काशी हिन्दू विश्वविद्यालय

BANARAS HINDU UNIVERSITY

DST-CIMS, Institute of Science

A Project report on

Expensive Product Purchase Prediction and Analysis

Submitted in partial completion of Master of Science in Statistics and Computing, 2022

Submitted by :

Name – Subhajit Das

Roll no. – 20419STC028

Enrollment no. – 428410

Paper code – MSMS408

DST-CIMS, Institute of Science

Banaras Hindu University

Project Supervisor :

Dr. Abhimanyu Singh Yadav

Assistant Professor

Department of Statistics

Institute of Science

Banaras Hindu University

ACKNOWLEDGEMENT

Place: Varanasi

Date: May, 2022

I would like to express my heartfelt gratitude to my supervisor, **Ass. Professor Abhimanyu Singh Yadav**, for sharing his guidance, helpful insights, innovative ideas, and unceasing experiences that have been beneficial all throughout my project. His expertise, pure and immense knowledge in the field of statistics has helped me in successfully completing this project. This has been a fruitful learning experience for me and I have improvised a lot throughout its preparation.

I would like to express my sincere gratitude to several individuals and **Banaras Hindu University** for supporting me throughout my Project. Hence, I am ensuring that this project has been successfully completed.

Last but not least, I express my gratitude for all my friends in Banaras Hindu University for their direct or indirect help in completing this project.

Date **:**

Name - Subhajit Das
M.Sc. in Statistics &Computing
Semester-IV
DST-CIMS, BHU

Certificate

This is to certify that **Subhajit Das**, student of Master of Science in Statistics & Computing, semester IV of the Department of DST-CIMS, Institute of Science, Banaras Hindu University, bearing exam roll no. **20419STC028** and enrolment no. **428410** of 2020-2022 has successfully completed his dissertation project entitled "***Expensive Product Purchase Prediction and Analysis***".

The project was completed under my direct supervision and guidance for the partial fulfilment of the course curriculum of Master of Science in Statistics & Computing. He carried out the whole work with utmost sincerity and the present report is the outcome of his sincere efforts and endeavours.

This work either in the present form or in part in any other form is not being submitted anywhere or for any other degree or diploma to the best of my knowledge.

Date:

Date: Dr. Abhimanyu Singh Yadav
Assistant Professor
Department of Statistics
Banaras Hindu University

Declaration

I, Subhajit Das, hereby undertake that the dissertation entitled "***Expensive Product Purchase Prediction and Analysis***" is the result of the postgraduate project work carried out by me, at the Department of DST-CIMS, Institute of Science, Banaras Hindu University, under the guidance of Dr. Abhimanyu Singh Yadav, assistant professor, Department of Statistics, Institute of Science, Banaras Hindu University.

I have collected the data from secondary sources as per the requirement of my dissertation, which are appropriately referred in the report. All the computations involved in this dissertation are results of my own calculations on the data that I collected.

The formulae used in this dissertation are acknowledged providing appropriate reference of the source from which they are obtained. No part of the dissertation is being submitted to any other institution for the purpose of any degree / diploma.

Date : Subhajit Das

Contents:

1.Abstract :

For my project, I am using machine learning techniques to predict whether a customer will buy an i-phone or not (I-phone is one of the costliest products available in today's market). I am trying to perform my prediction using factors related to the customer which would help provide insights on the customers I am dealing with. This would benefit the companies producing such costly product to get a vivid idea regarding their available market(customers). My purpose is the analyse if a customer will buy the product based on the customers age, sex and salary. Thus the company would be able to get a precise idea regarding which category of customers they should approach in order the increase the sales of their individual products. I have performed a detailed analysis on i-phone but it can be replaced by various other costly products such as OnePlus products, Mac-books etc. This describes the dynamic nature of this project and how beneficial it would turn out to certain product-based market holders.

Keywords : Machine Learning, Logistic Regression, KNN Classifier , Naïve Bayes Classifier, Accuracy score , Precision score , Recall score.

2. Introduction :

The term ***machine learning*** was coined in 1959 by Arthur Samuel, an American IBMer and pioneer in the field of computer gaming and artificial intelligence. Also the synonym *self-teaching computers* was used in this time period. A representative book of the machine learning research during the 1960s was the Nilsson's book on Learning Machines, dealing mostly with machine learning for pattern classification. Interest related to pattern recognition continued into the 1970s, as described by Duda and Hart in 1973. In 1981 a report was given on using teaching strategies so that a neural network learns to recognize 40 characters (26 letters, 10 digits, and 4 special symbols) from a computer terminal.

Tom M. Mitchell provided a widely quoted, more formal definition of the algorithms studied in the machine learning field: "A computer program is said to learn from experience *E* with respect to some class of tasks *T* and performance measure *P* if its performance at tasks in *T*, as measured by *P*, improves with experience *E*." This definition of the tasks in which machine learning is concerned offers a fundamentally operational definition rather than defining the field in cognitive terms. This follows Alan Turing's proposal in his paper "Computing Machinery and Intelligence", in which the question "Can machines think?" is replaced with the question "Can machines do what we (as thinking entities) can do?".

Modern day machine learning has two objectives, one is to classify data based on models which have been developed, the other purpose is to make predictions for future outcomes based on these models. A hypothetical algorithm specific to classifying data may use computer vision of moles coupled with supervised learning in order to train it to classify the cancerous moles. A machine learning algorithm for stock trading may inform the trader of future potential predictions.

Machine learning is a collection of algorithms that generate insight from data. That insight might be used by humans or other machines to make a decision. For example, a team of primary care researchers and computer scientists at Nottingham University used machine learning algorithms for cardiovascular r disease risk assessment. The machine learning algorithms were better than experienced medical doctors at gauging heart attack risk. Machine learning is used for two primary tasks. The first task, is forecasting future outcomes. For example, we might want to optimize the traffic light sequences at a busy intersection. A machine learning algorithm might be developed to predict the five minutes ahead traffic flow. The second task, is classification of object into specific classes. For example, coffee beans are usually classified into one of four grades (Specialty, Premium, Exchange, Standard). An industrial scale purchaser of beans, might develop a machine learning algorithm to automatically grade bean quality.

Machine learning approaches are traditionally divided into three broad categories, depending on the nature of the "signal" or "feedback" available to the learning system:

- Supervised learning: The computer is presented with example inputs and their desired outputs, given by a "teacher", and the goal is to learn a general rule that maps inputs to outputs.
- Unsupervised learning: No labels are given to the learning algorithm, leaving it on its own to find structure in its input. Unsupervised learning can be a goal in itself (discovering hidden patterns in data) or a means towards an end (feature learning).
- Reinforcement learning: A computer program interacts with a dynamic environment in which it must perform a certain goal (such as driving a vehicle or playing a game against an opponent). As it navigates its problem space, the program is provided feedback that's analogous to rewards, which it tries to maximize.

Supervised learning

A support-vector machine is a supervised learning model that divides the data into regions separated by a linear boundary. Here, the linear boundary divides the black circles from the white.

Supervised learning algorithms build a mathematical model of a set of data that contains both the inputs and the desired outputs. The data is known as training data, and consists of a set of training examples. Each training example has one or more inputs and the desired output, also known as a supervisory signal. In the mathematical model, each training example is represented by an array or vector, sometimes called a feature vector, and the training data is represented by a matrix. Through iterative optimization of an objective function, supervised learning algorithms learn a function that can be used to predict the output associated with new inputs. An optimal function will allow the algorithm to correctly determine the output for inputs that were not a part of the training data. An algorithm that improves the accuracy of its outputs or predictions over time is said to have learned to perform that task.

The most widely used learning algorithms are:

1.Support-vector machines

2.Linear regression

3.Logistic regression

4.Naive Bayes

5.Linear discriminant analysis

6.Decision trees

7.K-nearest neighbour algorithm

8.Neural networks (Multilayer perceptron)

9.Similarity learning

Unsupervised learning

Unsupervised learning algorithms take a set of data that contains only inputs, and find structure in the data, like grouping or clustering of data points. The algorithms, therefore, learn from test data that has not been labelled, classified or categorized. Instead of responding to feedback, unsupervised learning algorithms identify commonalities in the data and react based on the presence or absence of such commonalities in each new piece of data. A central application of unsupervised learning is in the field of density estimation in statistics, such as finding the probability density function. Though unsupervised learning encompasses other domains involving summarizing and explaining data features.

Cluster analysis is the assignment of a set of observations into subsets (called clusters) so that observations within the same cluster are similar according to one or more predesignated criteria, while observations drawn from different clusters are dissimilar. Different clustering techniques make different assumptions on the structure of the data, often defined by some similarity metric and evaluated, for example, by internal compactness, or the similarity between members of the same cluster, and separation, the difference between clusters. Other methods are based on estimated density and graph connectivity.

In my dissertation I have implemented three supervised Learning techniques:

Logistic Regression:

The logistic regression as a general statistical model was originally developed and popularized primarily by Joseph Berkson beginning in Berkson (1944), where he coined "logit". In statistics, the (binary) logistic model (or logit model) is a statistical model that models the probability of one event (out of two alternatives) taking place by having the log-odds (the logarithm of the odds) for the event be a linear combination of one or more independent variables ("predictors"). In regression analysis, logistic regression (or logit regression) is estimating the parameters of a logistic model (the coefficients in the linear combination). Formally, in binary logistic regression there is a single binary dependent variable, coded by a indicator variable, where the two values are labelled "0" and "1", while the independent variables can each be a binary variable (two classes, coded by an indicator variable) or a continuous variable (any real value). The corresponding probability of the value labelled "1" can vary between 0 (certainly the value "0") and 1 (certainly the value "1"), hence the labelling the function that converts log-odds to probability is the logistic function, hence the name. The unit of measurement for the log-odds scale is called a logit, from logistic unit, hence the alternative names.

