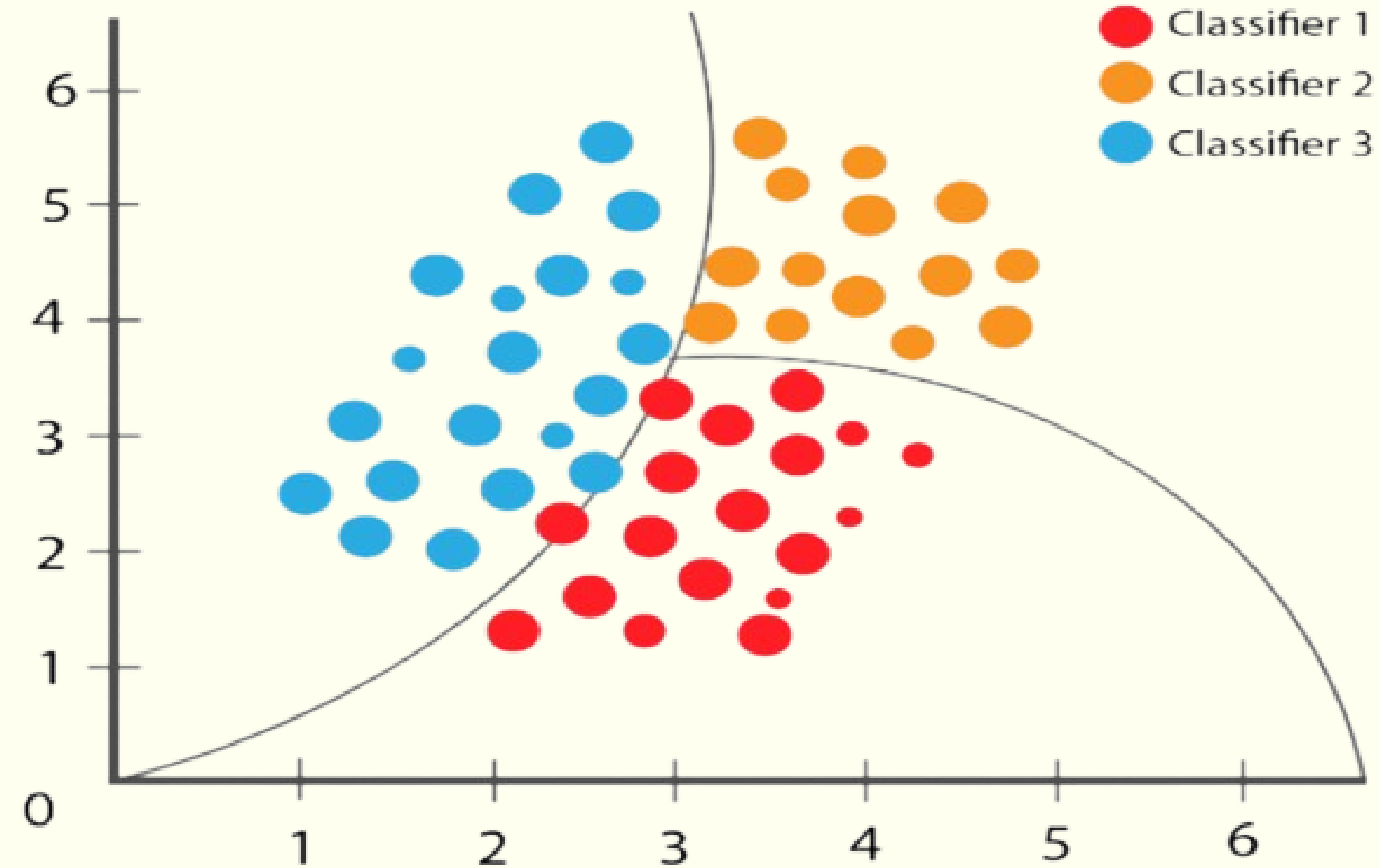
Bayes Theorem provides a principled way for calculating the conditional probability. The simple form of the calculation for Bayes Theorem is :

$$P(A|B) = P(B|A) * P(A) / P(B)$$



# Naive Bayes

Naive bayes classifier





# Implementing

# Naive Bayes Model

## Implementing Naive Bayes Model

```
In [30]: from sklearn.naive_bayes import GaussianNB
```

```
In [31]: nav_bayes_model = GaussianNB()
```

