

Prediction and classification of wages using regression tree and random forest method

PROJECT REPORT

Software Testing (ITE2004)

Slot: A2+TA2

Submitted in partial fulfillment for the award of the degree of
B. Tech in Information Technology

By

NAME REG.NO

1. Reshabh Kumar Jain 17BIT0048

2. Aradhya Mathur 17BIT0146

Under the guidance of

Dr. Uma K



VIT[®]
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

JUNE 2020

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	4
	LIST OF TABLES	5
	LIST OF FIGURES	5
1.	INTRODUCTION	
	1.1 Motivation	6
	1.2 Issues	6
2.	PROBLEM DEFINITION	
	2.1 Objective	8
	2.2 Literature Survey	8
	2.3 Dataset Description	12
3	PROJECT DESCRIPTION	
	3.1 Work Flow Diagram	14
	3.2 List of Modules	14
	3.3 Requirements Specification	15
	3.3.1 Software Requirements	15
	3.3.2 Hardware Requirements	15
	3.4 Module Description	15
	3.4.1 Data Preparation	15
	3.4.2 Pre-processing of the data	15
	3.4.3 Applying CART Decision Tree	16
	3.4.4 Applying J48Decision Tree	17
	3.4.5 Applying Random Forest Algorithm	17
4	IMPLEMENTATION OF PROJECT	
	4.1. Source code	19
5	TESTING	
	5.1 System Testing	21

	5.1.1 Unit Testing	21
	5.1.2 Integration Testing	22
	5.1.3 White box Testing	22
	5.1.4 Blackbox Testing	22
	5.1.5 Validation Testing	23
	5.1.6 User Acceptance Testing	23
	5.1.6 R-Unit Testing	23
6	RESULTS AND DISCUSSION	
	6.1 Data Visualization	25
	6.2 Result of CART Decision Tree	27
	6.3 Result of J48 Decision Tree	31
	6.4 Result of Random Forest Algorithm	33
	6.5 Comparison of Algorithm	33
7	Conclusion	35
	References	36
8	Digital Assignment 2	38

ABSTRACT

This project will investigate the data mining of demographic data in order to create one or more classification models which are capable of accurately identifying individuals whose salary exceeds a specified value. The data used in this project were sourced from the University of California Irvine data repository and are referred to as the Adult dataset and contain information on individuals such as age, level of education and current employment type.

We are interested to learn how well we can predict whether an individual's annual income exceeds \$50,000 using the set of variables in this data set. And Two different data mining algorithms classification and regression tree, random forest is used to answer the same question. A description of the predictive significance of each attribute, interesting or useful patterns which were found and any transformations applied to the data is provided with all proposed models.

LIST OF TABLES

Table No	Title	Page No
1	Dataset Description	12
2	Dataset Details	13
3	Results and Observations	34

LIST OF FIGURES

FIG. NO	TITLE	PAGE NO
1	Workflow Diagram	14
2	Random Forest Diagram	18
3	RUnit Testing Output	24
4	Box Plot of Age By Income	25
5	Count vs Numeric	26
6	Age vs Mean Hours per Week by Gender	26
7	CART Decision Output 1	27
8	CART Decision Output 2	27
9	CART Decision Output 3	28
10	CART Decision Output 4	29
11	CART Decision Output 5	30
12	CART Decision Output 6	30
13	J48 Decision Tree Output 1	31
14	J48 Decision Tree Output 2	32
15	Random Forest Output	33
16	Comparison of Algorithms 1	33
17	Comparison of Algorithms 2	34

CHAPTER 1 – INTRODUCTION

1. INTRODUCTION

1.1 Motivation

The adult dataset is from the 1994 Census database. It is also known as “Census Income” dataset. Details of this dataset can be found at UCI Machine Learning Repository. We are interested to learn how well can I predict whether an individual’s annual income exceeds \$50,000 using the set of variables in this data set. The question is inspected in two different classification and regression tree and random forest are used to answer the same question. We aim to predict whether an individual’s income will be greater than \$50,000 per year based on several attributes from the census data. The data mining objective is to create a classification model which can predict individuals whose salary exceeds fifty thousand US dollars by mining anonymized census data containing demographic information such as age, gender, education level and employment type.

In our initial stages, we will preprocess the data and develop understanding of the data and its useful features that explain the variances by doing various types of exploratory analysis. Later, we will move on to a classification task of predicting whether the income is $\geq 50k/\text{year}$ from a person’s attributes, by using important features. For the classification task, we plan to implement various data mining techniques. After implementing various data mining models, we compared their results on the training and the test set to arrive at a model that works best for the predictive task on both test and training data set with 88% accuracy.

1.2 Issues

Over the last two decades, humans have grown a lot of dependence on data and information in society and with this advent growth, technologies have evolved for their storage, analysis and processing on a huge scale. The fields of Data Mining and Machine Learning have not only exploited them for knowledge and discovery but also to explore certain hidden patterns and concepts which led to the prediction of future events, not easy to obtain. The problem of income

inequality has been of great concern in the recent years. Making the poor better off does not seem to be the sole criteria to be in quest for eradicating this issue. This model actually aims to conduct a comprehensive analysis to highlight the key factors that are necessary to predict an individual's income. Such an analysis helps to set focus on the important areas which can significantly determine the income of individuals.

CHAPTER 2 – PROBLEM DEFINITION

2.1 OBJECTIVE

The main objective of the project is to analyze the data set and predict income of the person. We will use **Decision Tree Algorithm and Random Forest Algorithm** to predict the same. The **Decision Tree Algorithm and Random Forest Algorithm is implemented in R language.**

The data mining objective is to create a classification model which can predict individuals whose salary exceeds fifty thousand US dollars by mining anonymised census data containing demographic information such as age, gender, education level and employment type. The original salary attribute in the census data has been anonymised to a binomial value indicating if a salary exceeds fifty thousand US dollars. For each proposed model an expected return on investment (and associated profit margins) must be provided. A clear description of all data transformations which were carried out must be provided and any useful insights into the data such as significant attributes or mining issues within the data should be included.

2.2 LITERATURE SURVEY

Income inequality is one of the key issues governments are trying to solve. Reduced income disparity ensures balanced social development across different groups and improves the economic growth and political stability of a country. Governments in different countries are using different interventions to address income inequality, some are succeeding while the others not. One of the key reasons behind failure is doing many things which results into reduced efficiency and lower results. This study aims to conduct preliminary analysis that can be used to understand which factors are more important in improving individuals income. By focusing on few important areas governments can improve efficiency and achieve success at higher rates in reducing income inequality.

Most university students selected their degree courses without any future career goals, choosing a field of study following their friends or current trends. This caused boredom while studying which often led to failure through bad performance in the examinations. Some students dropped out from university willingly while others were told to leave. To increase motivation, a salary prediction system was developed based on a decision tree model with two features. The first identified examples of successful graduate students with a good career and salary. The second feature was the salary prediction system, where salary was the main factor for improving the quality of life

Predict whether an individual's annual income exceeds \$50,000 using the set of variables in this data set. The question is inspected in two different approaches – traditional statistical modeling and machine learning techniques. Logistic regression is used as the statistical modeling tool as the outcome is binary.

In this paper after giving introduction of data mining, decision tree is discussed in detail, how it is constructed. The adult dataset is popular data set. In this there are more than thirty two thousand instances. Using Weka, an important data mining tool, the decision tree is constructed for country India in adult data set. Using that decision tree the knowledge can be extracted.