Binary variables are widely used in statistics to model the probability of a certain class or event taking place, such as the probability of a team winning, of a patient being healthy, etc. and the logistic model has been the most commonly used model for binary regression since about 1970. Binary variables can be generalized to categorical variables when there are more than two possible values (e.g. whether an image is of a cat, dog, lion, etc.), and the binary logistic regression generalized to multinomial logistic regression. If the multiple categories are ordered, one can use the ordinal logistic regression (for example the proportional odds ordinal logistic model. The logistic regression model itself simply models probability of output in terms of input and does not perform statistical classification (it is not a classifier), though it can be used to make a classifier, for instance by choosing a cut-off value and classifying inputs with probability greater than the cut-off as one class, below the cut-off as the other; this is a common way to make a binary classifier. The parameters of a logistic regression are most commonly estimated by maximum-likelihood estimation (MLE). This does not have a closed-form expression, unlike linear least squares. Logistic regression by MLE plays a similarly basic role for binary or categorical responses as linear regression by ordinary least squares (OLS) plays for scalar responses: it is a simple, well-analysed baseline model.

K-Nearest Neighbours:

In statistics, the k-nearest neighbours algorithm (k-NN) is a non-parametric supervised learning method first developed by Evelyn Fix and Joseph Hodges in 1951, and later expanded by Thomas Cover. It is used for classification and regression. In both cases, the input consists of the k closest training examples in a data set. The output depends on whether k-NN is used for classification or regression:

In k-NN classification, the output is a class membership. An object is classified by a plurality vote of its neighbours, with the object being assigned to the class most common among its k nearest neighbours (k is a positive integer, typically small). If k = 1, then the object is simply assigned to the class of that single nearest neighbour. In k-NN regression, the output is the property value for the object. This value is the average of the values of k nearest neighbours. k-NN is a type of classification where the function is only approximated locally and all computation is deferred until function evaluation. Since this algorithm relies on distance for classification, if the features represent different physical units or come in vastly different scales then normalizing the training data can improve its accuracy dramatically. Both for classification and regression, a useful technique can be to assign weights to the contributions of the neighbours, so that the nearer neighbours contribute more to the average than the more distant ones. For example, a common weighting scheme consists in giving each neighbour a weight of 1/d, where d is the distance to the neighbour.

Naïve Bayes Classifier:

In statistics, naive Bayes classifiers are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong (naive) independence assumptions between the features (see Bayes classifier). They are among the simplest Bayesian network models, but coupled with kernel density estimation, they can achieve high accuracy levels.

Naive Bayes classifiers are highly scalable, requiring a number of parameters linear in the number of variables (features/predictors) in a learning problem. Maximum-likelihood training can be done by evaluating a closed-form expression, which takes linear time, rather than by expensive iterative approximation as used for many other types of classifiers.

In the statistics literature, naive Bayes models are known under a variety of names, including simple Bayes and independence Bayes. All these names reference the use of **Bayes' theorem** in the classifier's decision rule, but naive Bayes is not (necessarily) a Bayesian method.

Project goal: to predict if a customer will purchase an iphone or not given their gender , age and salary.

.

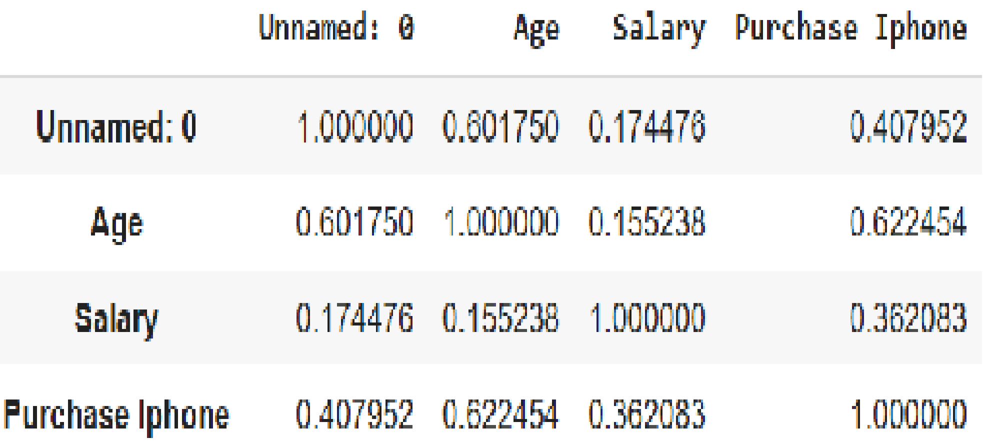
3. Data Description :

I have this information from*https://towardsdatascience.com* .It has been taken from a publicly available data source. Further , the attributes can be divided into two classes namely Numerical attributes and a categorical attribute .It has provided me a list of 400 unnamed customers but with useful and vital information with regards to their gender, age and salary. This dataset has proved to be very vital for my project as it has helped explore a lot regarding the goal i wanted to fulfil.

A description of my data set :

```
RangeIndex: 400 entries, 0 to 399
Data columns (total 5 columns):
 #   Column           Non-Null Count  Dtype 
---  ------           --------------  ----- 
 0   Unnamed: 0       400 non-null    int64 
 1   Gender           400 non-null    object
 2   Age              400 non-null    int64 
 3   Salary           400 non-null    int64 
 4   Purchase Iphone  400 non-null    int64 
dtypes: int64(4), object(1)
```

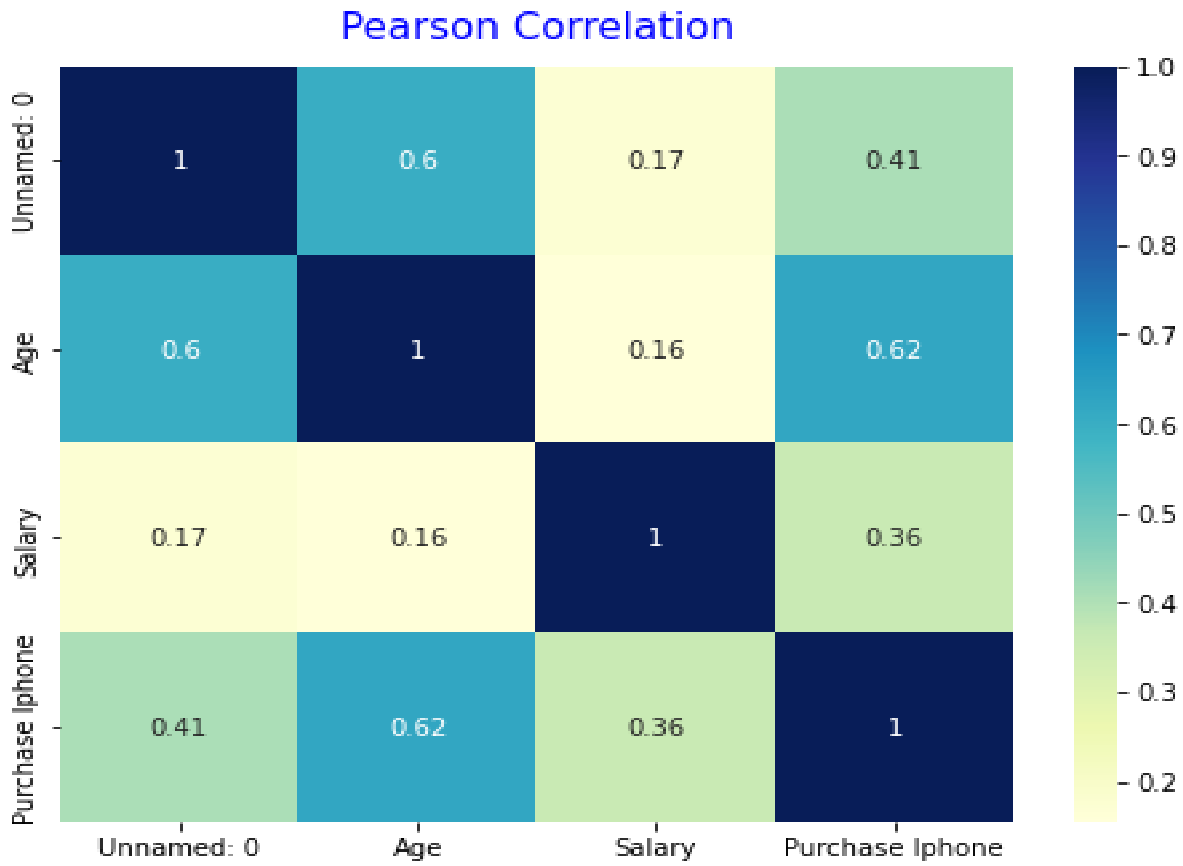
This is how they are correlated amongst themselves ;



	Unnamed: 0	Age	Salary	Purchase Iphone
Unnamed: 0	1.000000	0.601750	0.174476	0.407952
Age	0.601750	1.000000	0.155238	0.622454
Salary	0.174476	0.155238	1.000000	0.362083
Purchase Iphone	0.407952	0.622454	0.362083	1.000000