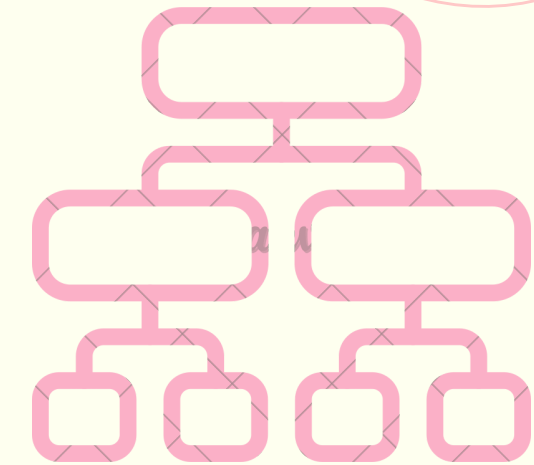
```
In [32]: nav_bayes_model.fit(X_train, Y_train)
```

```
Out[32]: GaussianNB()  
          ()
```

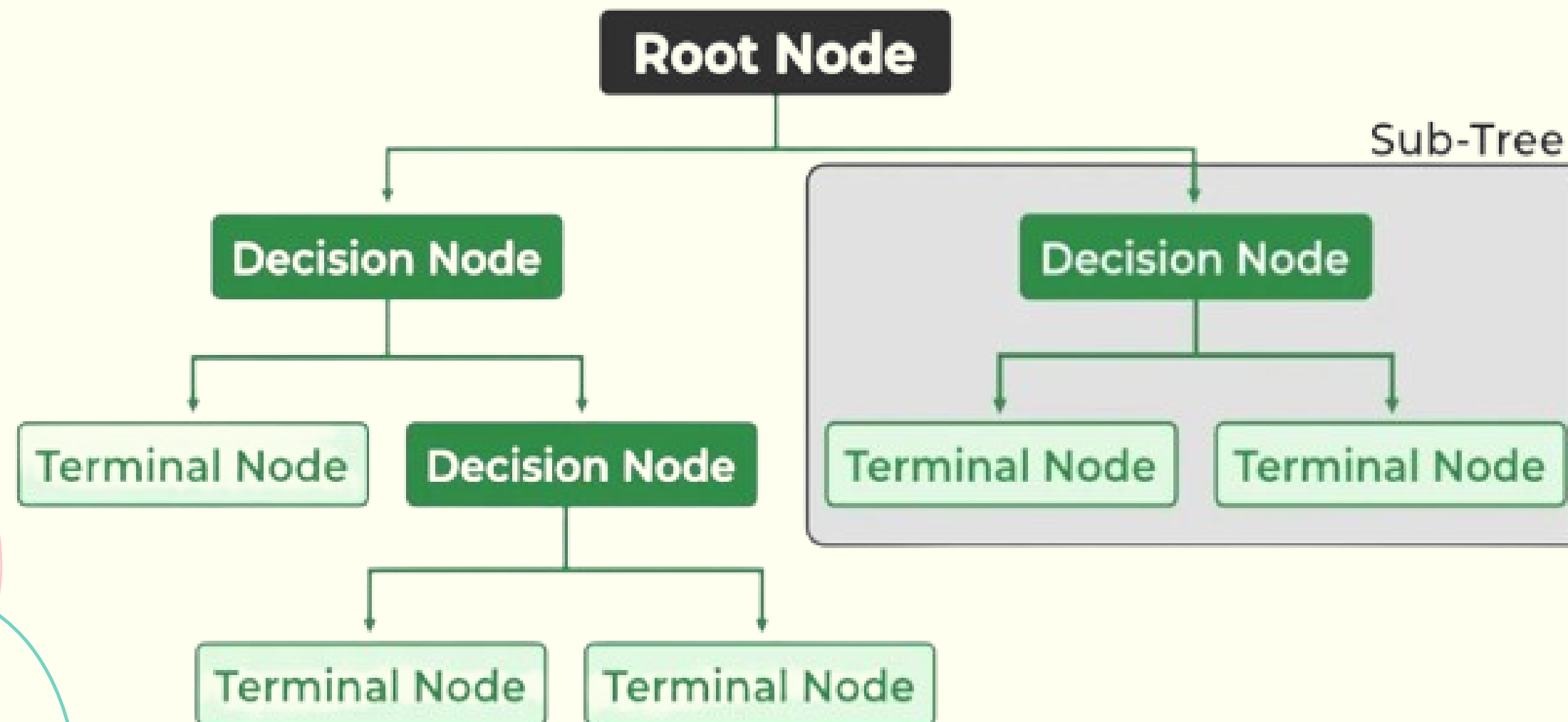
```
In [33]: nav_bayes_pred = nav_bayes_model.predict(X_test)
```