The purpose of this study is to verify the effectiveness of a data- driven approach for financial statement analysis. In the area of accounting, variable selection for construction of models to predict firm's earnings based on financial statement data has been addressed from perspectives of corporate valuation theory, etc., but there has not been enough verification based on data mining techniques.

This paper develops a framework for looking at the determination and prediction of permanent income and applies it to Chinese household data. By pooling information from both determinants and correlates of permanent income we can derive predictors of permanent income which statistically outperform observed income and consumption. Simultaneous estimation of the model enables us to compare how well different observable welfare indicators identify household permanent income. Our framework also provides insights into the processes generating permanent income and allows us to draw inferences concerning the power of different forms of public policy to influence permanent income.

People of different profiles have different investment strategies as per the humanly attributes. The work in the paper concerns the analysis of the investment patterns which are affected by the attributes of a person it carries in the given population. The classification of the population into groups as per the profile attributes and then analysis of the patterns in investment needs to be done. This paper describes data mining with predictive analytics for financial applications and explores methodologies and techniques in data mining area combined with predictive analytics for application driven results for financial data. The basic idea is to apply patterns on available data and generate new assumptions and anticipated behavior using predictive analysis.

This paper model wages by exploiting a unique Finnish dataset. For both genders the data is divided into four income quartiles. A panel data model is estimated with the three degree polynomials of age, the duration of employment, and GDP growth as explanatory variables. Individual variation within a wage quartile is shown to be large and an important risk factor. The model provides predictors also for individual wages.

With the help of our work, we will be predicting the income of the population and analyzing the factors which strongly affect the income. We will be giving suggestions based on the result obtained which level of qualification can lead to a higher income and people of which age group are earning more. We will implement different models and find out which one is of higher accuracy. Also, we will consider the outputs of all the models and take a vote to determine the overall result.

Decision tree is the considered as the powerful solution to the classification problems and it is applied in many real-world application. Many data mining techniques are used for weather forecasting in the present scenario, with various levels of accuracy. A) Data Collection B) Data Cleaning C) Data Selection D) Data Transformation. E) Data Mining Stage

On these datasets we applied logistic regression and decision trees (J48), as well as random forests, bagging and boosting of decision trees, in order to obtain predictive models of loan prediction. Best results in terms of predictive accuracy were obtained by bagging decision trees.

In this research paper statistical approach was used to predict loans. It is secondary data it is usually cheap and is less time consuming because someone else has compiled it. Patterns and correlations are clear and visible It is taken from large samples so the generalizability is high. It can be used and re-used to check different variables. Also, can be imitated to check changes which increases reliability and representativeness.

Decision trees implicitly perform variable screening or feature selection Decision trees require relatively little effort from users for data preparation Relatively simple learning algorithm Scales well to larger datasets with new GPGPU hardware and CUDA software Can significantly outperform other models when the conditions are right It is good in determine the relative influence of one or more predictor variables to the criterion value. It is good in to identify outliers, or anomalies.

Highly efficient since boosted trees are derived by optimizing an objective function, basically GBM can be used to solve almost all objective function that we can write gradient out. This including things like ranking and poisson regression, which RF is harder to achieve.

2.3 DATASET DESCRIPTION

	Attribute	Values	Missing					
Polynomials	Employment Class	Private (68%), Self employed 1 (8%), Local Gov(6%), State Gov(4%), Unknown (5%), Self employed 2 (3%), Federal Gov(3%), No Pay(1.5%), Never Worked (0.5%)	1836					
	Education Level	High School (32%), Some college (22%), Bachelors (16%), Masters (5%), Vocational (4%), 11th (4%), Assoc Academic (3%), 10th (3%), 7-8th (2%), Professional School (2%), 9th (2%), 12th (2%), Doctorate (1%), 5-6th (1%), 1-4th (1%), Preschool (1%)	0					
	Relationship	Husband (41%), Not-in-family (26%), Own child (16%), Unmarried (11%), Wife (4%), Other relative (2%)	0					
	Race	White (85%), Black (10%), Asian / Pacific Islander (3%), American Indian / Eskimo (1%), Other (1%)	0					
	Marital Status	Married-civ-spouse (46%), Never-married (33%), Divorced (14%) Separated (3%), Widowed (2%), Married-AF-spouse (1%), Married-spouse-absent (1%)	0					
	Occupation	15 categories	1843					
	Country	42 categories: USA (90%)	583					
Binomials	Salary[Label]	<=\$50K (76%), >\$50K (24%)	0					
	Gender	Male (67%), Female (33%)	0					
Real		Mean	Median	Std Dev	Skewness	Kurtosis	Range	
	Age	38.58	37	13.64	0.56	2.83	17 - 90	0
	Hours worked per week	40.44	40	12.35	0.23	5.92	1 - 99	0
	Education Number	10.08	10	2.57	-0.31	3.62	1 - 16	0
	Capitla Gain	1078	0	7385	11.95	157.77	0 - 99999	0
	Capital Loss	87.3	0	403	4.59	23.37	0 - 4356	0
	Survey Weight	189778	178356	105550	1.45	9.22	12285 - 1484705	0

Table1: Dataset Description

Census Income dataset has 48,842 entries. Each entry contains the following information about an individual:

- **age**: the age of an individual
 - Integer greater than 0
- **work class**: a general term to represent the employment status of an individual

- Private, Self-emp-not-inc, Self-emp-inc, Federal-gov, Local-gov, State-gov, Without-pay, Never-worked.
- **fnlwgt**: final weight. In other words, this is the number of people the census believes the entry represents.
 - Integer greater than 0
- **education**: the highest level of education achieved by an individual.

Number of Instances: 48842 Number of Attributes: 14

Data Set Characteristics:	Multi-variate	Number of Instances:	48842	Area:	Social
Attribute Characteristics:	Categorical, Integer	Number of Attributes:	14	Date Donated	1998-05-01
Associated Tasks:	Classification	Missing Values?	Yes	Number of Web Hits:	1058912

Table 2: Dataset Details

CHAPTER 3 – PROJECT DESCRIPTION

3.1 WORK FLOW DIAGRAM

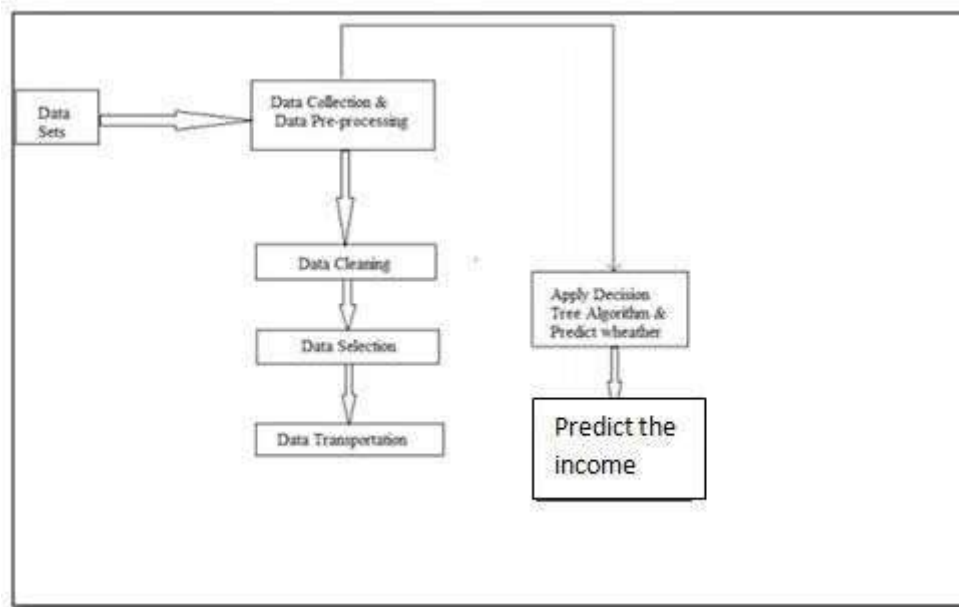


Figure1: Workflow Diagram

3.2 LIST OF MODULES

- Data Preparation
- Pre-Processing of the Data
- Apply CART Desicion Tree Algorithm
- Apply J48 Decision Tree Algorithm
- Apply Random Forest Algorithm