4.EXPLORATORY DATA ANALYSIS

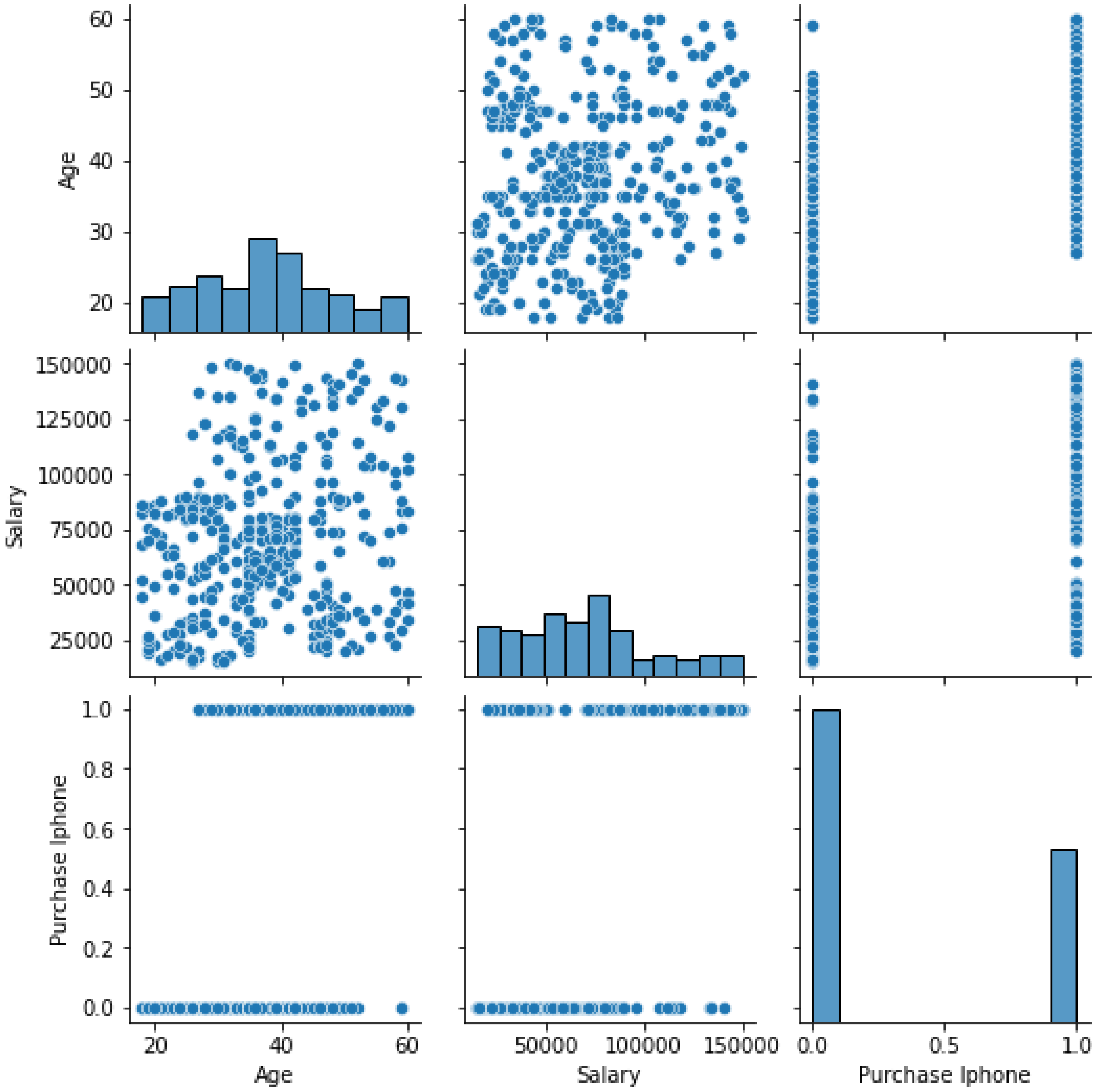
1. *Correlation Plot*



The above correlation plot tells us a lot about the features we have for analysis.

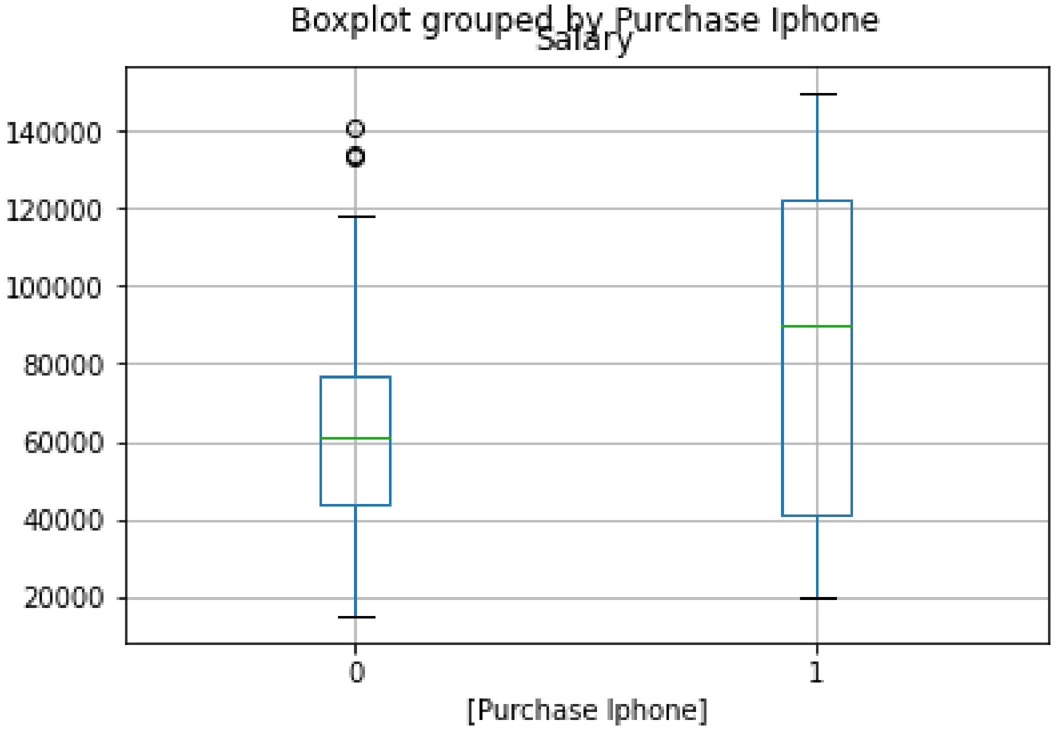
The analysis tells us that the variables will be successful in predicting which customer will the product.

2. *Pair Plots*

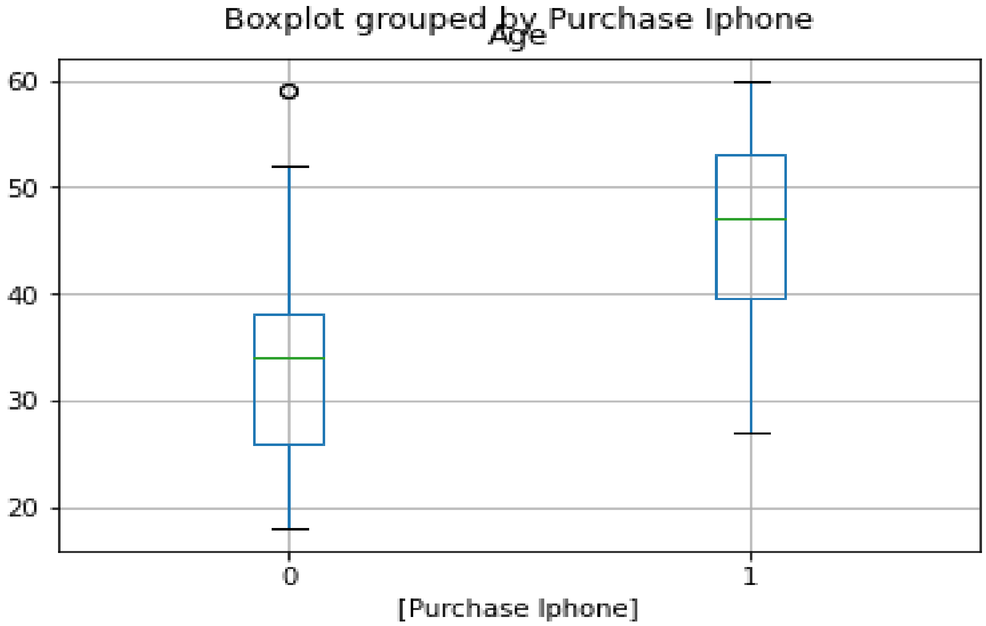


I have stored both the independent variables and the dependent variable in one variable "var" and I tried to obtain visual displays to elaborate on the understanding of all these attributes I am dealing with. Here, we show a complete visual analysis of all the four attributes in pairs to emphasize on how they are related to each other. Besides correlation this provides us good idea ready the way our dataset is structured.

3. BoxPlots



Boxplot of our dependant variable 'Purchase I-phone' against the independent variable 'Salary'.



Boxplot of our dependant variable 'Purchase I-phone' against the independent variable 'Age'.

5.METHODOLOGY

After performing EDA on the data-set we move directly towards the most vital part of our project (the analysis). Here, I am stating all the possible ways in which i performed my analysis .

Step 1: Load the dataset

Step 2: Convert Gender to Number (for converting the categorical variable into user friendly integer type variable)

Step 3: Split data into training and test set

Step 4: Feature Scaling (to specify which feature is more important)

Step 5: Check Accuracy of the Predictions (the procedure using which I am performing my analysis)

After the above steps I go directly into fitting my desired ML algorithms to obtain the results. My goal after obtaining the result is to obtaining the best fit ML algorithm and also the best fit measure of influence with regards to this dataset.

Step 6: Fitting the Logistic Regression

Step 7: Fitting the KNN classifier

Step 8: Fitting the Naïve Bayes Classifier

Now my analysis will be completed and the results will be displayed, after which we shall jump to the best possible conclusions.

6.Analysis :

Let’s look at the dataset. To make it interesting, let’s assume these are records of customers who 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.

Step 1: Load the dataset

X will contain all 3 independent variables “Gender”, “Age” and “Salary”

y will contain the dependent variable “Purchased iPhone”.

Step 2: Convert Gender to Number

For most machine learning algorithms, we have to convert categorical variables to numbers. Here we have the field “Gender” that has to be converted.

We will use the class Label Encoder to convert Gender to number.

In machine learning, we usually deal with datasets that contain multiple labels in one or more than one column. These labels can be in the form of words or numbers. To make the data understandable or in human-readable form, the training data is often labelled in words.