# Decision Tree

A decision tree is a non-parametric supervised learning algorithm, which is utilized for both classification and regression tasks. It has a hierarchical, tree structure, which consists of a root node, branches, internal nodes and leaf nodes. It is also used in Random Forest to train on different subsets of training data, which makes random forest one of the most powerful algorithms in machine learning.



# Decision Tree Model





# Implementing

# Decision Tree Model

## Implementing Decision Tree Model

```
In [34]: from sklearn.tree import DecisionTreeClassifier
```

```
In [35]: dec_tree_model = DecisionTreeClassifier()
```

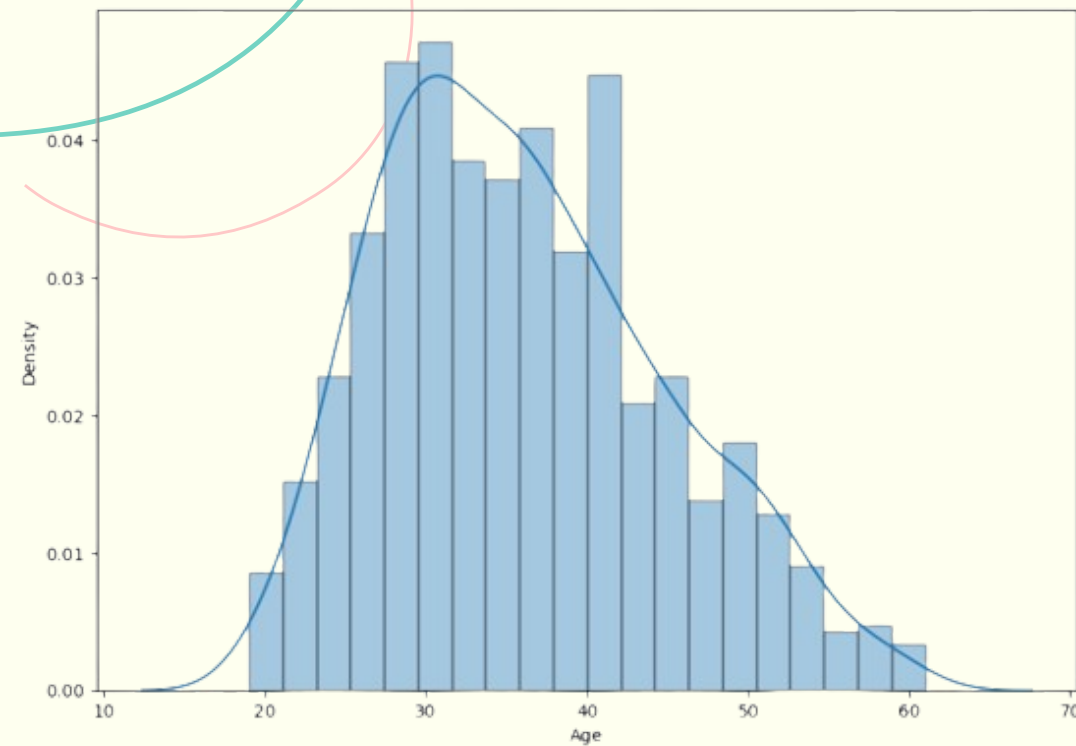
```
In [36]: dec_tree_model.fit(X_train, Y_train)
```

```
Out[36]: DecisionTreeClassifier()
```