3.3 REQUIREMENTS SPECIFICATION

3.3.1 SOFTWARE REQUIREMENTS

- Operating System : Ubuntu 12 or above / Windows 7 or above
- Language : R programming language
- Platform : R STUDIO
- Testing Tool : R Tool

3.3.2 HARDWARE REQUIREMENTS

- CPU speed : Pentium IV or above with 1.9GHz
- HDD : 40GB min
- Mouse type : Optical
- Keyboard type : 101 keys
- Monitor : min 15" VGA or above
- RAM : 2GB or above
- Smartphone : Android 4.4.2 or above
- Network : Wifi or hotspot, wired Ethernet network

3.4 MODULE DESCRIPTION

3.4.1 DATA PREPARATION

The process of data cleaning and preparation is highly dependent on the specific data mining algorithm and software chosen for the data mining task. Here we attempted to prepare the data according to the requirements of the selected data mining software.

3.4.2 PRE-PROCESSING OF THE DATA

A data set collected is not directly suitable for induction (knowledge acquisition), it comprises in most cases noise, missing values, and inconsistent data set is too large, and so on. Therefore, we need to minimize the noise in data, choose a strategy for handling missing (unknown) attribute values, use any suitable method for selecting and ordering attributes (features) according to their

informatively (so-called attribute mining), discredited/fuzzily numerical (continuous) attributes, validating part of training data to be used for creating model and eventually process continuous classes.

Some of the variables are not self-explanatory. The continuous variable `fnlwgt` represents final weight, which is the number of units in the target population that the responding unit represents. The variable `education_num` stands for the number of years of education in total, which is a continuous representation of the discrete variable education. The variable `relationship` represents the responding unit's role in the family. `capital_gain` and `capital_loss` are income from investment sources other than wage/salary.

For simplicity of this analysis, the weighting factor is discarded. Total number of years of education can represent by the highest education level completed. Role in the family can be assessed from gender and marital status. Thus, the following 3 variables are deleted education, relationship, and `fnlwgt`.

3.4.3 Applying CART Decision Tree

A decision tree is a decision support tool that uses a tree-like graph or model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements. A decision tree is a flowchart-like structure in which each internal node represents a "test" on an attribute (e.g. whether a coin flip comes up heads or tails), each branch represents the outcome of the test, and each leaf node represents a class label (decision taken after computing all attributes). The paths from root to leaf represent classification rules. In decision analysis, a decision tree and the closely related influence diagram are used as a visual and analytical decision support tool, where the expected values (or expected utility) of competing alternatives are calculated. A decision tree consists of three types of nodes:

1. Decision nodes – typically represented by squares
2. Chance nodes – typically represented by circles
3. End nodes – typically represented by triangles

3.4.4 Applying J48 Decision Tree

J48 builds decision trees from a set of training data using the concept of information entropy. The training data is a set of already classified samples. Each sample consists of a p-dimensional vector where the represent attribute values or features of the sample, as well as the class in which falls.

At each node of the tree, J48 chooses the attribute of the data that most effectively splits its set of samples into subsets enriched in one class or the other. The splitting criterion is the normalized information gain. The attribute with the highest normalized information gain is chosen to make the decision. The J48 algorithm then recurses on the partitioned sublists.

This algorithm has a few base cases.

- All the samples in the list belong to the same class. When this happens, it simply creates a leaf node for the decision tree saying to choose that class.
- None of the features provide any information gain. In this case, C4.5 creates a decision node higher up the tree using the expected value of the class.
- Instance of previously-unseen class encountered. Again, C4.5 creates a decision node higher up the tree using the expected value.

3.4.5 Applying Random Forest Algorithm

Random Forest is an ensemble learning (both classification and regression) technique. It is one of the commonly used predictive modelling and machine learning technique. In a normal decision tree, one decision tree is built and in a random forest algorithm number of decision trees are built during the process. A vote from each of the decision trees is considered in deciding the final class of a case or an object, this is called ensemble process. This is a democratic process. Since, many decision trees are built and used in a process of Random Forest algorithm, it is called a forest. Now, why is it “random”? A data frame (or SAS dataset) has two dimensions - observation (or rows) and variables (or columns). For a building a decision tree, samples of a data frame are

selected with replacement along with selecting a subset of variables for each of the decision tree. Both sampling of data frame and selection of subset of the variables are done randomly, so first word “random” is arrived.

Key advantages of using Random Forest

- Reduce chances of over-fitting
- Higher model performance or accuracy

Random Forest uses Gini Index based impurity measures for building decision tree. Gini Index is also used for building Classification and Regression Tree (CART). In earlier blogs we have explained working of CART Decision Tree and a worked out example of Gini Index calculation