Label Encoding refers to converting the labels into a numeric form so as to convert them into the machine-readable form. Machine learning algorithms can then decide in a better way how those labels must be operated. It is an important pre-processing step for the structured dataset in supervised learning.

Step 3: Split data into training and test set

The train-test split procedure is used to estimate the performance of machine learning algorithms when they are used to make predictions on data not used to train the model. It is a fast and easy procedure to perform, the results of which allow you to compare the performance of machine learning algorithms for your predictive modelling problem. Although simple to use and interpret, there are times when the procedure should not be used, such as when you have a small dataset and situations where additional configuration is required, such as when it is used for classification and the dataset is not balanced**.**

Step 4: Feature Scaling

For Logistic Regression algorithm we have to do feature scaling. We will use Standard

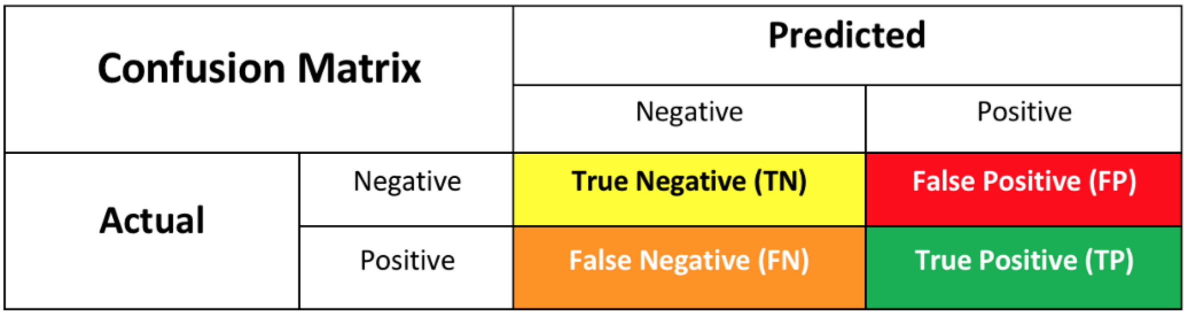
Scaler for this purpose.

Feature Scaling is a technique to standardize the independent features present in the data in a fixed range. It is performed during the data pre-processing to handle highly varying magnitudes or values or units. If feature scaling is not done, then a machine learning algorithm tends to weigh greater values, higher and consider smaller values as the lower values, regardless of the unit of the values.

Step 5: Check Accuracy of the Predictions

In classification problems, we can compare the predicted results with the actual results using the confusion matrix.

Confusion Matrix: It will tell us the number of correct and incorrect entries.



Confusion Matrix		Predicted	
		Negative	Positive
Actual	Negative	True Negative (TN)	False Positive (FP)
	Positive	False Negative (FN)	True Positive (TP)

Let’s spend a minute analysing this matrix. What does this tell us?

- If a person has not bought an iPhone and the predicted value also says they have not bought — it is True Negative (TN) i.e. Actual value is 0 and Predicted Value is also 0.
- If a person has not bought an iPhone but the predicted value says they did buy — it is False Positive (FP) i.e. Actual Value is 0 and Predicted Value is 1.
- If a person has bought an iPhone but the predicted value says they did not buy — it is False Negative (FN) i.e. Actual Value is 1 and Predicted Value is 0.
- If a person has bought an iPhone and the predicted value also says they bought — it is True Positive (TP) i.e. Actual value is 1 and Predicted Value is also 1.

Accuracy Score: This is the most common metric that is used for checking the accuracy of the model. It is the percentage of total number of correct predictions by total number of predictions.

Accuracy Score = (TP + TN) / (TP + TN + FP + FN)

Recall Score: It is the percentage of positive events that we predicted correctly.

Recall Score = TP / (TP + FN)

Precision Score: It is the percentage of predicted positive events that are actually positive.

Precision = TP / (TP + FP)

Error Rate: Error rate tells that over all how often the model is making a wrong prediction.

Classification error **=** (FP + FN)/ (TP + TN + FP + FN)

As clear from the above formulas,

Error = 1-Accuracy

I am going to perform the above predictions using three supervised machine learning classifiers **(Logistic regression , KNN (K-nearest neighbour classifier) and Naïve bayes classifier)** and compare their accuracy, recall and precision score accordingly in order the check which is the best model for our prediction.

Step 6: Fitting the Logistic Regression

We are going to use the Logistic Regression class from sklearn linear model library. When we create an object of this class, it takes many parameters. One of them is the "solver" which specifies which Algorithm to use in the optimization problem.

Logistic regression is similar to linear regression except that the target variable is a binary. For example, an Ethologist might assign a crab to one of two classes (say female or male). A male might be coded as y = 1, and a female coded by y = 0. Recall, in linear regression the target variable is related to the features via the linear relationship:

$$y = \alpha + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \varepsilon$$

. Suppose p(y) is the probability that a crab is male (we could write p(y = 1) but stick to the shorter notation). To relate p(y) to the features you might consider writing:

$$p(y) = \alpha + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3$$

. Unfortunately, this specification, can generate values for p(y) from -∞ to ∞. We need a model that generates probabilities in the 0 to 1 range. This is not guaranteed to be the case if we use the above equation Furthermore, linear regression assumes the values of y are normally distributed. In logistic regression y takes the values 0 or 1, so this assumption is clearly violated. We need a more appropriate transformation. This can be achieved using the logistic regression model:

$$\ln\left(\frac{p(y)}{1-p(y)}\right) = \alpha + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3$$

. It is a non-linear transformation of the linear regression model where the target variable is binary.

The predictions obtained with respect to logistic regression are :

```
[[65 3]
[ 6 26]]
Accuracy score: 0.91
Precision score: 0.896551724137931
Recall score: 0.8125
```

Error rate: 0.08999999999999997

We got an accuracy score of 91% which is pretty good.

Step 7 : Fitting the KNN classifier:

The k-Nearest Neighbours algorithm (KNN) is one of those machine learning algorithms that is very simple to understand and works incredibly well in practice. It is a non-parametric algorithm. Non-parametric simple means that the algorithm makes no assumptions about the probability distribution generating the sample data**.**

Measuring Closeness with Distance Metrics: KNN calculates the distance between the points that require classification and its neighbours. The predicted class is then determined by a majority vote. How can we measure the “closeness” of a neighbour? The distance between observations i and j can be measured in many ways. Euclidean distance is often used for continuous features. It is calculated as:

$$\mathcal{D}(x_i, x_j) = \sqrt{\sum_{m=1}^{n} (x_{im} - x_{jm})^2}$$

where n is the number of features. Other popular metrics include the Manhattan distance :

$$\mathcal{D}(x_i, x_j) = \sum_{m=1}^{n} |x_{im} - x_{jm}|;$$

The Minkowski metric:

$$\mathcal{D}(x_i, x_j) = \left(\sum_{m=1}^{n} (|x_{im} - x_{jm}|)^q \right)^{\frac{1}{q}};$$

And the Canberra metric:

$$\mathcal{D}(x_i, x_j) = \sum_{m=1}^{n} \frac{|x_{im} - x_{jm}|}{|x_{im}| + |x_{jm}|}$$

Both Euclidean distance and the Manhattan distance can be regarded as special cases of Minkowski distance. The Euclidean distance results from the selection q = 2, the Manhattan distance for the parameter value q = 1. The Canberra distance is a weighted version of the Manhattan distance.

The predictions obtained with respect to **KNN classifier** are :

```
[[65 4]
[ 3 29]]
Accuracy score: 0.93
Precision score: 0.8787878787878788
Recall score: 0.90625
```

```
Error rate: 0.06999999999999995
```

We got an accuracy score of 93% which is also pretty good.

<u>Step 8: Fitting the Naïve Bayes Classifier :</u>

The Naive Bayes Classifier is a supervised classifier. This simply means that the class labels are available for use in building the model. We walk through a detailed step by step example to help clarify this further shortly. However, before we get there, you should know that there are two broad types of supervised classification algorithms:

1. Classification using generative algorithms via the use of the Bayes rule;

2. Classification via discriminant analysis where the classification boundaries are directly learnt from data

A Bayes classifier is a generative classifier that can be used to solve classification tasks. To begin let us denote x = [x1, ...,xn] as the input vector of features/ attributes, and the K classes by C = {c1, c2, ..., cK}. A Bayes classifier is a decision rule to estimate the most probably class ci for an observation given the set of input features or attributes x. It makes good use of Bayes rule. Conceptually, Bayes rule provides a mechanism to go from the conditional probability of the evidence provided by the input features given the classes P(x|ci), to the conditional probability of the class given the evidence provided by the attributes P(ci |x). This is important because in practice, you will often know how frequently some particular evidence occurs, given a known class or outcome.

The predictions obtained with respect to **Naïve Bayes Classifier** are :

```
[[66 2]
[ 7 25]]
Accuracy score: 0.91
Precision score: 0.9259259259259259
Recall score: 0.78125
```

```
Error rate: 0.08999999999999997
```

We got an accuracy score of 91% which is also pretty good.

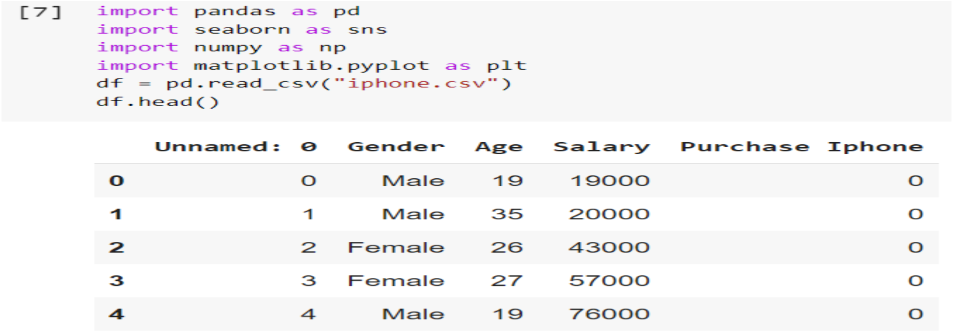
7. Result:

Accuracy, precision and recall are three metrics that can be built upon the confusion matrix. We can use accuracy when we are interested in predicting both 0 and 1 correctly and our dataset is balanced enough. We use precision when we want the prediction of 1 to be as correct as possible and we use recall when we want our model to spot as many real 1 as possible. Choosing the correct metric for our model can actually increase its predictive power and give us a great competitive advantage. As long as we have a clear idea of what our model must do, we can select one of them without doubts when we have to select a model or during Hyperparameter tuning phase.