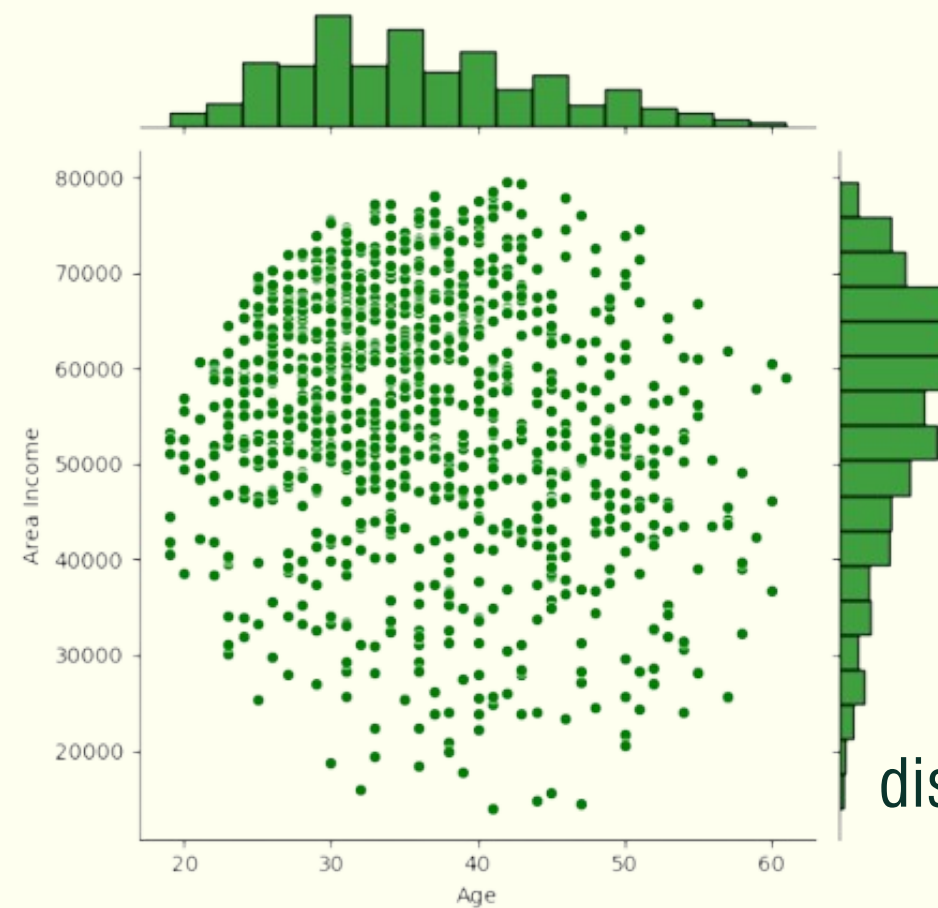
```
In [37]: dec_tree_pred = dec_tree_model.predict(X_test)
```