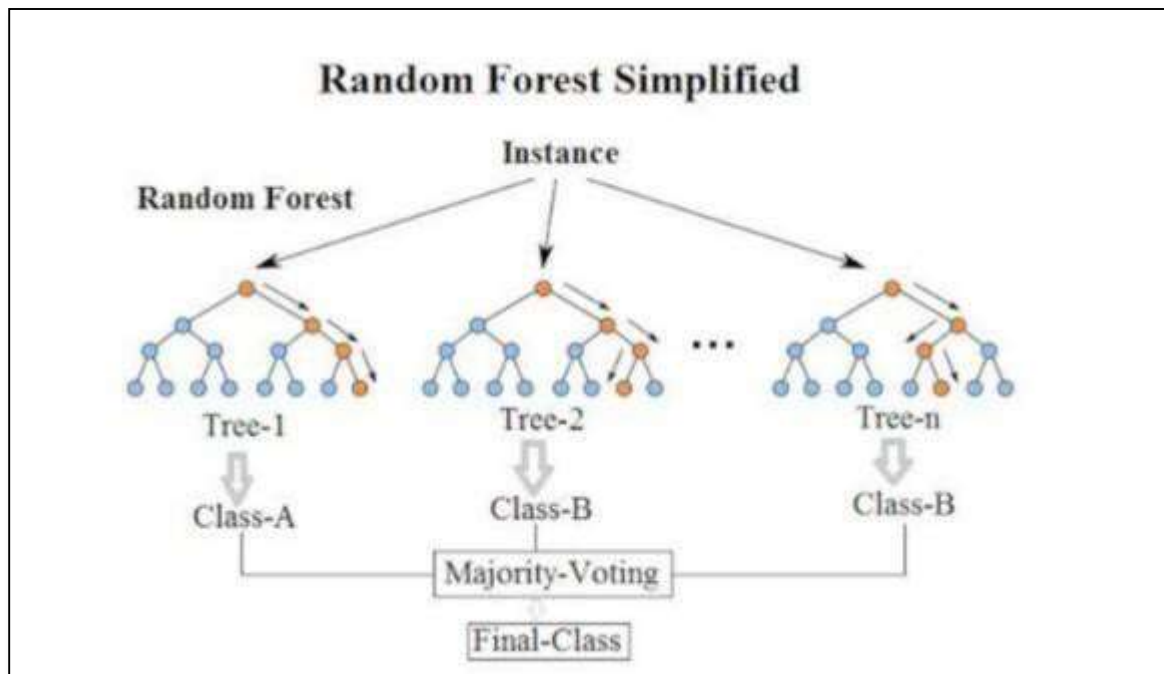


Figure2: RandomForest Diagram

CHAPTER 4 – IMPLEMENTATION

4.1 Source Code

```
levels(adult$workclass)[1] <- 'Unknown' adult$workclass <- gsub('^Federal- gov', 'Government',
adult$workclass) adult$workclass <- gsub('^Local-gov', 'Government', adult$workclass)
adult$workclass <- gsub('^State-gov', 'Government', adult$workclass) adult$workclass <-
gsub('^Self-emp-inc', 'Self- Employed', adult$workclass) adult$workclass <- gsub('^Self-emp-
not-inc', 'Self- Employed', adult$workclass) adult$workclass <- gsub('^Never-worked', 'Other',
adult$workclass) adult$workclass <- gsub('^Without-pay', 'Other', adult$workclass)
adult$workclass <- gsub('^Other', 'Other/Unknown', adult$workclass) adult$workclass <-
gsub('^Unknown', 'Other/Unknown', adult$workclass) adult$workclass <-
as.factor(adult$workclass)
```

```
summary(adult$workclass)
```

```
levels(adult$occupation)[1] <- 'Unknown' adult$occupation <- gsub('Adm- clerical', 'White-
Collar', adult$occupation) adult$occupation <- gsub('Craft- repair', 'Blue-Collar',
adult$occupation) adult$occupation <- gsub('Exec- managerial', 'White-Collar', adult$occupation)
adult$occupation <- gsub('Farming-fishing', 'Blue-Collar', adult$occupation) adult$occupation <-
gsub('Handlers-cleaners', 'Blue-Collar', adult$occupation) adult$occupation <- gsub('Machine-op-
inspct', 'Blue-Collar', adult$occupation) adult$occupation <- gsub('Other-service', 'Service',
adult$occupation) adult$occupation <- gsub('Priv-house-serv', 'Service', adult$occupation)
adult$occupation <- gsub('Prof-specialty', 'Professional', adult$occupation) adult$occupation <-
gsub('Protective-serv', 'Service', adult$occupation) adult$occupation <- gsub('Tech-support',
'Service', adult$occupation) adult$occupation <- gsub('Transport-moving', 'Blue-Collar',
adult$occupation) adult$occupation <- gsub('Unknown', 'Other/Unknown', adult$occupation)
adult$occupation <- gsub('Armed-Forces', 'Other/Unknown', adult$occupation) adult$occupation
<- as.factor(adult$occupation)
```

```
summary(adult$occupation)
```

```
adult$marital_status <- gsub('Married-AF-spouse', 'Married', adult$marital_status)
adult$marital_status <- gsub('Married-civ-spouse', 'Married', adult$marital_status)
adult$marital_status <- gsub('Married-spouse-absent', 'Married', adult$marital_status)
adult$marital_status <- gsub('Never-married', 'Single', adult$marital_status) adult$marital_status
<- as.factor(adult$marital_status) summary(adult$marital_status)
```

```
sum(adult$capital_gain == 0)/length(adult$capital_gain) ## [1] 0.9167102
```

```
sum(adult$capital_gain == 0)/length(adult$capital_loss) ## [1] 0.9533491
```

```
adult$capital_gain <- NULL adult$capital_loss <- NULL adult$native_country <- NULL
```

```
j48tree<-J48(income~.,data=training_set,control=Weka_control(),options=NULL) j48tree
tree.predict48<- predict(j48tree, testing_set, type = "class")
confusionMatrix(testing_set$income, tree.predict48)
```

```
> rf.model<- randomforest(income~., data = training_set,importance=TRUE,keep.forest=TRUE, ntree = 2000)
> rf.predict <- predict(rf.model, testing_set)
> confusionMatrix(testing_set$income, rf.predict)
Confusion Matrix and Statistics
```

```
tree2 <- rpart(income ~ ., data = training_set, method = 'class', cp =
1e-2) tree2.pred <- predict(tree2, newdata = testing_set, type =
'class')
confusionMatrix(testing_set$income, tree2.pred) fancyRpartPlot(tree2)
```

CHAPTER –5

Testing

5.1 SYSTEM TESTING

Testing of Product:

System testing is the stage of implementation, which aimed at ensuring that system works accurately and efficiently before the live operation commence. Testing is the process of executing a program with the intent of finding an error. A good test case is one that has a high probability of finding an error. A successful test is one that answers a yet undiscovered error.

Testing is vital to the success of the system. System testing makes a logical assumption that if all parts of the system are correct, the goal will be successfully achieved. The candidate system is subject to variety of tests-on-line response, Volume Street, recovery and security and usability test. A series of tests are performed before the system is ready for the user acceptance testing. Any engineered product can be tested in one of the following ways. Knowing the specified function that a product has been designed to from, test can be conducted to demonstrate each function is fully operational. Knowing the internal working of a product, tests can be conducted to ensure that “al gears mesh”, that is the internal operation of the product performs according to the specification and all internal components have been adequately exercised.

5.1.1 UNIT TESTING:

Unit testing is the testing of each module and the integration of the overall system is done. Unit testing becomes verification efforts on the smallest unit of software design in the module. This is also known as ‘module testing’. The modules of the system are tested separately. This testing is carried out during the programming itself. In this testing step, each

model is found to be working satisfactorily as regard to the expected output from the module. There are some validation checks for the fields. For example, the validation check is done for verifying the data given by the user where both format and validity of the data entered is included. It is very easy to find error and debug the system.

5.1.2 INTEGRATION TESTING:

Data can be lost across an interface, one module can have an adverse effect on the other sub function, when combined, may not produce the desired major function. Integrated testing is systematic testing that can be done with sample data. The need for the integrated test is to find the overall system performance. There are two types of integration testing. They are:

- i) Top-down integration testing.
- ii) Bottom-up integration testing.

5.1.3 WHITE BOX TESTING:

White Box testing is a test case design method that uses the control structure of the procedural design to drive cases. Using the white box testing methods, we derived test cases that guarantee that all independent paths within a module have been exercised at least once.

5.1.4 BLACK BOX TESTING:

- Black box testing is done to find incorrect or missing function
- Interface error
- Errors in external database access
- Performance errors
- Initialization and termination errors

In 'functional testing', is performed to validate an application conforms to its specifications of correctly performs all its required functions. So this testing is also called 'black box testing'. It tests the external behavior of the system. Here the engineered product can be tested knowing the

specified function that a product has been designed to perform, tests can be conducted to demonstrate that each function is fully operational.

5.1.5 VALIDATION TESTING:

After the culmination of black box testing, software is completed assembly as a package, interfacing errors have been uncovered and corrected and final series of software validation tests begin validation testing can be defined as many, but a single definition is that validation succeeds when the software functions in a manner that can be reasonably expected by the customer.

5.1.6 USER ACCEPTANCE TESTING:

User acceptance of the system is the key factor for the success of the system. The system under consideration is tested for user acceptance by constantly keeping in touch with prospective system at the time of developing changes whenever required.

