Capstone Project

Title: Predict the relation between the attributes and finding out which all clients is buying the cars (SUVs).

Abstract

The Dataset contains social network information about his clients. In which one of his client made an advertisement about sale of car (SUVs). The goal of this project is to analyze the given data and predict the correlation between the attributes and find out which of the clients is buying the car (SUVs). The data was divided into two parts i.e. the training and the test dataset. Models were developed based on the training dataset and applied to the test dataset to find out the accuracy of each model based on the predicted values generated. Based on these values we can determine how good a model is.

Submitted by:

Sunny Patel

Table of Contents

Capstone Project

Predict the relation between the attributes and finding out which all clients is buying the cars (SUVs).

A. Introduction

Using the fictional dataset of Gender, Age, Salary, Purchased (Target variable), the company wants to know whether a customer will buy its product or not.

```
> summary(data)
    User.ID          Gender       Age        EstimatedSalary    Purchased
 Min.   :15566689   Female:204   Min.   :18.00   Min.   : 15000   Min.   :0.0000
 1st Qu.:15626764   Male  :196   1st Qu.:29.75   1st Qu.: 43000   1st Qu.:0.0000
 Median :15694342                Median :37.00   Median : 70000   Median :0.0000
 Mean   :15691540                Mean   :37.66   Mean   : 69743   Mean   :0.3575
 3rd Qu.:15750363                3rd Qu.:46.00   3rd Qu.: 88000   3rd Qu.:1.0000
 Max.   :15815236                Max.   :60.00   Max.   :150000   Max.   :1.0000
```

Summary of the Data/Review of Literature

- From summary, we can see clients User.IDs . Each clients has different IDs .
- We can see that there are total 400 clients of which 204 are females and 196 are male clients.
- The Age of the client ranges from 18 yrs. to maximum age of 60 years.
- It also shows us the estimated salary of the clients that is 15000 to maximum 150000 as per their Age given.
- In the data the most important is the purchased column which means if the client has purchased it will show 1 and if not then it will show 0 .Because of this it's easy to find who all clients has purchased the cars (SUVs).

B. Transforming Data

- Missing Data

  The missing data must be treated to ensure accurate analysis.
  But as we saw on the previous page in the summary of the dataset there is no missing data.

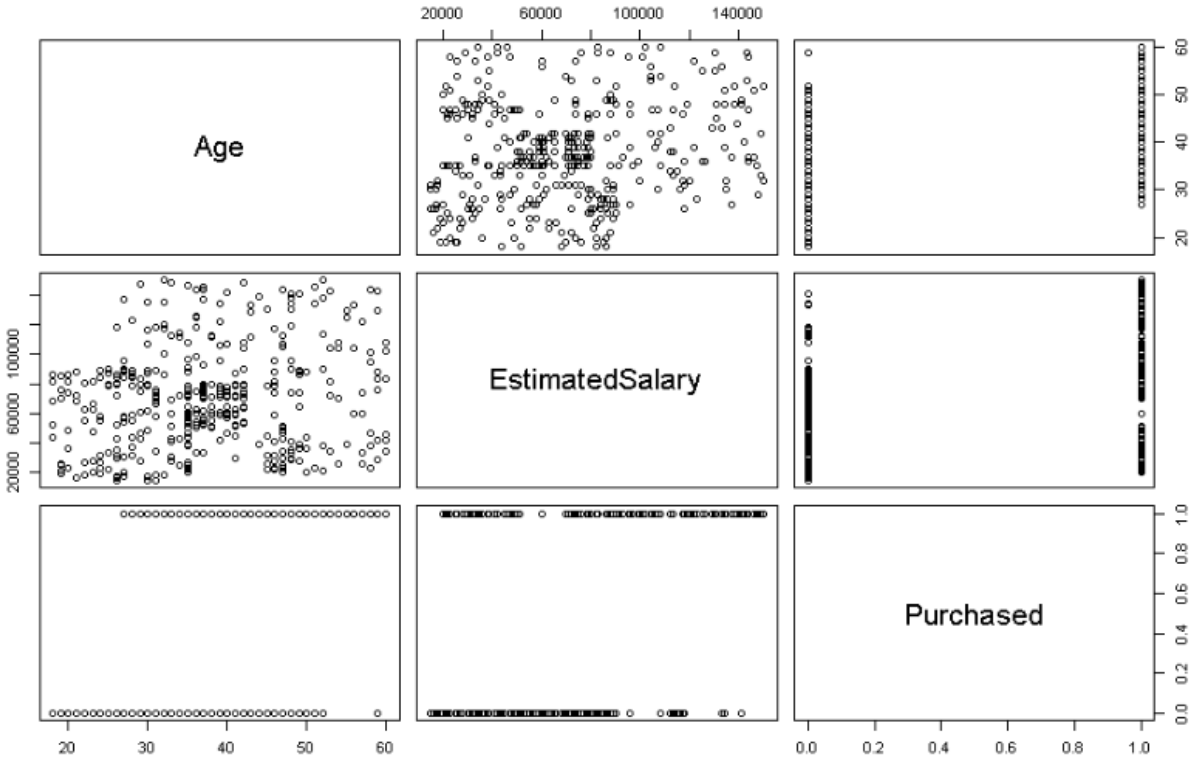
- Encoding categorical data

  The purchased column is already been categorized and named by 0,1 respectively so no need to categorize the dataset.