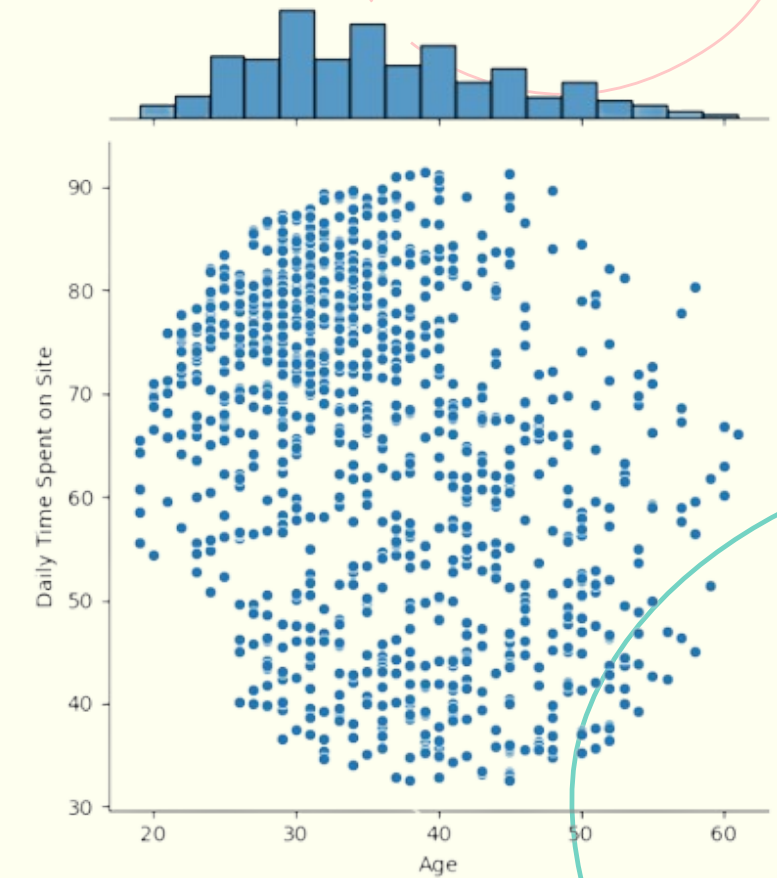
# DATA ANALYSIS



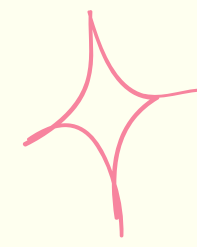
What age group does the dataset majorly consist of?



What is the income distribution in different age groups?