5.1.7 R Unit Testing

The RUnit approach is more in line with the above listed requirements and can be seen as a complement to the existing process in that:

- test cases are called and executed from the R prompt
- the programmer decides which result or functionality to put under testing, e.g. formatting issues of textual output do not need to matter
- test and reference data files need not be maintained separately but are combined into one file
- test cases need not be limited to testing/using functionality from one package checked at a time

The basic idea of this component is to execute a set of test functions defined through naming conventions, store whether or not the test succeeded in a central logger object and finally write a test protocol that allows to precisely identify the problems.

Implementation

Fibo.R:

```
Fibonacci <- function(n){  
  a <- 0  
  b <- 1  
  for (i in 1:n){  
    temp <- b  
    b <- a  
    a <- a + temp  
  }  
  return(a)  
}
```

test_fibo.R :

```
test_that("Test Fibo(15)",{  
  phi <- (1 + sqrt(5))/2  
  psi <- (1 - sqrt(5))/2  
  expect_equal(Fibonacci(15), (phi**15 - psi**15)/sqrt(5))  
})
```

Run_tests.R

```
library(testthat)  
  
source("path/to/fibo.R")  
  
test_results <- test_dir("path/to/tests", reporter="summary")
```

```
v | OK F W S | Context  
- | 1      | 0  
  
== Results ==  
OK:          1  
Failed:      0  
Warnings:    0  
Skipped:     0  
>  
> test_results  
      file context      test nb failed skipped error warning  user system  real  
1 test_fibo.R    Test Fibo(15) 1      0  FALSE FALSE      0 0.006 0.001 0.007
```