C. Data Visualization

1. ScatterPlot matrices

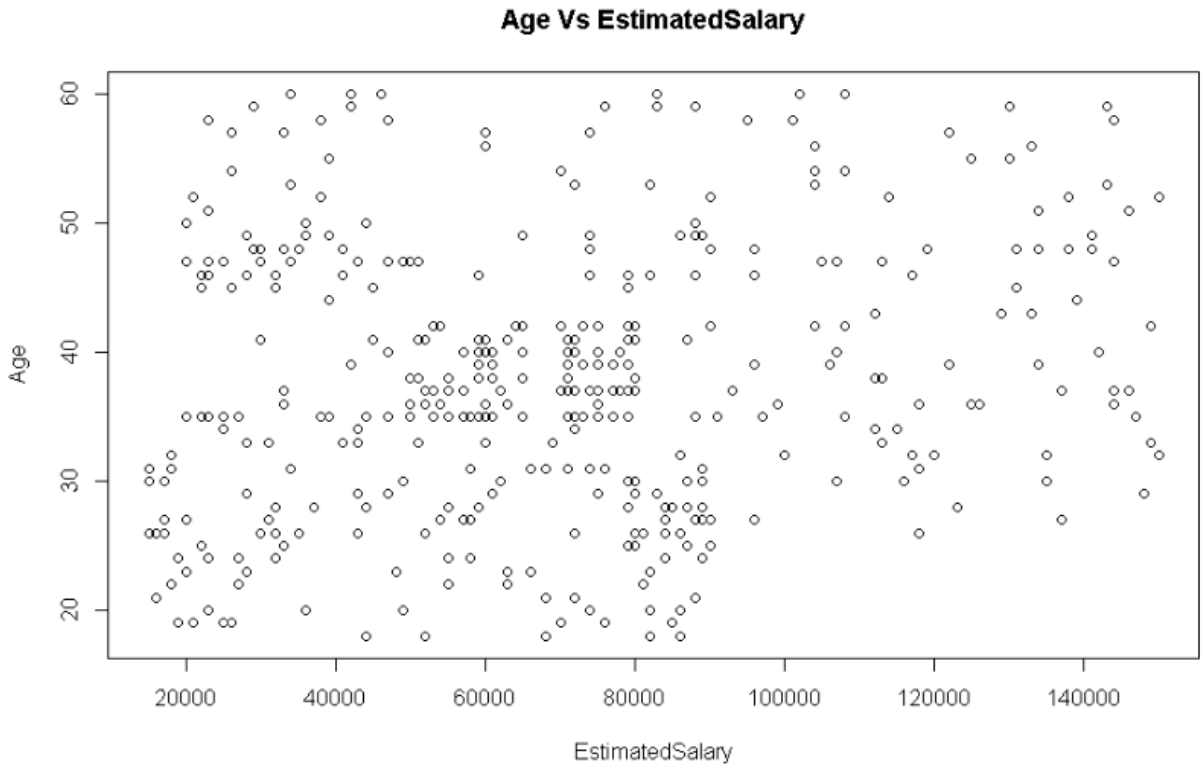
```
> #Scatterplot Matrices
> Scatterplot <- pairs(dataset)
```



From the above Plot you can see, the correlation ship between the dependent aka. target variable (Purchased) and the independent variables (Age and Estimated Salary).

2. ScatterPlot

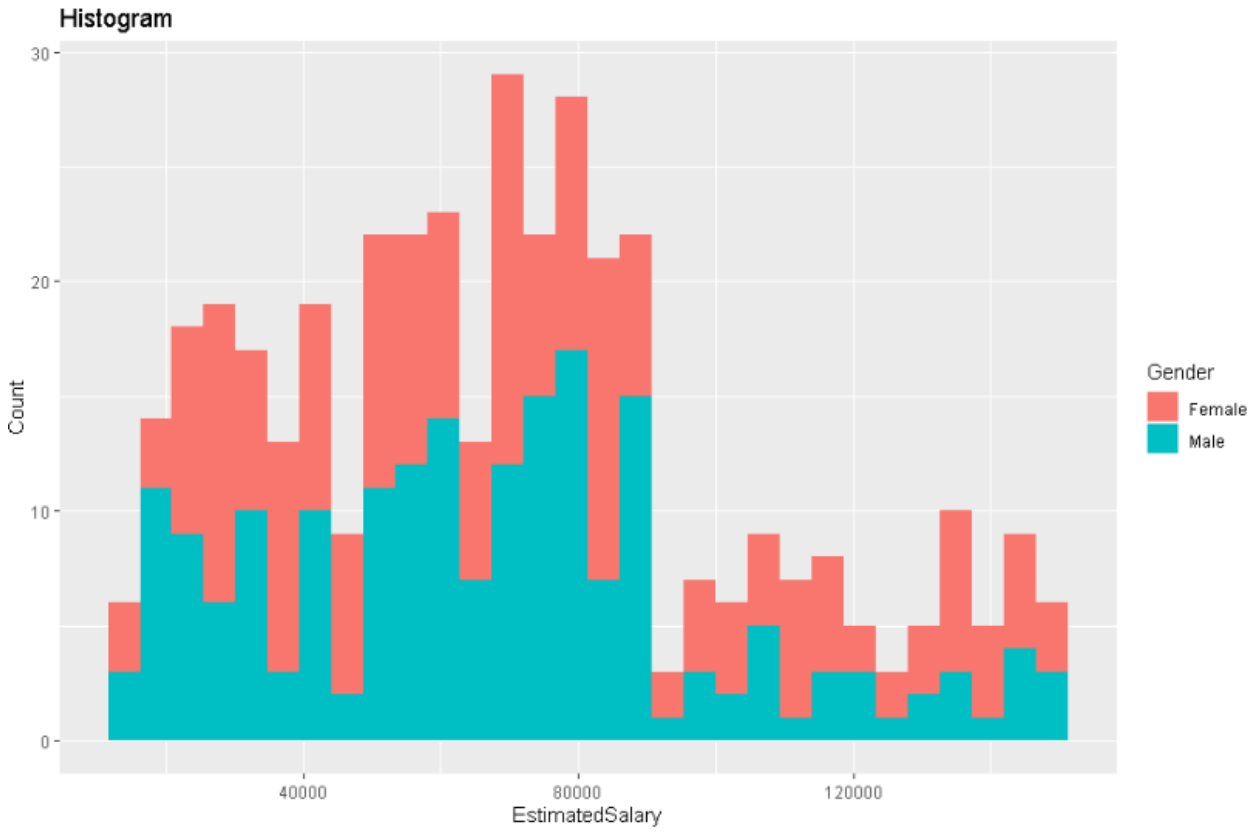
```
> #ScatterPlot
> plot(x =data$EstimatedSalary,y =data$Age,main ="Age Vs EstimatedSalary",xlab ="EstimatedSalary",ylab
 ="Age")
```