Comparing the three ml algorithms I have implemented it is **conclusive that K nearest neighbour classifier is the one with the highest accuracy score about 0.93**. But here, if I prefer to choose to predict the 1s as correct as possible so I would prefer the precision score. **Precision score is highest for the naïve bayes classifier approx. 0.93 .** The only metric measure left is the recall score and I have obtained that the **K nearest neighbour classifier is the one with the highest recall score about 0.90625**.

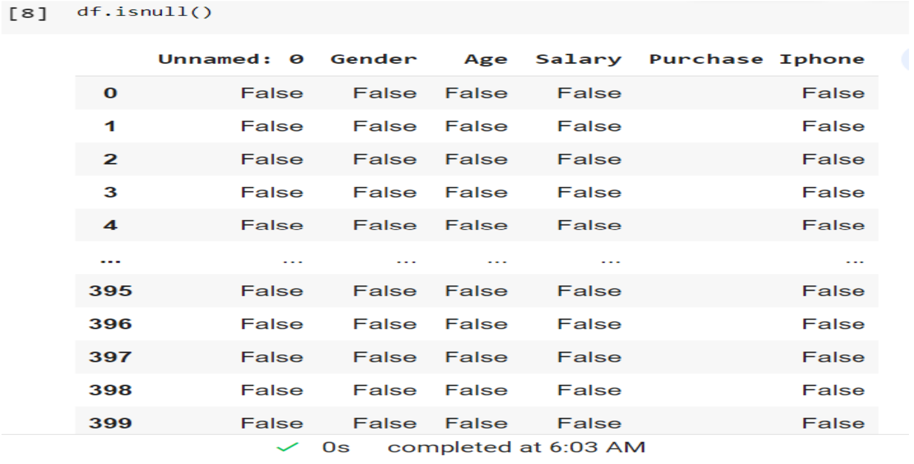
Thus, KNN seems to perform better in most of the metric measures as compared to the other ML algorithms I have used here. I conclude that the marketing firms should employ the KNN algorithms to solve problems on which customers they should approach when they are selling their most expensive products available in the market. They can simply gather basic information regarding age, sex and salary of their customers, run the KNN algorithm and check if the customer falls within their desired category, thus makes work efficient and productive as well.

8.APPENDIX:



```
[7]  import pandas as pd
     import seaborn as sns
     import numpy as np
     import matplotlib.pyplot as plt
     df = pd.read_csv("iphone.csv")
     df.head()
```

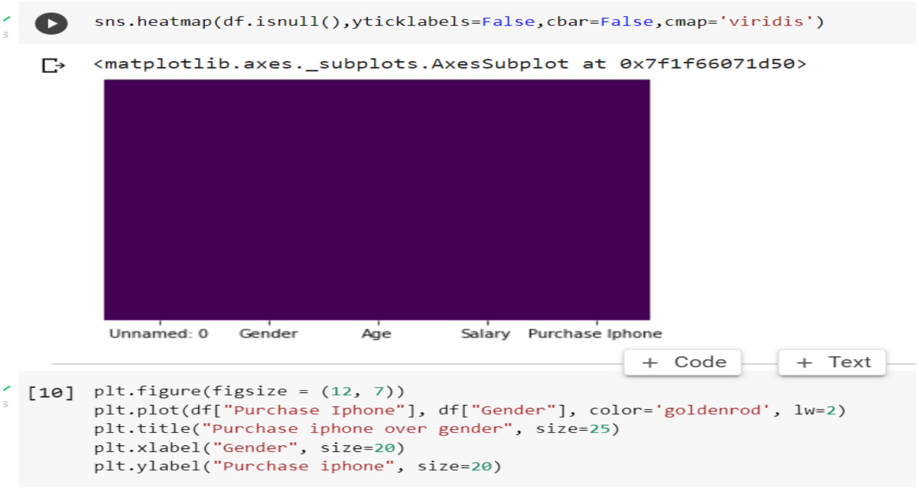
```
    Unnamed: 0  Gender  Age  Salary  Purchase Iphone
0            0    Male   19   19000                0
1            1    Male   35   20000                0
2            2  Female   26   43000                0
3            3  Female   27   57000                0
4            4    Male   19   76000                0
```



```
[8]  df.isnull()
```

```
     Unnamed: 0  Gender    Age  Salary  Purchase Iphone
0         False   False  False   False            False
1         False   False  False   False            False
2         False   False  False   False            False
3         False   False  False   False            False
4         False   False  False   False            False
...         ...     ...    ...     ...              ...
395       False   False  False   False            False
396       False   False  False   False            False
397       False   False  False   False            False
398       False   False  False   False            False
399       False   False  False   False            False
```

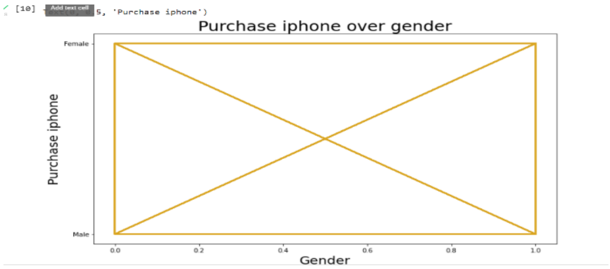
✓ 0s completed at 6:03 AM



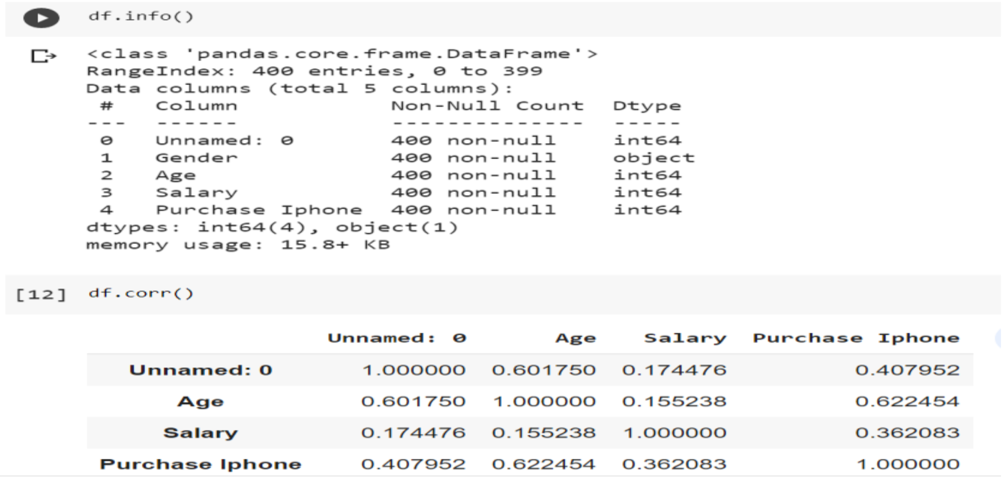
```
sns.heatmap(df.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1f66071d50>
```

```
[10]  plt.figure(figsize = (12, 7))
      plt.plot(df["Purchase Iphone"], df["Gender"], color='goldenrod', lw=2)
      plt.title("Purchase iphone over gender", size=25)
      plt.xlabel("Gender", size=20)
      plt.ylabel("Purchase iphone", size=20)
```



```
[10] 5, 'Purchase iphone')
```