Figure3: RUnit Testing Output

CHAPTER – 6 RESULTS AND DISCUSSIONS

6.1 DATA VISUALIZATION

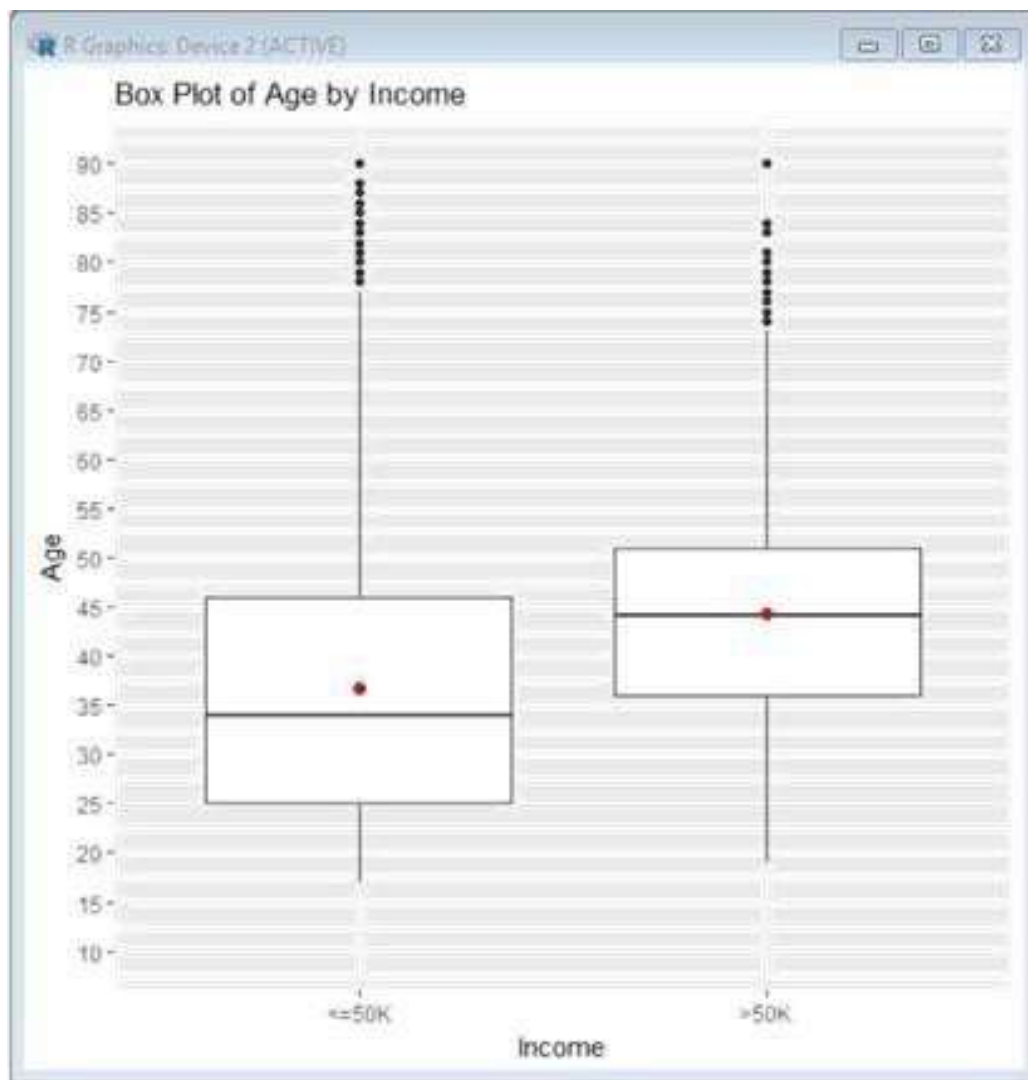


Figure4: Box Plot of Age By Income

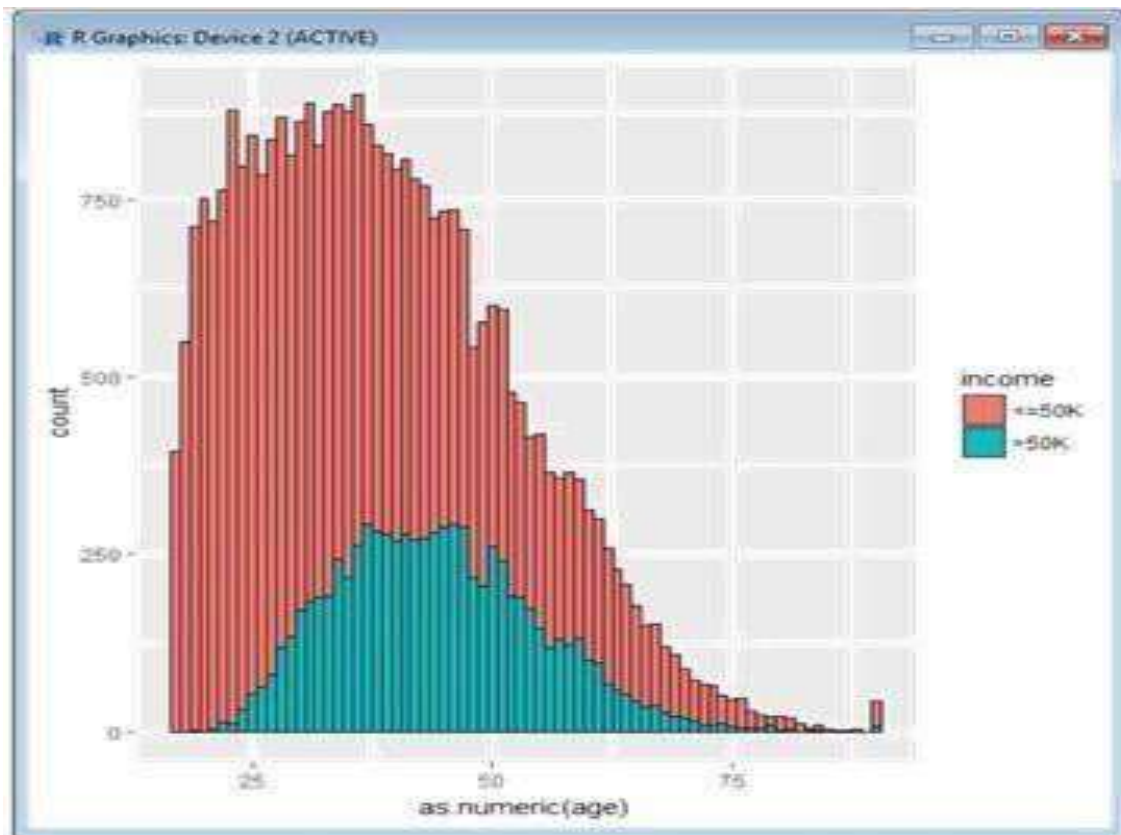


Figure5: Count vs Numeric

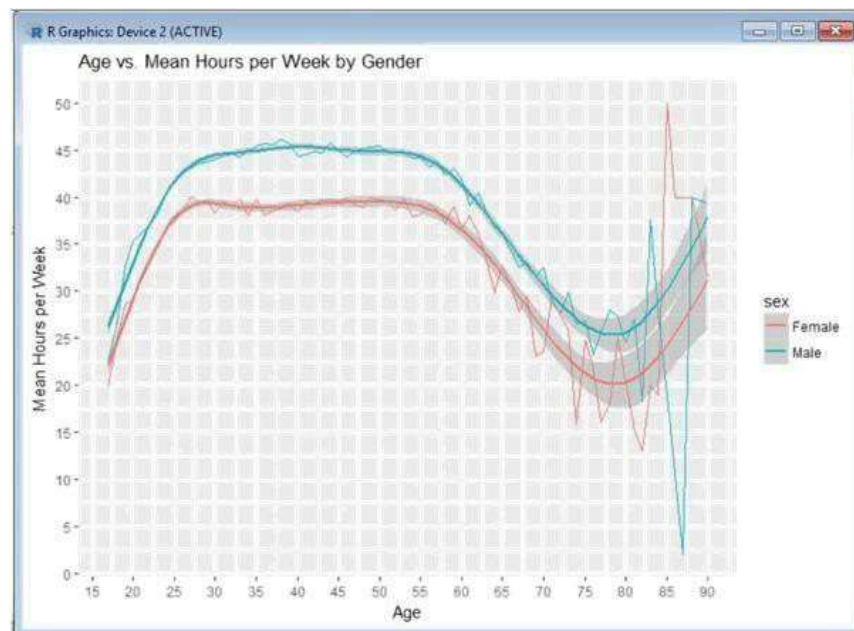


Figure6: Age vs Mean Hours per Week by Gender

6.2 RESULT OF CART DECISION TREE

Figure7: CART Decision Output 1

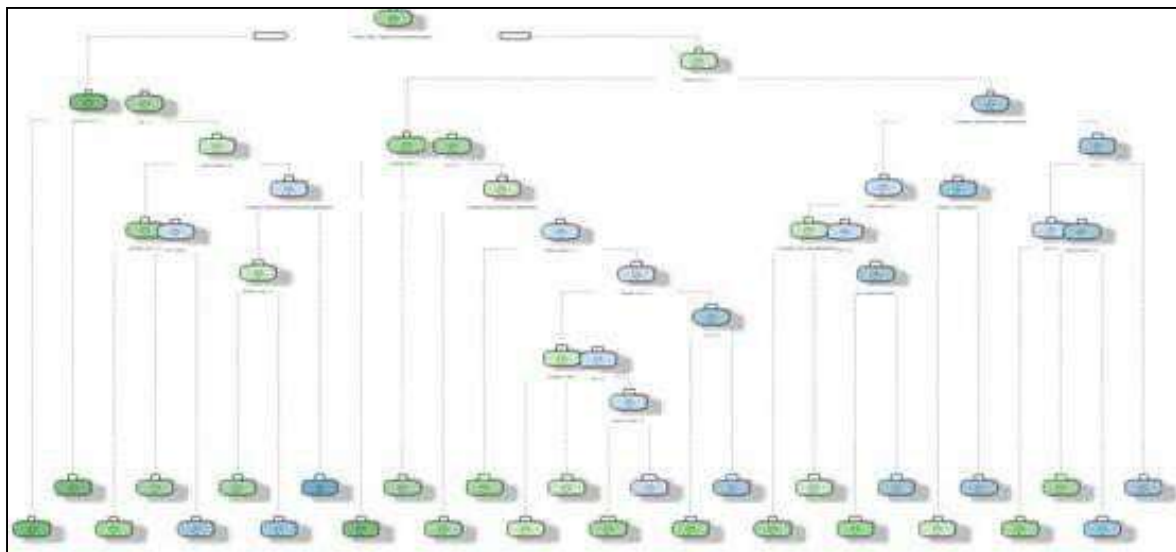
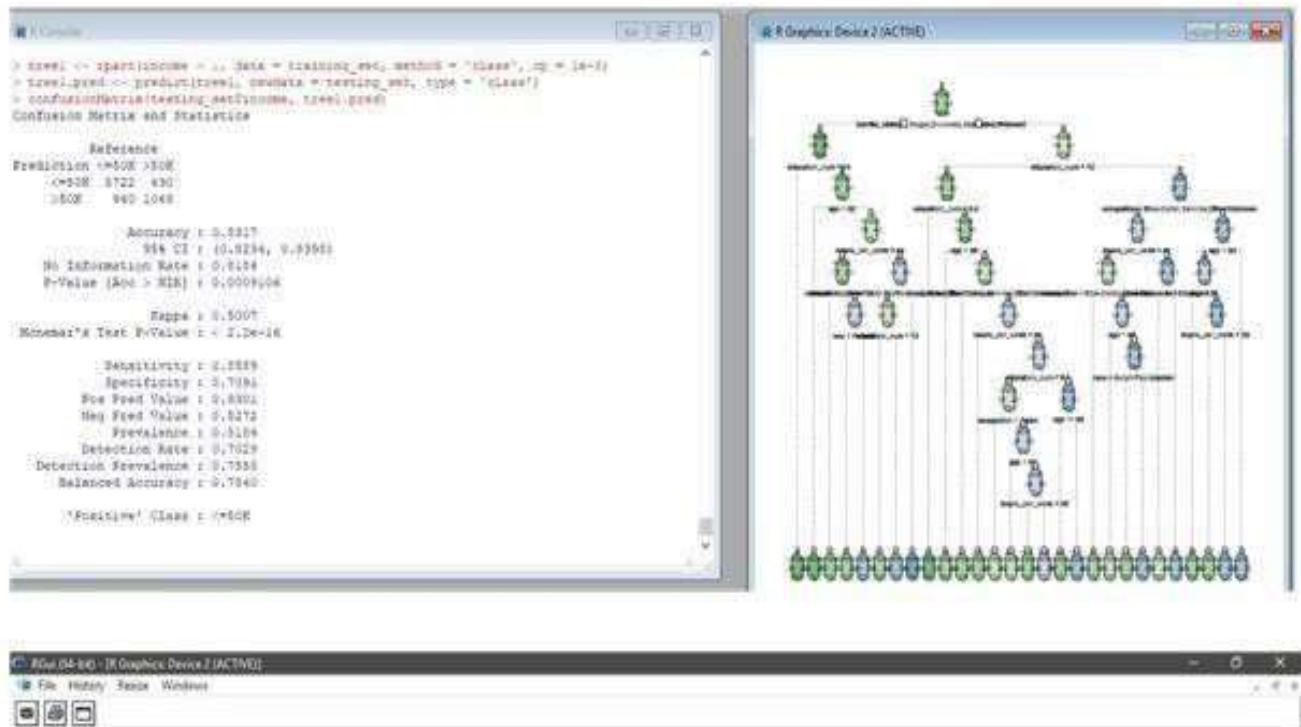