The above scatterplot is been drawn using plot() function. This graph shows us the relationship between the Age and EstimatedSalary of the client. The graph is very much scattered and vague, this shows us that the Estimated Salary is not dependent on Age. Because even the clients age is minimum let's say 20-25 then also, they are getting paid high salary.

3. Histogram

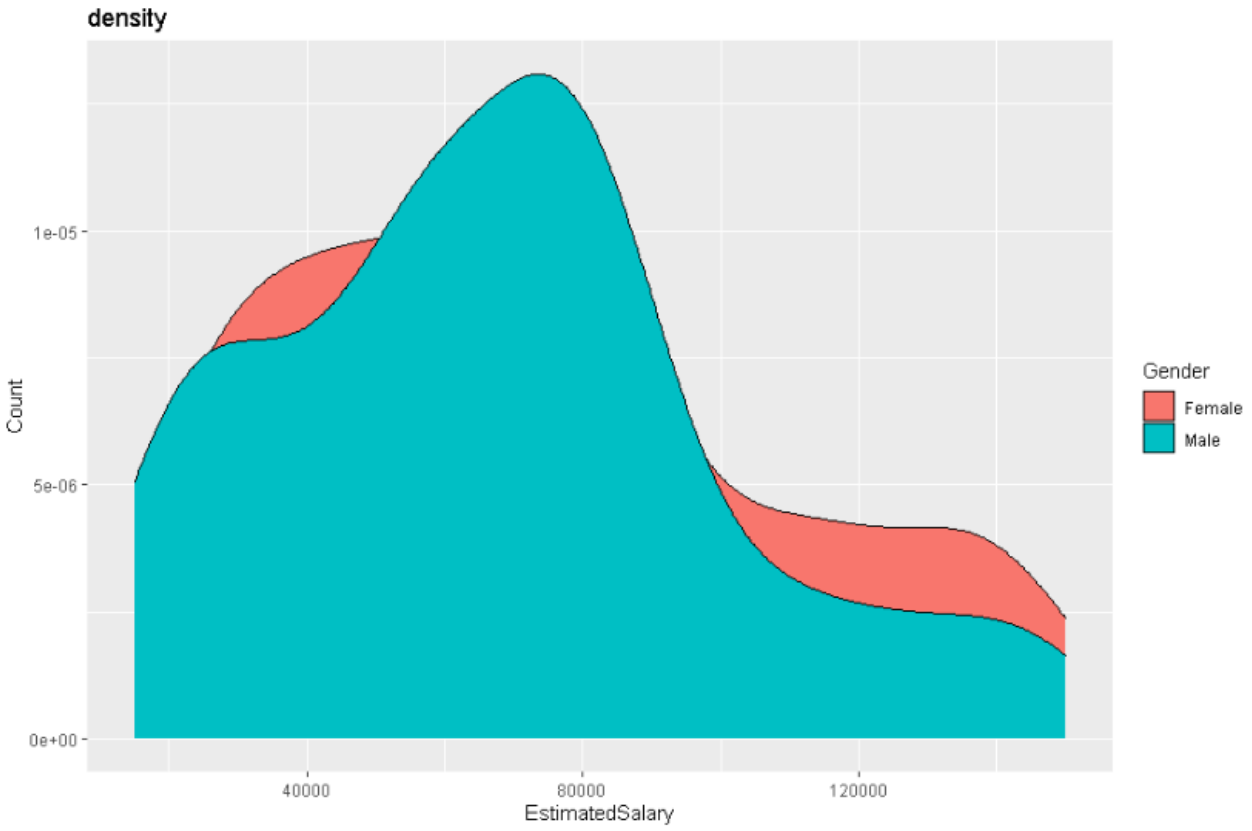
```
> s <-ggplot(data=data,aes(x=EstimatedSalary))
> s + geom_histogram(aes(fill=Gender))+
+       ggtitle("Histogram")+
+       xlab("EstimatedSalary")+
+       ylab("Count")
```



The age distribution of the genders is plotted .We see that this distribution is fairly normal. However, we notice that there are significantly a greater number of female clients as compared to the male clients.