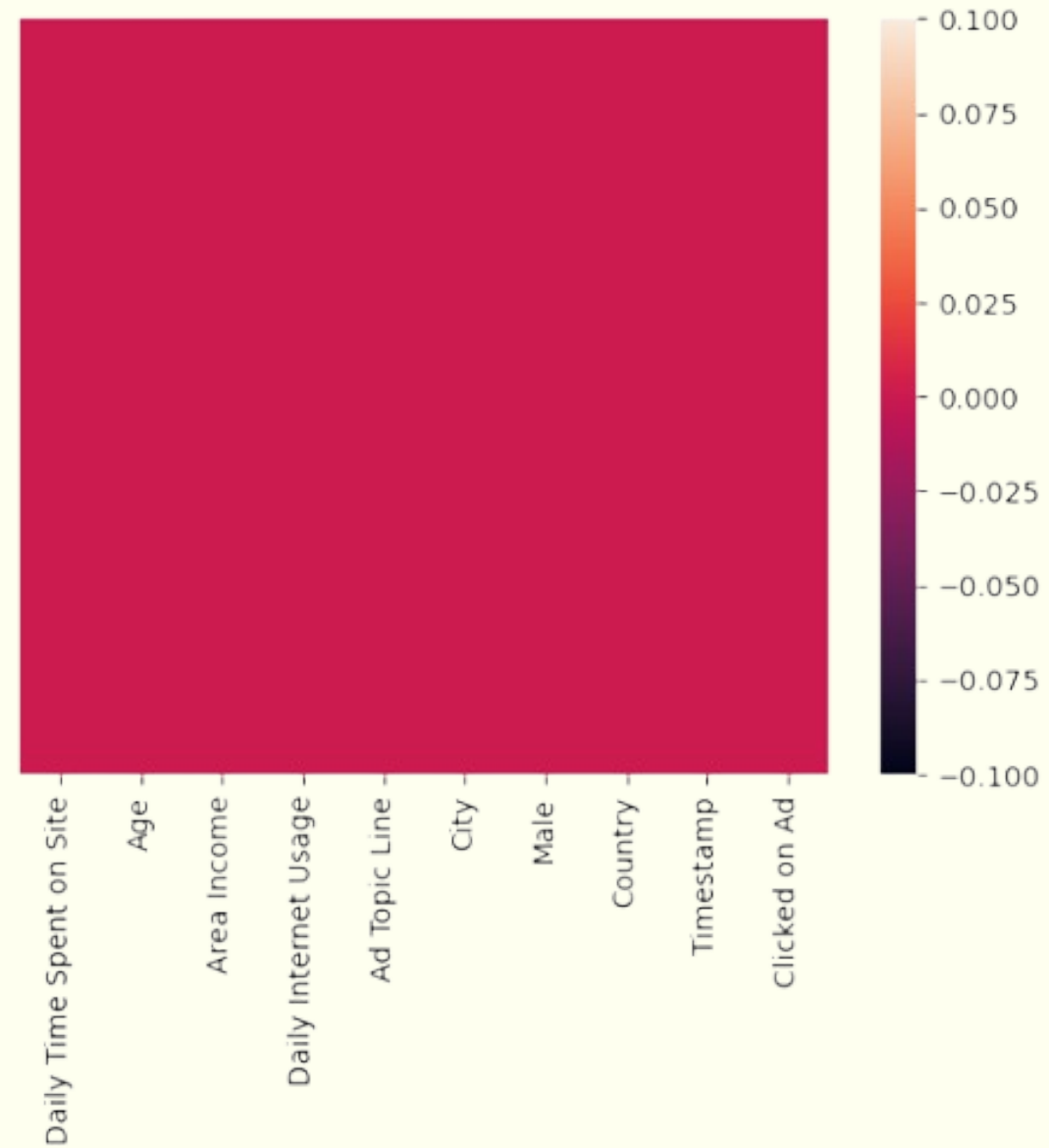


Which age group is spending maximum time on the internet?



# DATA CLEANING

zero error because of no duplicates





# Checking Accuracy

## IN EACH MODEL

Finding accuracy in each model

```
In [38]: from sklearn.metrics import accuracy_score
```

**Logistic Regression**

```
In [39]: log_reg_accuracy = accuracy_score(log_reg_pred, Y_test)
print(log_reg_accuracy*100)
```

98.66666666666666

**Naive Bayes**

```
In [40]: nav_bayes_accuracy = accuracy_score(nav_bayes_pred, Y_test)
print(nav_bayes_accuracy*100)
```

96.0

**Decision Tree**

```
In [41]: dec_tree_accuracy = accuracy_score(dec_tree_pred, Y_test)
print(dec_tree_accuracy*100);
```