Figure8: CART Decision Output 2

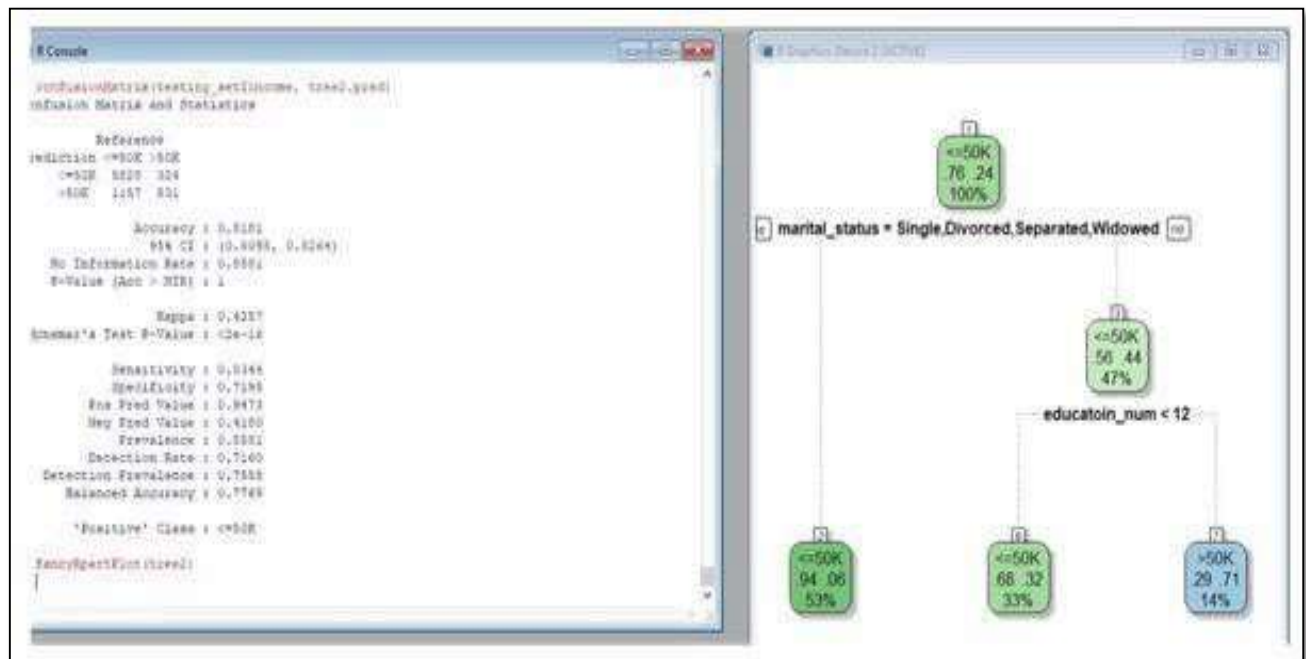


Figure9: CART Decision Output 3

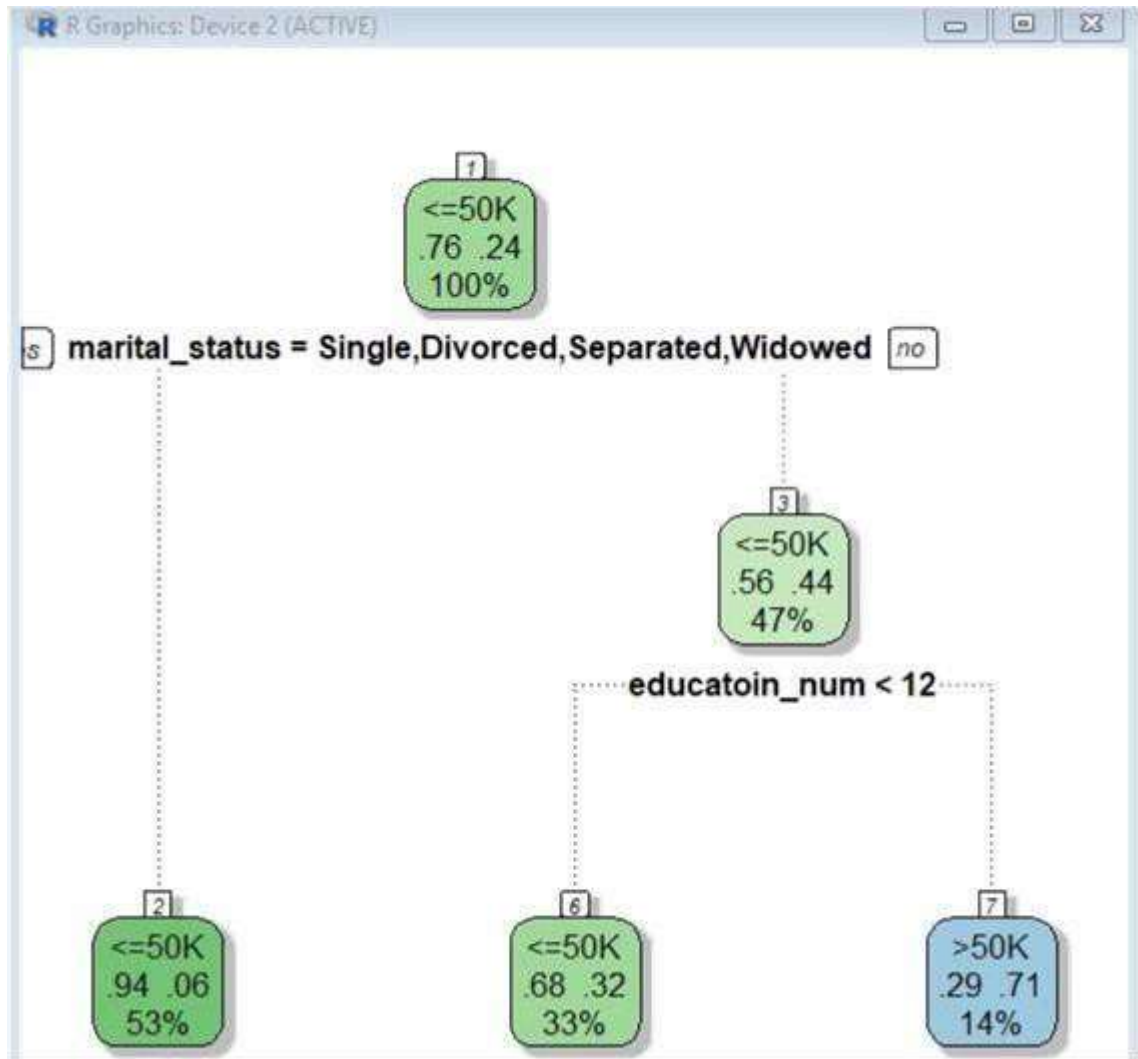


Figure10: CART Decision Output 4

```

> confusionMatrix(testing_set$income, tree1.pred)
Confusion Matrix and Statistics

          Reference
Prediction <=50K >50K
   <=50K   5722   430
   >50K     940  1048

      Accuracy : 0.8317
      95% CI   : (0.8234, 0.8398)
    No Information Rate : 0.8184
    P-Value [Acc > NIR] : 0.0009106

      Kappa : 0.5007
  McNemar's Test P-Value : < 2.2e-16

      Sensitivity : 0.8589
      Specificity : 0.7091
    Pos Pred Value : 0.9301
    Neg Pred Value : 0.5272
      Prevalence : 0.8184
    Detection Rate : 0.7029
    Detection Prevalence : 0.7558
    Balanced Accuracy : 0.7840

'Positive' Class : <=50K

