SRI SIDDHARTHA INSTITUTE OF TECHNOLOGY

MARALURU, TUMAKURU-572103

(A Constituent college of Sri Siddhartha Academy of Higher Education, Deemed to be University)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Report On

"IPHONE SALES PREDICTION"

Submitted in partial fulfilment of the requirement for the completion of VI semester of

BACHELOR OF ENGINEERING

Submitted by:

PRATHUASHA K B (20CS058)

Under the guidance of:

SINDHU T N
Assistant Professor
Dept. of CSE
SSIT, TUMKURU

MADHUMALA G
Assistant Professor
Dept. of CSE
SSIT, TUMKURU

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

2022-23

iPhone Sales Prediction

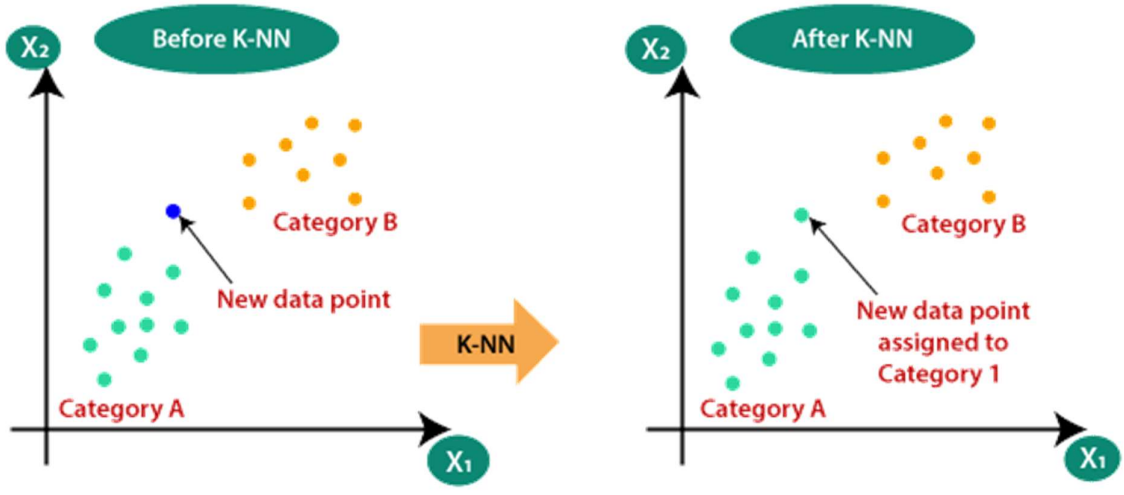
K-Nearest Neighbours (K-NN) Algorithm

K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique. K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.

This algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K-NN algorithm. It is a **non-parametric algorithm**, which means it does not make any assumption on underlying data. It is also called a **lazy learner algorithm** because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.

Why do we need a K-NN Algorithm?

Suppose there are two categories, i.e., Category A and Category B, and we have a new data point x1, so this data point will lie in which of these categories. To solve this type of problem, we need a K-NN algorithm. With the help of K-NN, we can easily identify the category or class of a particular dataset. Consider the below diagram:



Working of K-NN Algorithm

The K-NN working can be explained based on the below algorithm:

- **Step-1:** Select the number K of the neighbors
- **Step-2:** Calculate the Euclidean distance of **K number of neighbors**
- **Step-3:** Take the K nearest neighbors as per the calculated Euclidean distance.

- **Step-4:** Among these k neighbors, count the number of the data points in each category.
- **Step-5:** Assign the new data points to that category for which the number of the neighbors is maximum.

Overview of Project

iPhone sales prediction refers to the process of estimating or forecasting the future sales performance of iPhones. iPhone sales prediction is vital for effective planning, decision-making, and optimization of various aspects of Apple's business operations. It empowers them to align their production, marketing, and supply chain strategies, respond to market dynamics, and stay competitive in the rapidly evolving smartphone industry. Hence this process involves analysing historical sales data, identifying patterns and trends, and using statistical or machine learning techniques to make informed predictions about future sales.

Several algorithms can be utilized for iPhone sales prediction like Linear Regression, Time Series Analysis, Decision Trees, Neural Networks, K-Nearest Neighbours (K-NN) and so on, depending on the nature of the data and the specific requirements of the prediction task. Here is an attempt of iPhone sales prediction using K-NN algorithm, that classifies or predicts based on the similarity to neighbouring data points. It can be applied to sales prediction by finding the most similar historical instances to make forecasts based on their sales outcomes.