4. Density

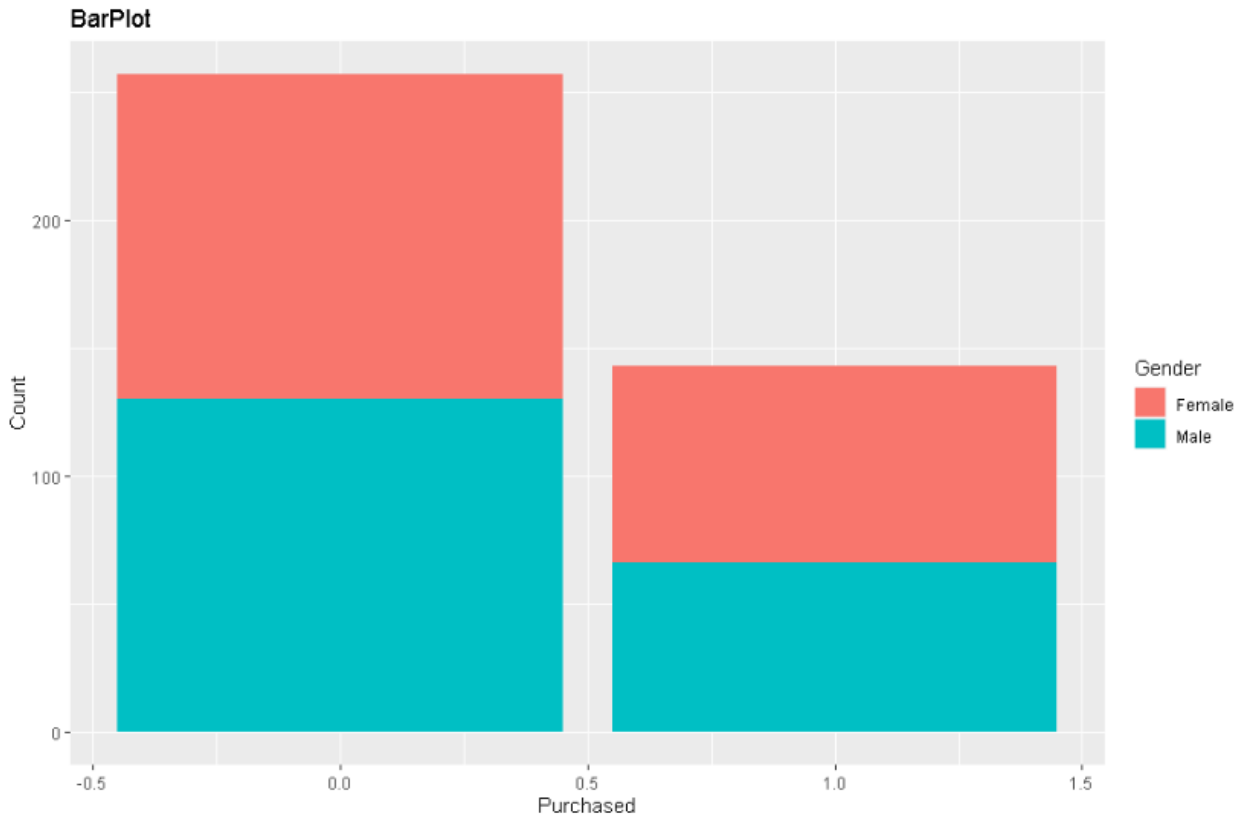
```
> s <-ggplot(data=data,aes(x=EstimatedSalary))
> s + geom_density(aes(fill=Gender))+
+   ggtitle("density")+
+   xlab("EstimatedSalary")+
+   ylab("Count")
```



By plotting the line graphs of the same we can make out that the distributions are very similar i.e. both the males and the females constituted a major Salary share between 150000 to 100000.

5. BarPlot

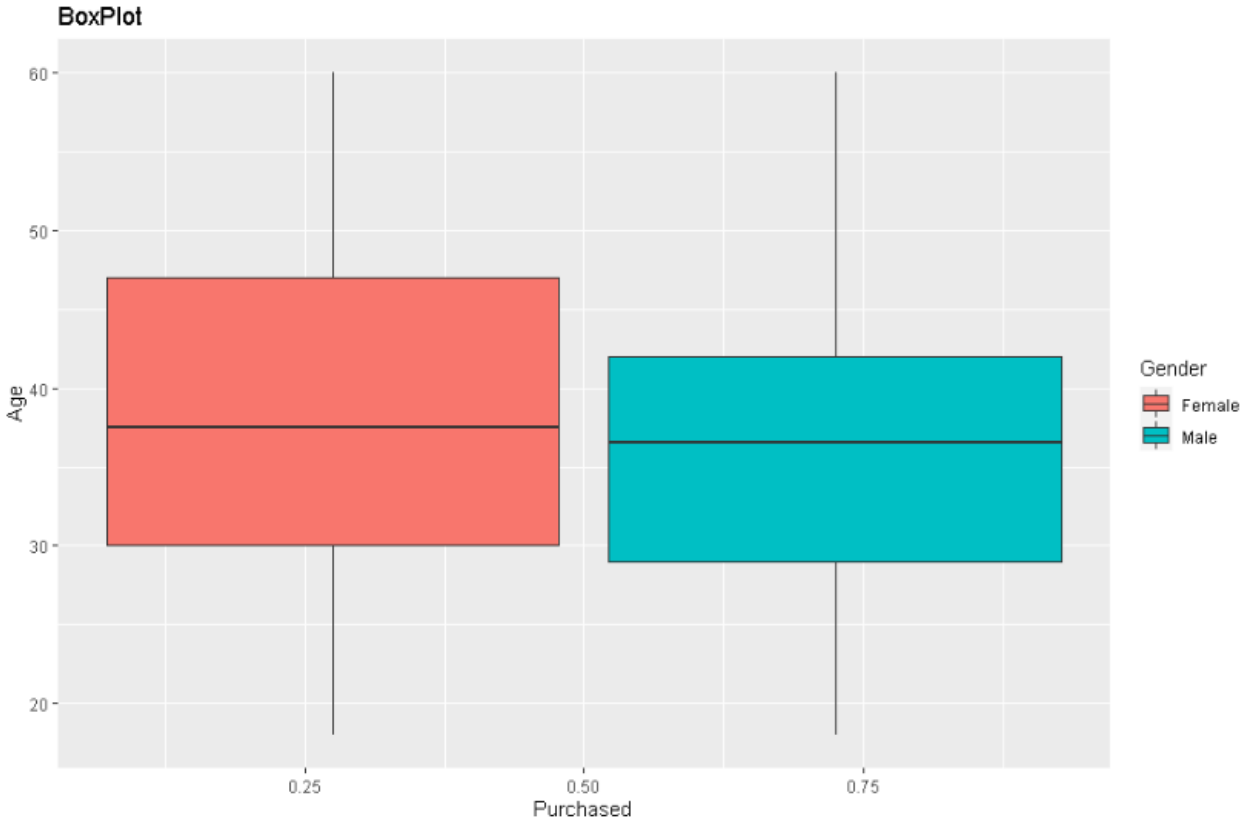
```
> s <-ggplot(data=data,aes(x=Purchased))
> s + geom_bar(aes(fill=Gender))+
+   ggtitle("BarPlot")+
+   xlab("Purchased")+
+   ylab("Count")
```