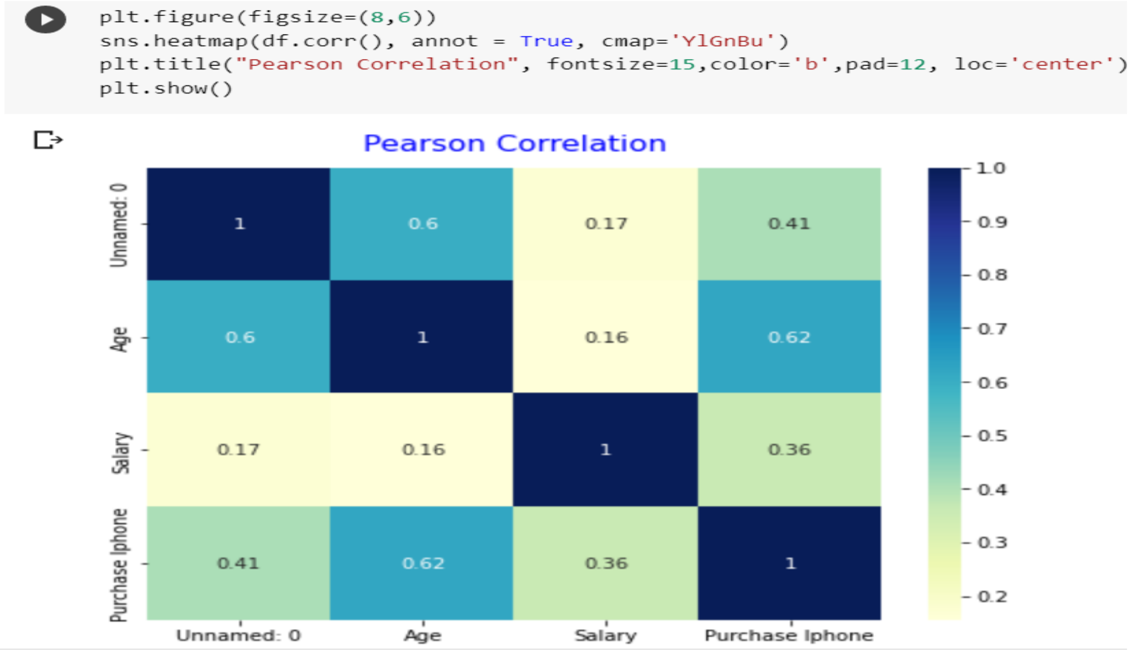


```
df.info()
```

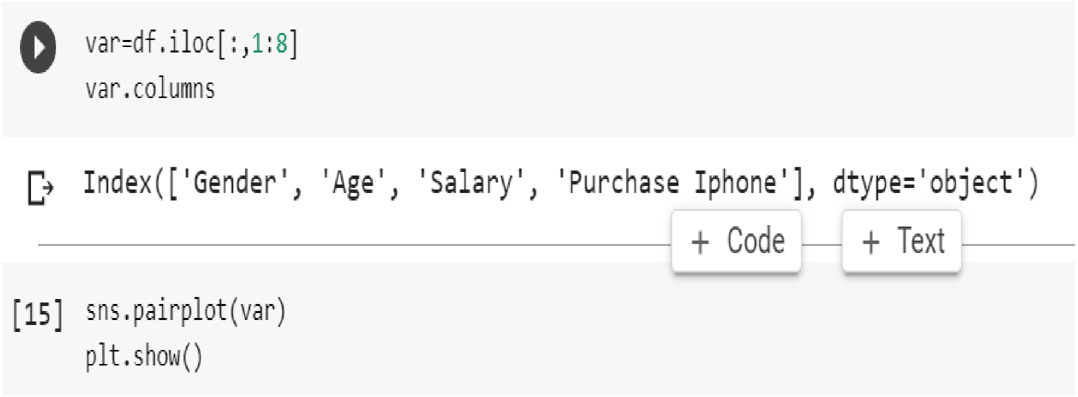
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 5 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Unnamed: 0       400 non-null    int64
 1   Gender           400 non-null    object
 2   Age              400 non-null    int64
 3   Salary           400 non-null    int64
 4   Purchase Iphone  400 non-null    int64
dtypes: int64(4), object(1)
memory usage: 15.8+ KB
```

```
[12] df.corr()
```

	Unnamed: 0	Age	Salary	Purchase Iphone
Unnamed: 0	1.000000	0.601750	0.174476	0.407952
Age	0.601750	1.000000	0.155238	0.622454
Salary	0.174476	0.155238	1.000000	0.362083
Purchase Iphone	0.407952	0.622454	0.362083	1.000000



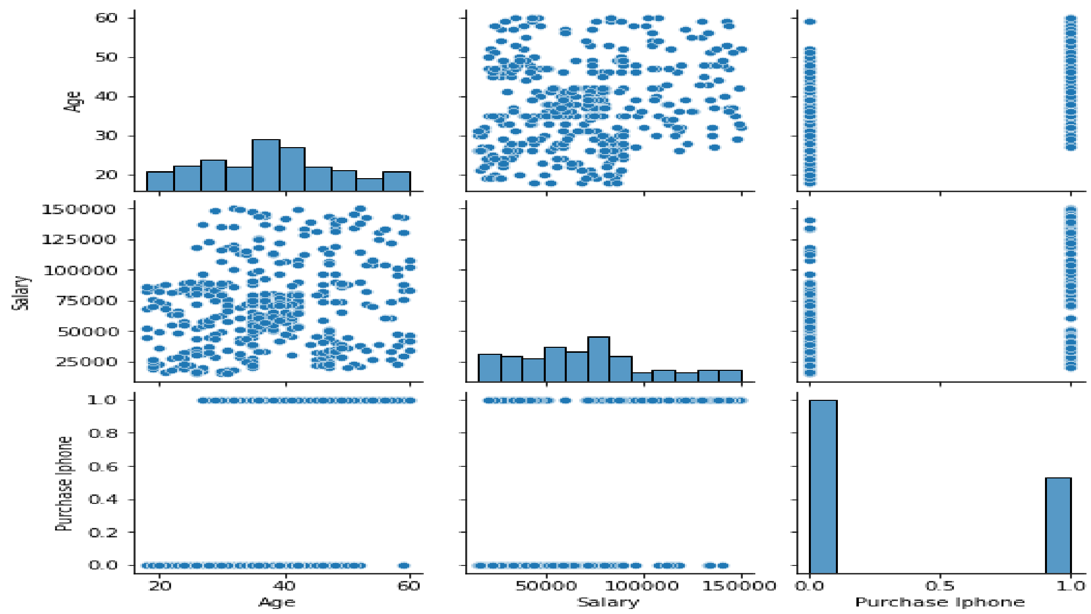
```
plt.figure(figsize=(8,6))
sns.heatmap(df.corr(), annot = True, cmap='YlGnBu')
plt.title("Pearson Correlation", fontsize=15,color='b',pad=12, loc='center')
plt.show()
```

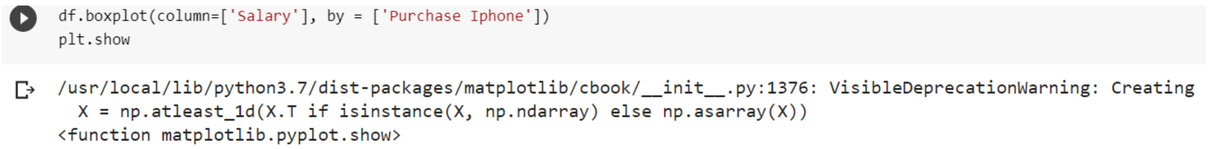


```
var=df.iloc[:,1:8]
var.columns
```

```
Index(['Gender', 'Age', 'Salary', 'Purchase Iphone'], dtype='object')
```

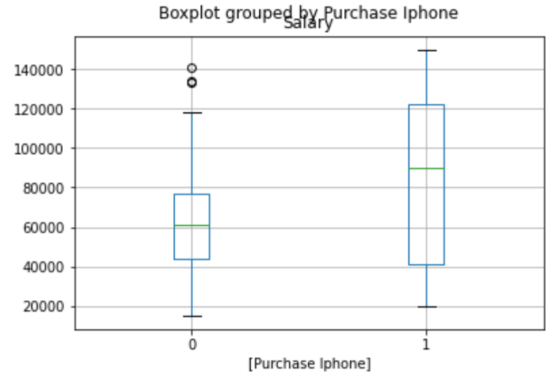
```
[15] sns.pairplot(var)
     plt.show()
```

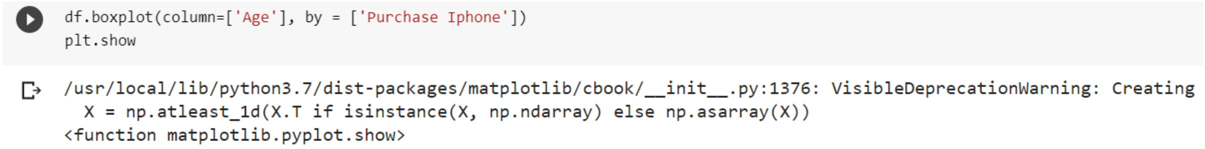




```
df.boxplot(column=['Salary'], by = ['Purchase Iphone'])
plt.show
```

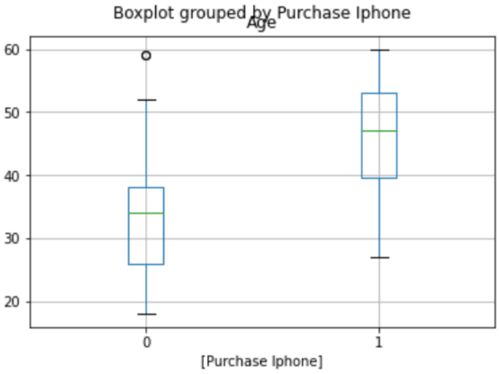
```
/usr/local/lib/python3.7/dist-packages/matplotlib/cbook/__init__.py:1376: VisibleDeprecationWarning: Creating
  X = np.atleast_1d(X.T if isinstance(X, np.ndarray) else np.asarray(X))
<function matplotlib.pyplot.show>
```

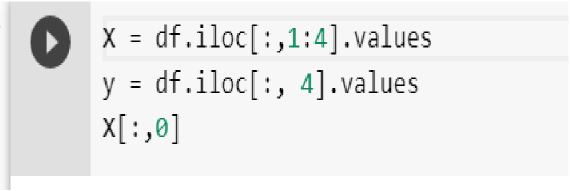




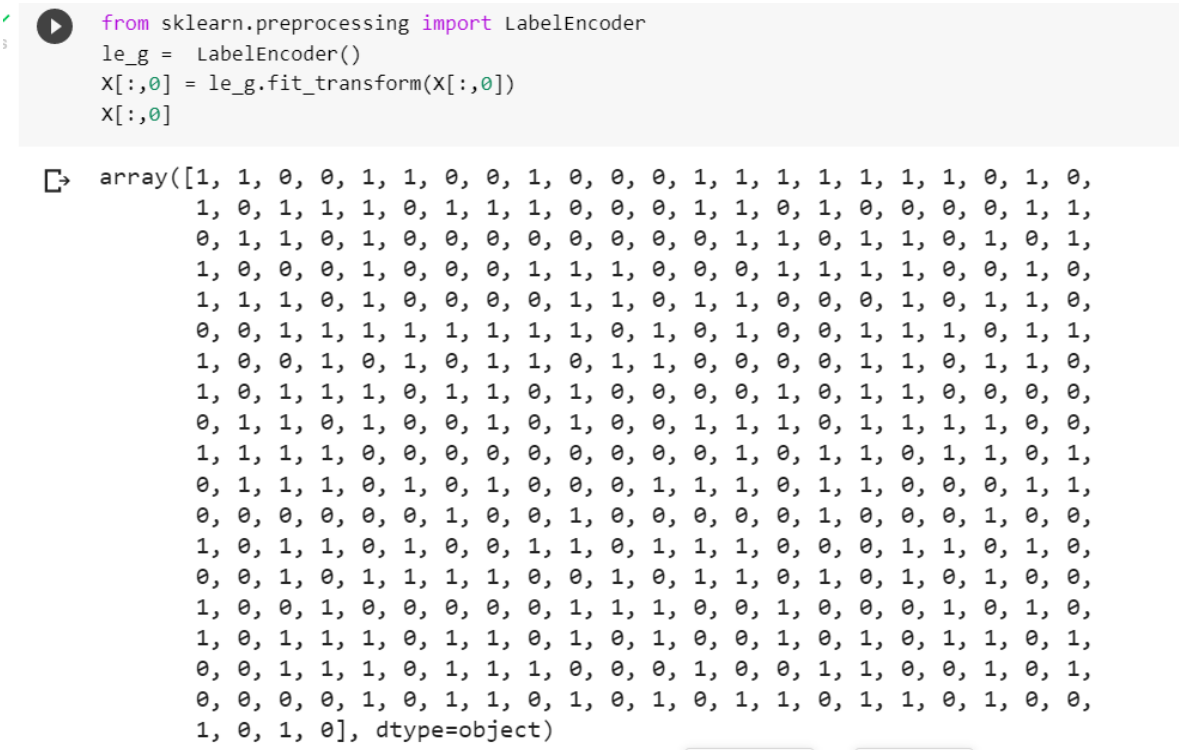
```
df.boxplot(column=['Age'], by = ['Purchase Iphone'])
plt.show
```