Objective : *We have a dataset that shows which users have purchased or did not purchase an iPhone. Our goal in this project is to predict if the customer will purchase an iPhone or not given their gender, age and salary using K-NN algorithm.*

Code Implementation

```
#Load Data

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt


dataset = pd.read_csv("iphone_purchase_records.csv")

dataset.info()
```

```
[OP]: <class 'pandas.core.frame.DataFrame'>
      RangeIndex: 400 entries, 0 to 399
      Data columns (total 4 columns):
       #   Column           Non-Null Count  Dtype
      ---  ------           --------------  -----
       0   Gender           400 non-null    object
```

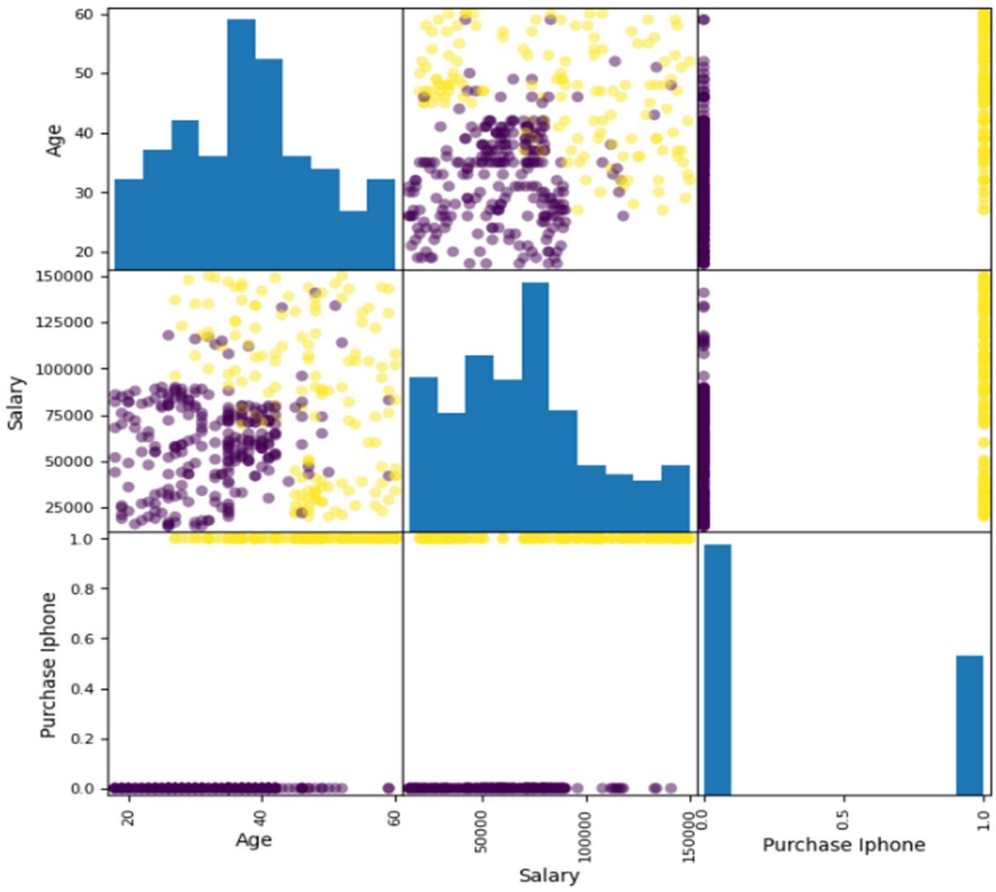
```
 1   Age              400 non-null    int64
 2   Salary           400 non-null    int64
 3   Purchase Iphone  400 non-null    int64
dtypes: int64(3), object(1)
memory usage: 12.6+ KB
```

```
dataset.drop('Gender', axis=1, inplace=True)

dataset.info()
```

[OP]:
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 3 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Age              400 non-null    int64
 1   Salary           400 non-null    int64
 2   Purchase Iphone  400 non-null    int64
dtypes: int64(3)
memory usage: 9.5 KB
```

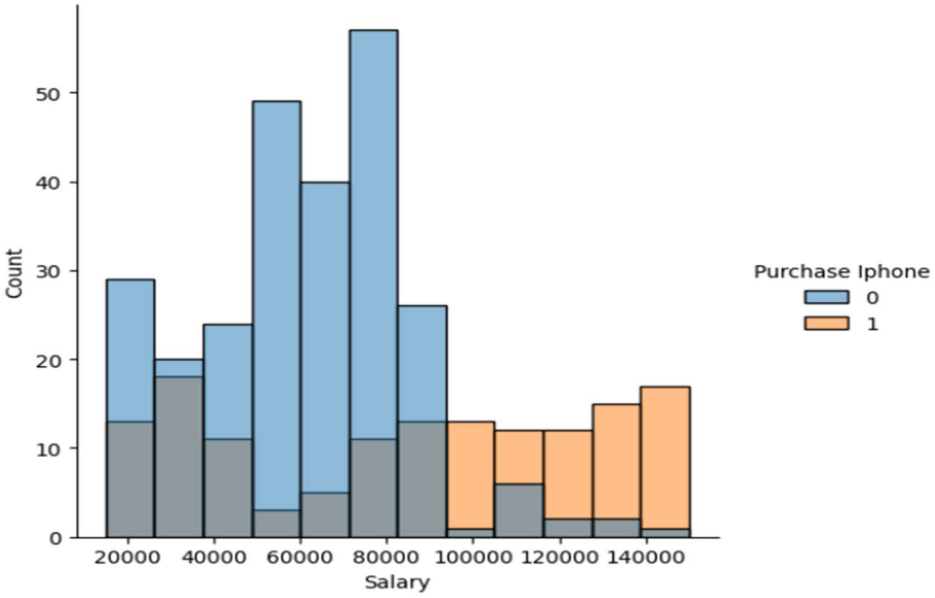
```
X = dataset.drop('Purchase Iphone', axis=1).values