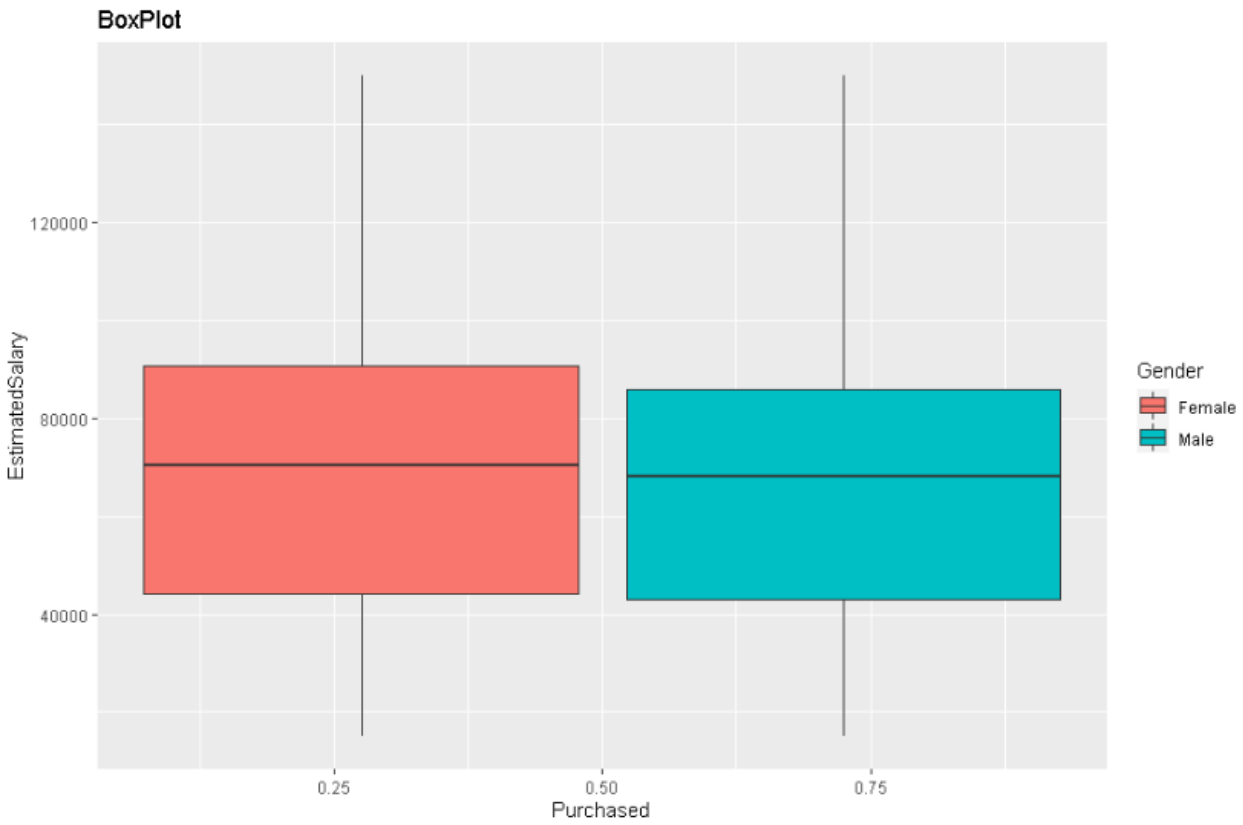
This graph represents the total number of males and females' clients who Purchased. By initial examination of this graph one can make out that a greater number of female clients purchased than males. This could be because most of the female clients had no private cars for themselves, this could a reason or better offer was pulled off by them.

6. Boxplot

```
> s <-ggplot(data=data,aes(x=Purchased,y=Age))
> s + geom_boxplot(aes(fill=Gender))+
+   ggtitle("BoxPlot")+
+   xlab("Purchased")+
+   ylab("Age")
```




```
> s <-ggplot(data=data,aes(x=Purchased,y=EstimatedSalary))
> s + geom_boxplot(aes(fill=Gender))+
+   ggtitle("BoxPlot")+
+   xlab("Purchased")+
+   ylab("EstimatedSalary")
```


BoxPlot
120000
80000
40000
EstimatedSalary
0.25
0.50
0.75
Purchased
Gender
Female
Male

D. Steps Performed

1.Installing the necessary packages

- caTools

```
> library(caTools)
```

- ggplot2

```
> library(ggplot2)
```

- class

```
> library(class)
```

- caret

```
> library(caret)
```

2.Fetching the respective packages from the library

3. Loading the data in R from your working directory

a. The data Social_Network_Ads.csv is a windows comma separated value (csv) file that contains 5 variables and 400 observations