93.33333333333333

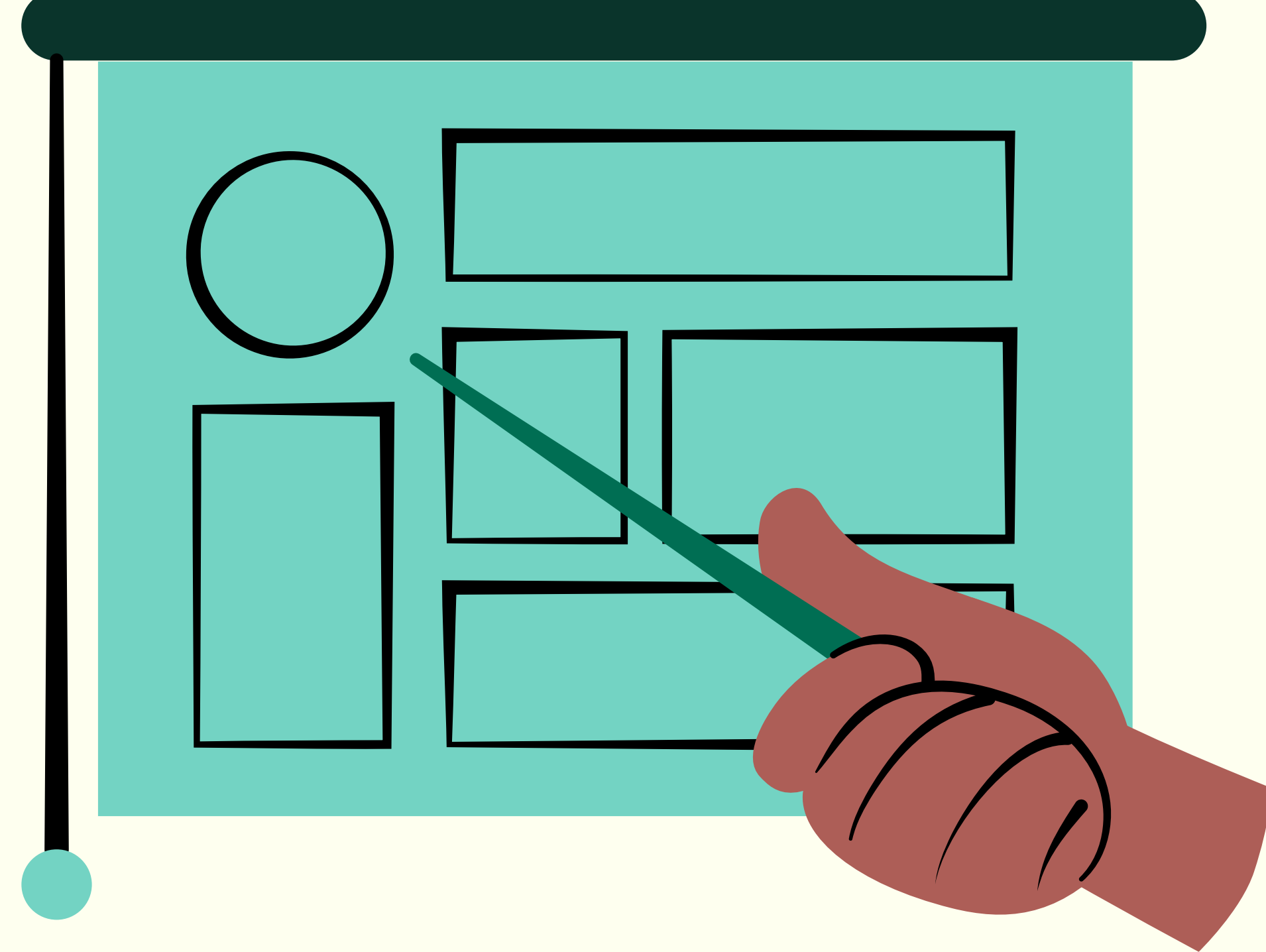
# ACCURACY

## In Different Models



	Logistic	Decision Tree	Naive Bayes
Accuracy in %	90.6	93.3	96.0
Number of Users	1000	1000	1000
Product Launch	✓	✓	✓
Level	Low	Moderate	High
Feedback	★	★ ★	★ ★ ★

- In this project different methods with a different logics have been performed on a specified dataset in order to calculate the accurate value of the output produced by those models for the further usage of the data.
- **Logistic Regression Model** : It estimates the probability of a binary outcome. May not perform well if the relationship between predictors and the target is highly non-linear. The accuracy produced is 90.666 !!!
- **Naive Bayes Model** : Naive Bayes is based on probabilistic principles. Requires a small amount of training data. Well-suited for categorical data and text classification tasks. The accuracy produced is 96.0 !!!
- **Decision Tree Model** : Decision Trees are capable of capturing complex non-linear relationships in the data. Can handle missing values in the dataset without requiring imputation. The accuracy produced is 93.333 !!!



# Project Results

# Any Queries?

Related to the presentation..!





*Thank you*

*for watching!*

**By : Team**