```
/usr/local/lib/python3.7/dist-packages/matplotlib/cbook/__init__.py:1376: VisibleDeprecationWarning: Creating
  X = np.atleast_1d(X.T if isinstance(X, np.ndarray) else np.asarray(X))
<function matplotlib.pyplot.show>
```



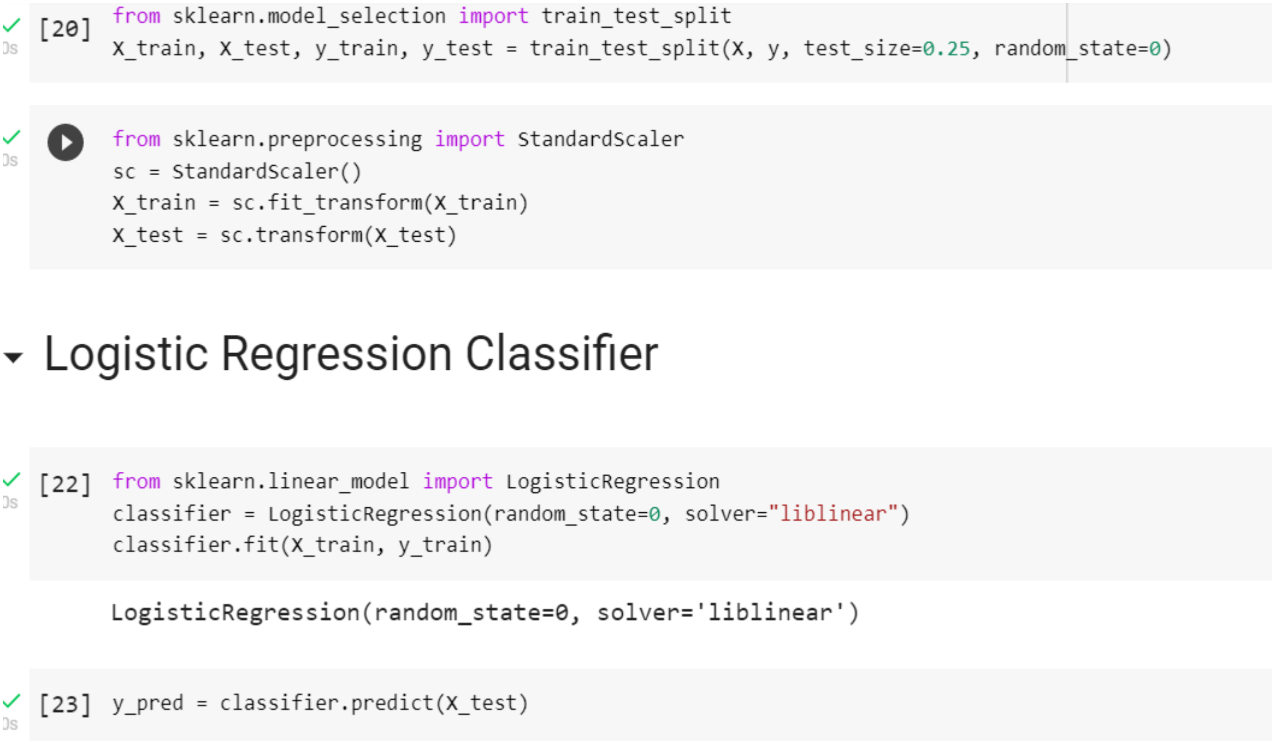


```
X = df.iloc[:,1:4].values
y = df.iloc[:, 4].values
X[:,0]
```



```
from sklearn.preprocessing import LabelEncoder
le_g =  LabelEncoder()
X[:,0] = le_g.fit_transform(X[:,0])
X[:,0]
```

```
array([1, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0,
       1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1,
       0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 0, 1,
       1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0,
       1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0,
       0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1,
       1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0,
       1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0,
       0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 0,
       1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1,
       0, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 1, 1,
       0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0,
       1, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0,
       0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0,
       1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0,
       1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1,
       0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1,
       0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0, 0,
       1, 0, 1, 0], dtype=object)
```



```
[20] from sklearn.model_selection import train_test_split
     X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_state=0)
```

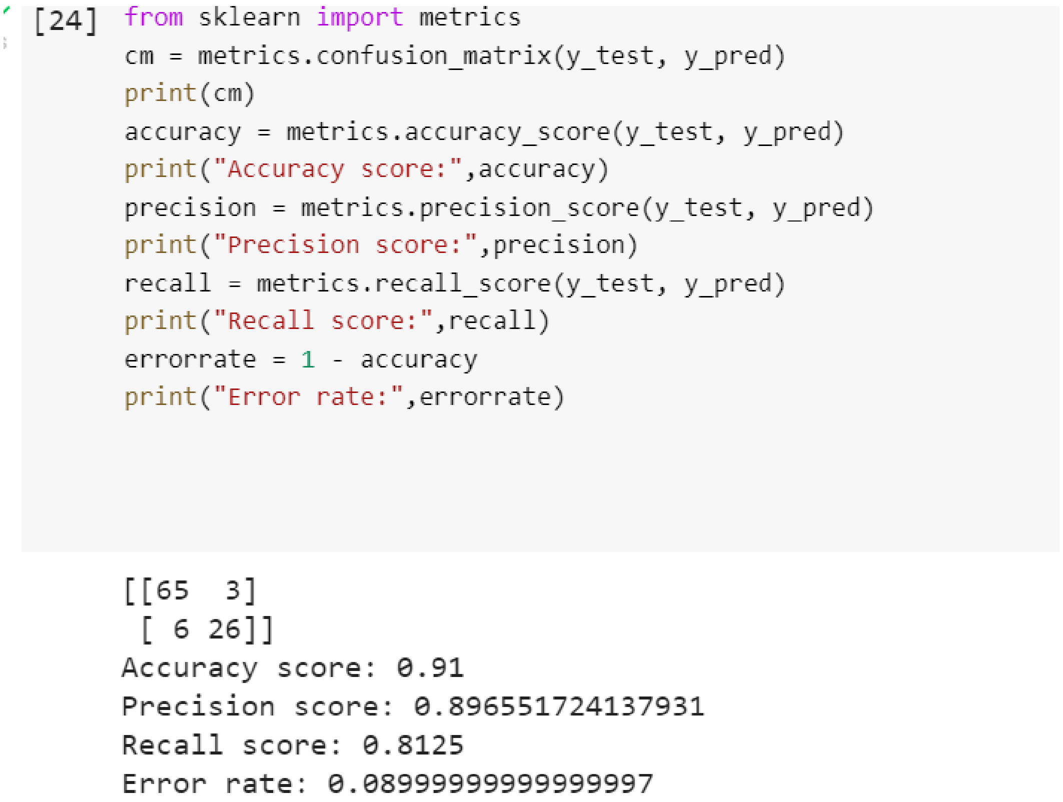
```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

Logistic Regression Classifier

```
[22] from sklearn.linear_model import LogisticRegression
     classifier = LogisticRegression(random_state=0, solver="liblinear")
     classifier.fit(X_train, y_train)
```

```
LogisticRegression(random_state=0, solver='liblinear')
```