4.Feature Scaling the dataset

Before feature scaling dataset

```
> dataset
     User.ID Gender Age EstimatedSalary Purchased
1   15624510   Male  19            19000         0
2   15810944   Male  35            20000         0
3   15668575 Female  26            43000         0
4   15603246 Female  27            57000         0
5   15804002   Male  19            76000         0
6   15728773   Male  27            58000         0
7   15598044 Female  27            84000         0
8   15694829 Female  32           150000         1
9   15600575   Male  25            33000         0
10  15727311 Female  35            65000         0
11  15570769 Female  26            80000         0
12  15606274 Female  26            52000         0
13  15746139   Male  20            86000         0
14  15704987   Male  32            18000         0
15  15628972   Male  18            82000         0
16  15697686   Male  29            80000         0
17  15733883   Male  47            25000         1
18  15617482   Male  45            26000         1
19  15704583   Male  46            28000         1
20  15621083 Female  48            29000         1
21  15649487   Male  45            22000         1
22  15736760 Female  47            49000         1
23  15714658   Male  48            41000         1
24  15599081 Female  45            22000         1
25  15705113   Male  46            23000         1
```

From summary we can see that not all the variables are numeric so the variables which are numeric shall be considered and not the categorical data. So, as you see only variable numeric variable are scaled off for regression.

After Feature Scaling the Dataset

```
> dataset = dataset[,3:5]
> dataset
    Age EstimatedSalary Purchased
1    19           19000         0
2    35           20000         0
3    26           43000         0
4    27           57000         0
5    19           76000         0
6    27           58000         0
7    27           84000         0
8    32          150000         1
9    25           33000         0
10   35           65000         0
11   26           80000         0
12   26           52000         0
13   20           86000         0
```

As you can see the difference where the only numeric variable is been considered and rest of is been excluded from the dataset. Age, Estimated Salary and the Purchased column is been considered .

5.Data Cleaning

a. Missing value

b. Categorical value

6. Creating a Training and Test dataset with 75% being the training data and 25% being the test data. (Note: Both Training and Test data should wholly represent the original dataset.)

```
> #Splitting the dataset into training and testing dataset
> #installing the package (caTools)
> library(caTools)
> set.seed(123)
> split = sample.split(dataset$Purchased,SplitRatio=0.75)
> training_set = subset(dataset,split==TRUE)
> testing_set = subset(dataset,split==FALSE)
```

As you can see the dataset is been splitted in 2 parts i.e ., training data and testing data which helps to us predict the data and helps us understand easily.

7.Machine Learning Models

Model 1: Logistic Regression

Fitting Logistic regression to the training set.

```
> #Splitting the dataset into training and testing dataset
> #installing the package (caTools)
> library(caTools)
> set.seed(123)
> split = sample.split(dataset$Purchased,SplitRatio=0.75)
> training_set = subset(dataset,split==TRUE)
> testing_set = subset(dataset,split==FALSE)
> #---------------Feature Scaling
> training_set[,1:2] = scale(training_set[,1:2])
> testing_set[,1:2] = scale(testing_set[,1:2])
> #Logistic regression
> classifier = glm(formula = Purchased ~.,
+                  family = binomial,
+                  data=training_set)
```

Purchase is our dependent variable and Age and Estimated Salary are the independent attributes. Glm function is used to fit generalized linear models specified by giving symbolic description of linear predictor and description of the error distribution.

➢ Statistics

Summary of the classifier

```
> summary(classifier)

Call:
glm(formula = Purchased ~ ., family = binomial, data = training_set)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.0753  -0.5235  -0.1161   0.3224   2.3977

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)      -1.1923     0.2018  -5.908 3.47e-09 ***
Age               2.6324     0.3461   7.606 2.83e-14 ***
EstimatedSalary   1.3947     0.2326   5.996 2.03e-09 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 390.89  on 299  degrees of freedom
Residual deviance: 199.78  on 297  degrees of freedom
AIC: 205.78

Number of Fisher Scoring iterations: 6
```

As you can see from the summary above Deviance Residuals, coefficients in this in the last column you can see that the p-values of all are highly statistically significant as they are well below 0.05.Both variables has 100% level of confidence.

AIC can be used to compare one model to another. Lastly the Number of Fisher Scoring iterations (6) tells us how quickly the glm () function converged maximum likelihood estimates for the coefficients.

Confidence Interval of the fit

```
> #Confidence interval of the fit
> confint(classifier)
Waiting for profiling to be done...
                    2.5 %     97.5 %
(Intercept)     -1.6085769 -0.8139815
Age              2.0143765  3.3786905
EstimatedSalary  0.9666725  1.8837549
```

In this, the default confidence interval is been set that is 95%.

➢ Predicting Values

Predicting the test set Results using the classifier in the previous page.

```
> #Predicting the Test set Results
> yprob = predict(classifier,type="response",newdata =testing_set)
> ypred =ifelse(yprob < 0.5 ,0,1)
> ypred
  2   4   5   9  12  18  19  20  22  29  32  34  35  38  45  46  48  52  66  69  74  75  82  84  85
  0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 86  87  89 103 104 107 108 109 117 124 126 127 131 134 139 148 154 156 159 162 163 170 175 176 193
  0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
199 200 208 213 224 226 228 229 230 234 236 237 239 241 255 264 265 266 273 274 281 286 292 299 302
  0   0   1   1   1   0   1   0   1   1   1   0   1   1   1   0   1   1   1   1   1   0   1   1   1
305 307 310 316 324 326 332 339 341 343 347 353 363 364 367 368 369 372 373 380 383 389 392 395 400
  0   1   0   0   0   0   1   0   1   0   1   1   1   1   1   1   0   1   0   1   1   0   0   0   1
```

Usually we do it one step but in logistic regression we do in 2 steps. So first let's take anyway the probabilities.

So where are we going to call them problem pred which is the vector of period the which will be the vector of the predicted probabilities of our test set observations by the classifier.