Y = dataset['Purchase Iphone'].values


#Exploring data

pd.plotting.scatter_matrix(dataset, c=Y, figsize=[8,8], s=150)
```

[OP]:
```
array([[<Axes: xlabel='Age', ylabel='Age'>,
        <Axes: xlabel='Salary', ylabel='Age'>,
        <Axes: xlabel='Purchase Iphone', ylabel='Age'>],
       [<Axes: xlabel='Age', ylabel='Salary'>,
        <Axes: xlabel='Salary', ylabel='Salary'>,
        <Axes: xlabel='Purchase Iphone', ylabel='Salary'>],
       [<Axes: xlabel='Age', ylabel='Purchase Iphone'>,
        <Axes: xlabel='Salary', ylabel='Purchase Iphone'>,
        <Axes: xlabel='Purchase Iphone', ylabel='Purchase Iphone'>]],
      dtype=object)
```



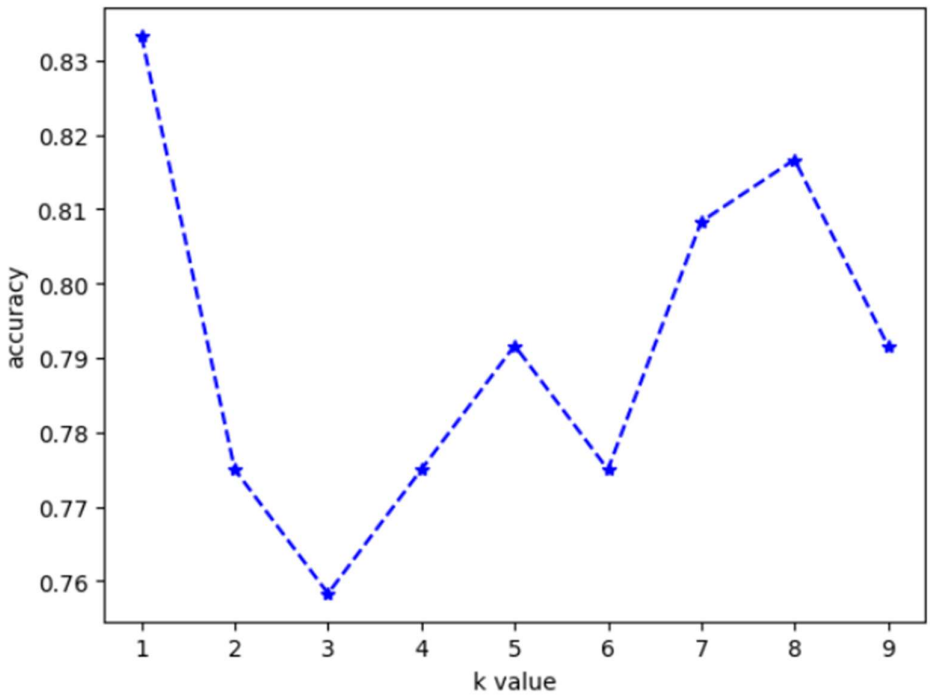

```
sns.displot(dataset, x='Salary', hue='Purchase Iphone')
```

[OP]: <seaborn.axisgrid.FacetGrid at 0x7f64cd5a09a0>



```
#split data into training and test set
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
from sklearn import metrics

X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.3,
random_state=18, stratify=Y)

#Make predictions using KNN algorithm
accuracy = []
for i in range(1,10):
  knn = KNeighborsClassifier(n_neighbors=i).fit(X_train,Y_train)
  Y_pred = knn.predict(X_test)
  accuracy.append(metrics.accuracy_score(Y_test, Y_pred))
plt.plot(range(1,10),accuracy, color='blue', linestyle='dashed',
marker='*')
plt.xlabel('k value')
plt.ylabel('accuracy')
```

[OP] : Text(0, 0.5, 'accuracy')



```
#Check Accuracy of the Predictions
knn = KNeighborsClassifier(n_neighbors=8).fit(X_train, Y_train)
Y_pred = knn.predict(X_test)
metrics.accuracy_score(Y_test, Y_pred)
[OP] : 0.8166666666666667
```

Results

Upon executing the code, the following results were obtained :