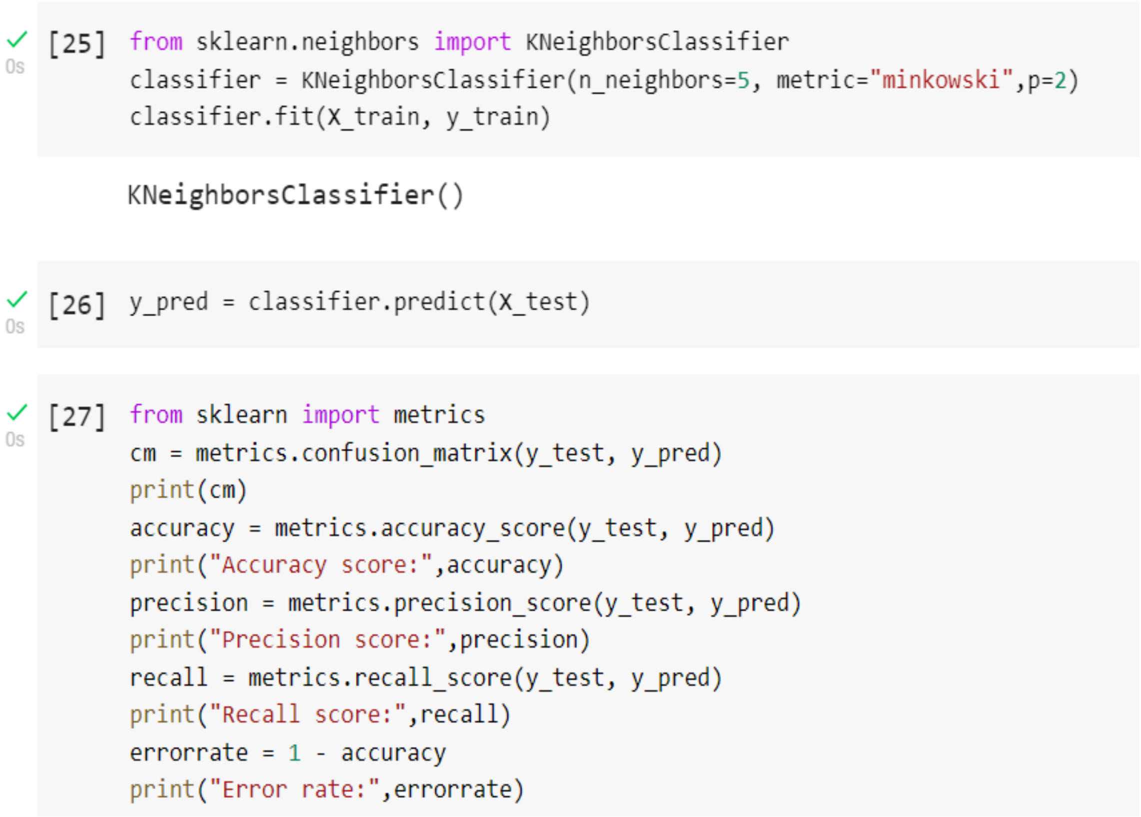
```
[23] y_pred = classifier.predict(X_test)
```



```
[24] from sklearn import metrics
     cm = metrics.confusion_matrix(y_test, y_pred)
     print(cm)
     accuracy = metrics.accuracy_score(y_test, y_pred)
     print("Accuracy score:",accuracy)
     precision = metrics.precision_score(y_test, y_pred)
     print("Precision score:",precision)
     recall = metrics.recall_score(y_test, y_pred)
     print("Recall score:",recall)
     errorrate = 1 - accuracy
     print("Error rate:",errorrate)
```

```
[[65  3]
 [ 6 26]]
Accuracy score: 0.91
Precision score: 0.896551724137931
Recall score: 0.8125
Error rate: 0.08999999999999997
```

K nearest neighbour Classifier (KNN)

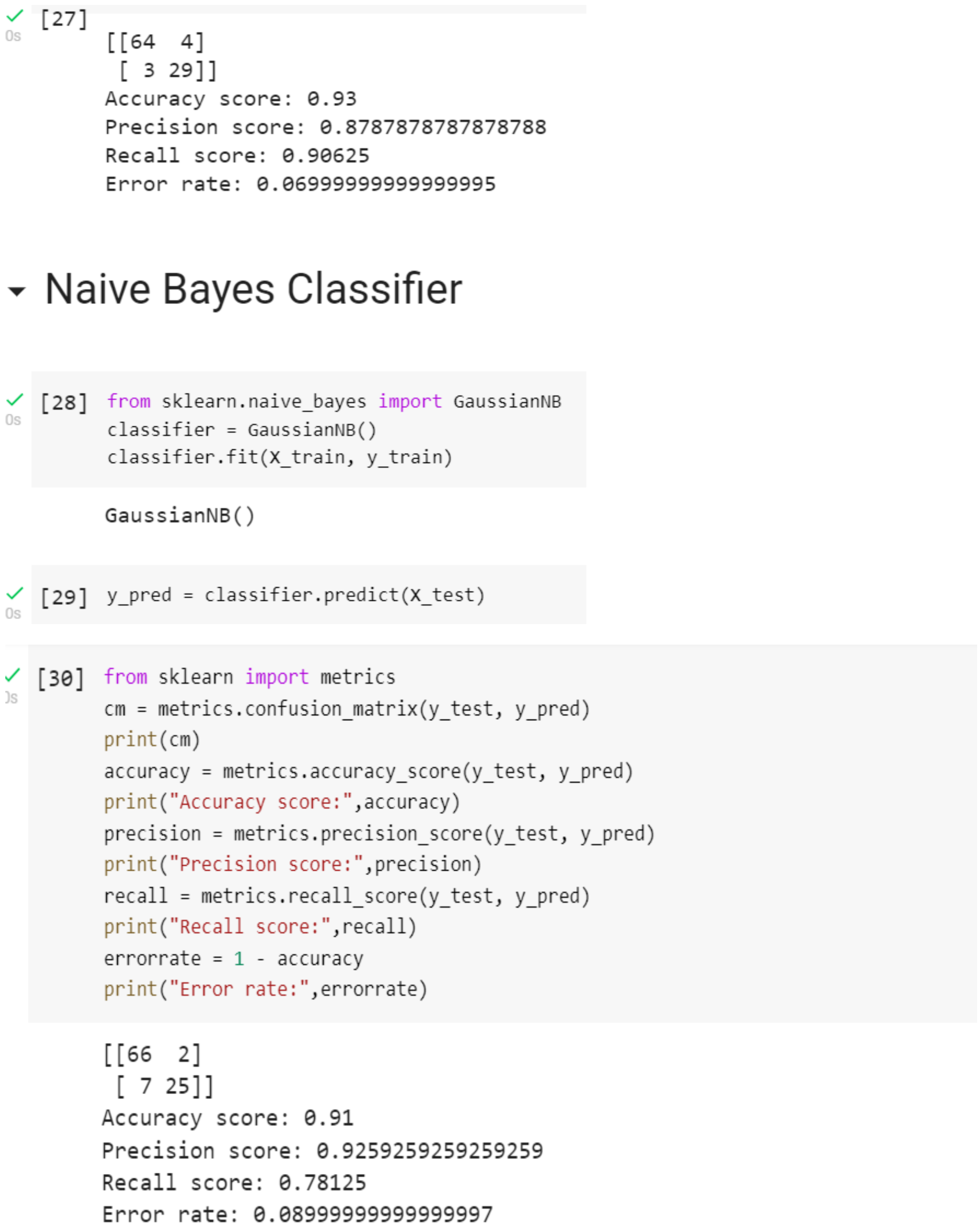


```
[25] from sklearn.neighbors import KNeighborsClassifier
     classifier = KNeighborsClassifier(n_neighbors=5, metric="minkowski",p=2)
     classifier.fit(X_train, y_train)
```

```
KNeighborsClassifier()
```

```
[26] y_pred = classifier.predict(X_test)
```

```
[27] from sklearn import metrics
     cm = metrics.confusion_matrix(y_test, y_pred)
     print(cm)
     accuracy = metrics.accuracy_score(y_test, y_pred)
     print("Accuracy score:",accuracy)
     precision = metrics.precision_score(y_test, y_pred)
     print("Precision score:",precision)
     recall = metrics.recall_score(y_test, y_pred)
     print("Recall score:",recall)
     errorrate = 1 - accuracy
     print("Error rate:",errorrate)
```



```
[27]
        [[64  4]
         [ 3 29]]
        Accuracy score: 0.93
        Precision score: 0.8787878787878788
        Recall score: 0.90625
        Error rate: 0.06999999999999995
```

Naive Bayes Classifier

```
[28]  from sklearn.naive_bayes import GaussianNB
      classifier = GaussianNB()
      classifier.fit(X_train, y_train)

      GaussianNB()
```

```
[29]  y_pred = classifier.predict(X_test)
```

```
[30]  from sklearn import metrics
      cm = metrics.confusion_matrix(y_test, y_pred)
      print(cm)
      accuracy = metrics.accuracy_score(y_test, y_pred)
      print("Accuracy score:",accuracy)
      precision = metrics.precision_score(y_test, y_pred)
      print("Precision score:",precision)
      recall = metrics.recall_score(y_test, y_pred)
      print("Recall score:",recall)
      errorrate = 1 - accuracy
      print("Error rate:",errorrate)

      [[66  2]
       [ 7 25]]
      Accuracy score: 0.91
      Precision score: 0.9259259259259259
      Recall score: 0.78125
      Error rate: 0.08999999999999997
```

9. References

1. *MACHINE LEARNING MADE EASY With R An Intuitive Step by Step Blueprint for Beginners Dr. N.D Lewis.*
2. *Research Methodology (Research and techniques) by C.R. Kothari*
3. *Bruce, P., Bruce, A., & Gedeck, P. (2020). Practical Statistics for Data Scientists: 50+ Essential Concepts Using R and Python (2nd ed.). O'Reilly Media.*
4. *James, G., Witten, D., Hastie, T., & Tibshirani, R. (2021). An Introduction to Statistical Learning: with Applications in R (Springer Texts in Statistics) (2nd ed. 2021 ed.). Springer.*
5. *Hastie, T., Tibshirani, R., & Friedman, J. (2022). The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition (Springer Series in Statistics) (2nd ed.). Springer.*
6. *Shah, K., et al., A comparative analysis of logistic regression, random forest and KNN models for the text classification. Augmented Human Research, 2020.* ***5****(1): p. 1-16.*
7. *Guo, G., et al. KNN model-based approach in classification. in OTM Confederated International Conferences" On the Move to Meaningful Internet Systems". 2003. Springer.*
8. *https://towardsdatascience.com*
9. *https://www.yourdatateacher.com*
10. *https://www.analyticsvidhya.com*