In the next step we will predict the test set using the vector of the predicted probabilities. We are interested in the value 0 or 1.By using the yprob vector if the value is smaller than 0.5 then its 0 and has low chances to buy the SUVs bigger than 0.5 then 1 it has higher chances to buy SUVs. By this we get the predicted values using the test results.

➢ Creating actual confusion matrix based on predicted values

Now we will evaluate those predictions by making the confusion matrix which will count the number of correct predictions and the number of incorrect predictions.

```
> #Confusion Matrix
> cm = table(testing_set[,3],ypred)
> cm
   ypred
     0  1
  0 57  7
  1 10 26
```

➢ Interpreting Results

And here it is the most important thing to understand here is that the 57 and the 26 here are the correct predictions and the 10 and the 7 here are the incorrect predictions.

So what's interesting here at first sight is that the classifier made 57 plus 26 equals 83 correct predictions and 10 plus 7 equals 17 are Incorrect predictions.

All right so 17 incorrect predictions on the test set.

That's not bad but we can do better and we will do better with other classifiers.

As performance of model is calculate using formula that is :

(TP + TN)/ (TP +TN + FP + FN)

(57 + 26)/ (57 + 26 + 10 + 7) = 83% Accuracy of the model.

Model 2: Multilinear regression

Multiple regression simultaneously considers influence variable in response to the Y variable.

Fitting multilinear Regression to training data set

```
> # Multilinear regression
> #Splitting the dataset
> library(caTools)
> set.seed(123)
> split = sample.split(dataset$Purchased,SplitRatio = 0.75)
> training_data = subset(dataset,split==TRUE)
> testing_set = subset(dataset,split==FALSE)
>
> #Feature Scaling
> training_data[,1:2] = scale(training_data[,1:2])
> testing_set[,1:2] = scale(testing_set[,1:2])
>
> #Fitting Multilinear regression to trainig data
> regressor = lm(formula = Purchased ~ Age + EstimatedSalary,
+                data=training_data)
```

As you can see it is a linear regression model .lm is used to fit linear models. lm() function is used to find out the correlation between the attributes. Purchased column is our dependent variable and the age and Estimated Salary is our independent variable from which we are going to predict.

➢ Statistics

Summary of the Regressor

```
> summary(regressor)

Call:
lm(formula = Purchased ~ Age + EstimatedSalary, data = training_data)

Residuals:
     Min       1Q   Median       3Q      Max
-0.97297 -0.28534 -0.03195  0.25710  0.81937

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)      0.35667    0.02031  17.564  < 2e-16 ***
Age              0.27570    0.02061  13.375  < 2e-16 ***
EstimatedSalary  0.13787    0.02061   6.689 1.12e-10 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.3517 on 297 degrees of freedom
Multiple R-squared:  0.4662,    Adjusted R-squared:  0.4626
F-statistic: 129.7 on 2 and 297 DF,  p-value: < 2.2e-16
```

Generally, good threshold for level of significance is 5%. That means that if your p-value is lower than the threshold which is 5% it is highly significant and if not then low significant or else no significant. By looking at the last column both have 3 stars that means both the independent variable are highly statistically significant to dependent variable. Both the variables are having 100% level of confidence. In this you can see that the Multiple R-squared and the Adjusted R-squared is 46.62% which means the variables contribute 46.62% to the variability which is not so good. It shows us that Residual standard error is 35%.

Confidence Interval of the fit

```
> #Confidence interval of the fit
> confint(regressor)
                    2.5 %    97.5 %
(Intercept)     0.31670235 0.3966310
Age             0.23513443 0.3162681
EstimatedSalary 0.09730796 0.1784416
```

To compute the confidence interval we use confint() function, by specifying the name of the fitted model in this case our fitted model name is regressor.

Usually, our default confidence interval is 95%,and we can adjust it by changing the level by any percent you want.

Coefficient correlation

```
> #Coefficient correlation
> coef(regressor)
    (Intercept)             Age EstimatedSalary
      0.3566667       0.2775035       0.1358544
```

As you can see, that there is good correlation between dependent and independent variables. Both independent variables have good impact on the Purchased(target variable).

➢ Predicting Values

Predicting the test set Results using the regressor vector in the previous page.

```
> #Predict
> xpred = predict(regressor,type="response",newdata=testing_set)
> pred = ifelse(xpred < 0.5,0,1)
> pred
  2   4   5   9  12  18  19  20  22  29  32  34  35  38  45  46  48  52  66  69  74  75  82  84  85
  0   0   0   0   0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 86  87  89 103 104 107 108 109 117 124 126 127 131 134 139 148 154 156 159 162 163 170 175 176 193
  0   0   0   0   1   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
199 200 208 213 224 226 228 229 230 234 236 237 239 241 255 264 265 266 273 274 281 286 292 299 302
  0   0   1   1   1   0   1   0   1   1   1   0   1   1   1   0   1   1   1   1   1   0   1   1   1
305 307 310 316 324 326 332 339 341 343 347 353 363 364 367 368 369 372 373 380 383 389 392 395 400
  0   1   0   0   0   0   1   0   1   0   1   1   1   1   1   1   0   1   0   1   1   0   0   0   0
```

Converting the values which are less than 0.5 to 0 and which are bigger than 0.5 equals to 1 that is the chances of them are higher compared to less than 0.5 ones to buy SUVs.

➢ Creating actual confusion matrix based on predicted values

Now we will evaluate those predictions by making the confusion matrix which will count the number of correct predictions and the number of incorrect predictions.

```
> #Confusion matrix
> Multi = table(testing_set$Purchased,pred)
> Multi
   pred
     0  1
  0 57  7
  1 11 25
```

➢ Interpreting Results

And here it is the most important thing to understand here is that the 57 and the 25 here are the correct predictions and the 11 and the 7 here are the incorrect predictions.

So what's interesting here at first sight is that the classifier made 57 plus 25 equals 82 correct predictions and 11 plus 7 equals 18 are Incorrect predictions.

All right so 18 incorrect predictions on the test set.

That's not bad but we can do better.

As performance of model is calculate using formula that is :

(TP + TN)/ (TP +TN + FP + FN)

(57 + 25)/ (57 + 25 + 11 + 7) = 82% Accuracy of the model.

Model 3: KNN

K – Nearest Neighbour

K- neighbour classification for test set from training set. We need to specify the number of neighbours. KNN is not a linear classifier.

```
> # KNN
>
> dataset <-read.csv(file.choose(),header=T)
> dataset = dataset[,3:5]
> View(dataset)
>
> #Split the dataset
> library(caTools)
> set.seed(123)
> split = sample.split(dataset$Purchased,SplitRatio=0.75)
> training_set = subset(dataset,split==TRUE)
> View(training_set)
> testing_set = subset(dataset,split== FALSE)
> View(testing_set)
>
> #Feature Scaling
> training_set[,1:2] = scale(training_set[,1:2])
> testing_set[,1:2] = scale(testing_set[,1:2])
```

As Usual, we split the dataset in training data and testing set from which we can predict .We consider only attributes which are numeric.

➢ Predicting Values

```
> #Fitting KNN on training data set
>
> library(class)
> ypred = knn(train= training_set,
+             test=testing_set,
+             cl=training_set[,3],
+             k=5)
> ypred
  [1] 0 0 0 0 0 1 1 1 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 [49] 0 0 0 0 0 1 1 0 1 0 1 1 1 0 0 1 0 0 1 1 1 1 1 1 1 0 1 0 1 0 1 1 0 1 0 1 0 1 1 1 0 1 1 0 1 0 1 1
 [97] 1 1 0 1
Levels: 0 1
```

As you can see, we have installed new package that is the class() package .By the help of the class package we can use knn function to predict the values. As we have number of arguments first one is the train i.e., dataset of training cases. Then we have test similar to the first argument dataset of testing cases. Argument cl means factor of true classification of training set i.e., the Purchased column in our dataset and the last argument is the k value which is number of neighbours considered .By default k value is set to be 5.

- Creating actual confusion matrix based on predicted values

```
> #Making confusion matrix
>
> cm2 <- table(testing_set$Purchased,ypred)
> cm2
   ypred
     0  1
  0 62  2
  1  0 36
```

- Interpreting Results

And here it is the most important thing to understand here is that the 62 and the 36 here are the correct predictions and the 0 and the 2 here are the incorrect predictions.

So what's interesting here at first sight is that the classifier made 62 plus 36 equals 98 correct predictions and 0 plus 2 equals 2 are Incorrect predictions.

All right so 2 incorrect predictions on the test set.

As performance of model is calculate using formula that is :

(TP + TN)/ (TP +TN + FP + FN)

(62 + 36)/ (62 + 36 + 0 + 2) = 98% Accuracy of the model.

E. Results

```
> #model 1
> abc <-confusionMatrix(cm)
> abc
Confusion Matrix and Statistics

   ypred
     0  1
  0 57  7
  1 10 26

               Accuracy : 0.83
                 95% CI : (0.7418, 0.8977)
    No Information Rate : 0.67
    P-Value [Acc > NIR] : 0.0002624

                  Kappa : 0.6242

 Mcnemar's Test P-Value : 0.6276258

            Sensitivity : 0.8507
            Specificity : 0.7879
         Pos Pred Value : 0.8906
         Neg Pred Value : 0.7222
             Prevalence : 0.6700
         Detection Rate : 0.5700
   Detection Prevalence : 0.6400
      Balanced Accuracy : 0.8193

       'Positive' Class : 0
```

```
> #Model 2
> pqr <-confusionMatrix(Multi)
> pqr
Confusion Matrix and Statistics

   pred
     0  1
  0 57  7
  1 11 25

               Accuracy : 0.82
                 95% CI : (0.7305, 0.8897)
    No Information Rate : 0.68
    P-Value [Acc > NIR] : 0.001247

                  Kappa : 0.5996

 Mcnemar's Test P-Value : 0.479500

            Sensitivity : 0.8382
            Specificity : 0.7812
         Pos Pred Value : 0.8906
         Neg Pred Value : 0.6944
             Prevalence : 0.6800
         Detection Rate : 0.5700
   Detection Prevalence : 0.6400
      Balanced Accuracy : 0.8097

       'Positive' Class : 0
```

```
> #Model 3
> cm3 <-confusionMatrix(data=cm2)
> cm3
Confusion Matrix and Statistics

   ypred
     0  1
  0 62  2
  1  0 36

               Accuracy : 0.98
                 95% CI : (0.9296, 0.9976)
    No Information Rate : 0.62
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9571

 Mcnemar's Test P-Value : 0.4795

            Sensitivity : 1.0000
            Specificity : 0.9474
         Pos Pred Value : 0.9688
         Neg Pred Value : 1.0000
             Prevalence : 0.6200
         Detection Rate : 0.6200
   Detection Prevalence : 0.6400
      Balanced Accuracy : 0.9737

       'Positive' Class : 0
```

Depending on the predicted values from various models we have created a confusion matrix for each model.

As you can see from the above that Model 3 i.e. KNN model is highly accurate model that is 98% compared to other two models and can be used to predict whether which of the clients have purchased the cars. Model 3 is best followed by Model 1 i.e. 83% and Model 2 i.e. 82%.Model 1 and 2 has not much difference but Model 3 is way better than this two.

F. Conclusion

We can develop ‘n’ number of models but in our case, we have put 3 models which are put to test. Among them Model 3 “KNN Model” is the most accurate model i.e. 98%.Based on the analysis carried out we can say buying of the cars by clients can be predicted upto 98%.