```

Figure11: CART Decision Output 5

```

> confusionMatrix(testing_set$income, tree2.pred)
Confusion Matrix and Statistics

          Reference
Prediction <=50K >50K
   <=50K   5828   324
   >50K   1157   831

      Accuracy : 0.8181
      95% CI   : (0.8095, 0.8264)
    No Information Rate : 0.8581
    P-Value [Acc > NIR] : 1

      Kappa : 0.4257
  McNemar's Test P-Value : <2e-16

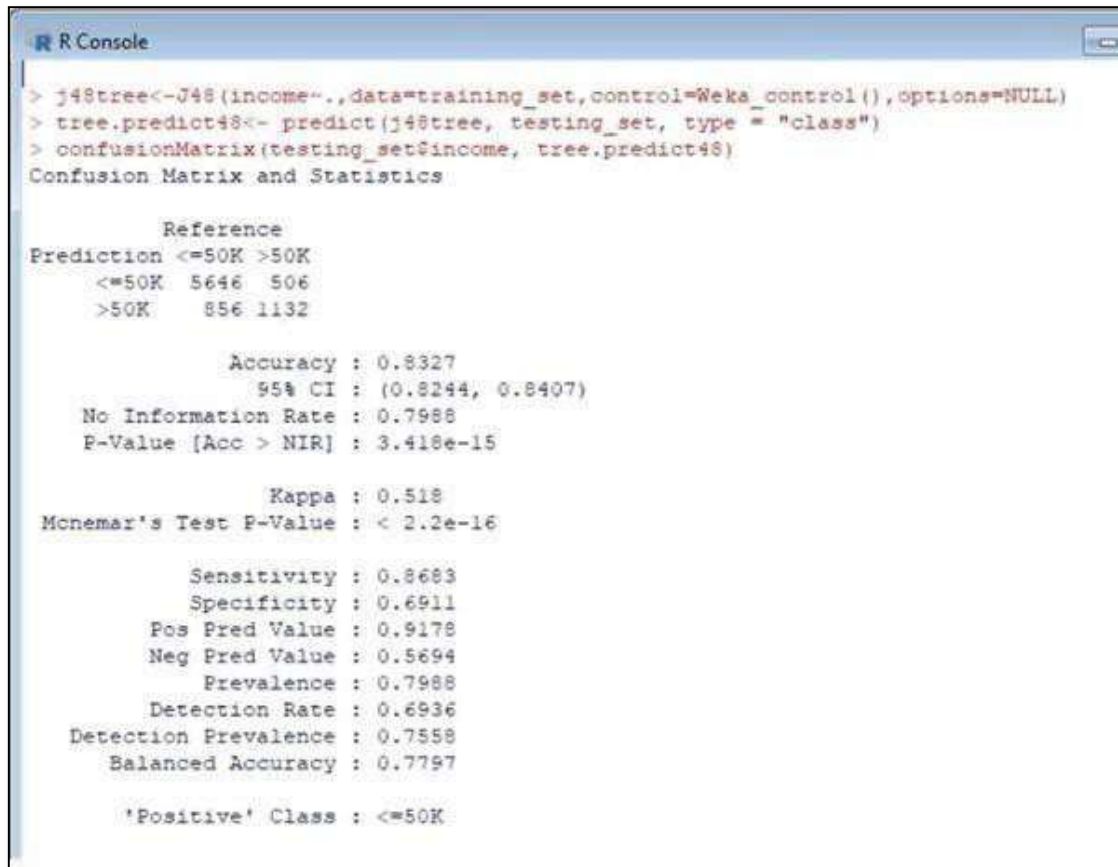
      Sensitivity : 0.8344
      Specificity : 0.7195
    Pos Pred Value : 0.9473
    Neg Pred Value : 0.4180
      Prevalence : 0.8581
    Detection Rate : 0.7160
    Detection Prevalence : 0.7558
    Balanced Accuracy : 0.7769

'Positive' Class : <=50K

```

Figure12: CART Decision Osnutput 6

6.3 RESULT OF J48 DESCION TREE

The image shows a screenshot of an R Console window. The title bar reads "R Console". The console contains several lines of R code and their corresponding output. The code defines a J48 tree, predicts on a testing set, and prints the confusion matrix and various performance statistics. The output includes a confusion matrix, accuracy, 95% confidence interval, No Information Rate, P-value, Kappa, McNemar's Test P-value, Sensitivity, Specificity, Positive and Negative Predictive Values, Prevalence, Detection Rate, Detection Prevalence, Balanced Accuracy, and the positive class label.

```
> j48tree<-J48(income~.,data=training_set,control=Weka_control(),options=NULL)
> tree.predict48<- predict(j48tree, testing_set, type = "class")
> confusionMatrix(testing_set$income, tree.predict48)
Confusion Matrix and Statistics

              Reference
Prediction <=50K >50K
   <=50K    5646    506
   >50K      856   1132

      Accuracy : 0.8327
      95% CI   : (0.8244, 0.8407)
    No Information Rate : 0.7988
    P-Value [Acc > NIR] : 3.418e-15

      Kappa : 0.518
  McNemar's Test P-Value : < 2.2e-16

    Sensitivity : 0.8683
    Specificity : 0.6911
   Pos Pred Value : 0.9178
   Neg Pred Value : 0.5694
    Prevalence : 0.7988
    Detection Rate : 0.6936
    Detection Prevalence : 0.7558
    Balanced Accuracy : 0.7797

    'Positive' Class : <=50K
```

Figure13: J48 Decision Tree Output 1

```

> confusionMatrix(testing_set$income, tree.predict48)
Confusion Matrix and Statistics

      Reference
Prediction <=50K >50K
 <=50K      5646   506
 >50K       856 1132

      Accuracy : 0.8327
      95% CI   : (0.8244, 0.8407)
    No Information Rate : 0.7988
    P-Value [Acc > NIR] : 3.418e-15

      Kappa : 0.518
  McNemar's Test P-Value : < 2.2e-16

      Sensitivity : 0.8683
      Specificity : 0.6911
    Pos Pred Value : 0.9178
    Neg Pred Value : 0.5694
      Prevalence : 0.7988
    Detection Rate : 0.6936
    Detection Prevalence : 0.7558
    Balanced Accuracy : 0.7797

      'Positive' Class : <=50K

```

Figure14: J48 Decision Tree Output 2

6.4 RESULT OF RANDOM FOREST ALGORITHM

```
> confusionMatrix(testing_set$income, rf.predict)
Confusion Matrix and Statistics

          Reference
Prediction <=50K >50K
   <=50K   5715  437
   >50K    890 1098

      Accuracy : 0.837
      95% CI   : (0.8288, 0.8449)
    No Information Rate : 0.8114
    P-Value [Acc > NIR] : 1.081e-09

      Kappa   : 0.5215
  Mcnemar's Test P-Value : < 2.2e-16

    Sensitivity : 0.8653
    Specificity : 0.7153
   Pos Pred Value : 0.9290
   Neg Pred Value : 0.5523
    Prevalence : 0.8114
    Detection Rate : 0.7021
    Detection Prevalence : 0.7558
    Balanced Accuracy : 0.7903

    'Positive' Class : <=50K
```

Figure15: Random Forest Output

6.5 COMPARISION OF ALGORITHMS

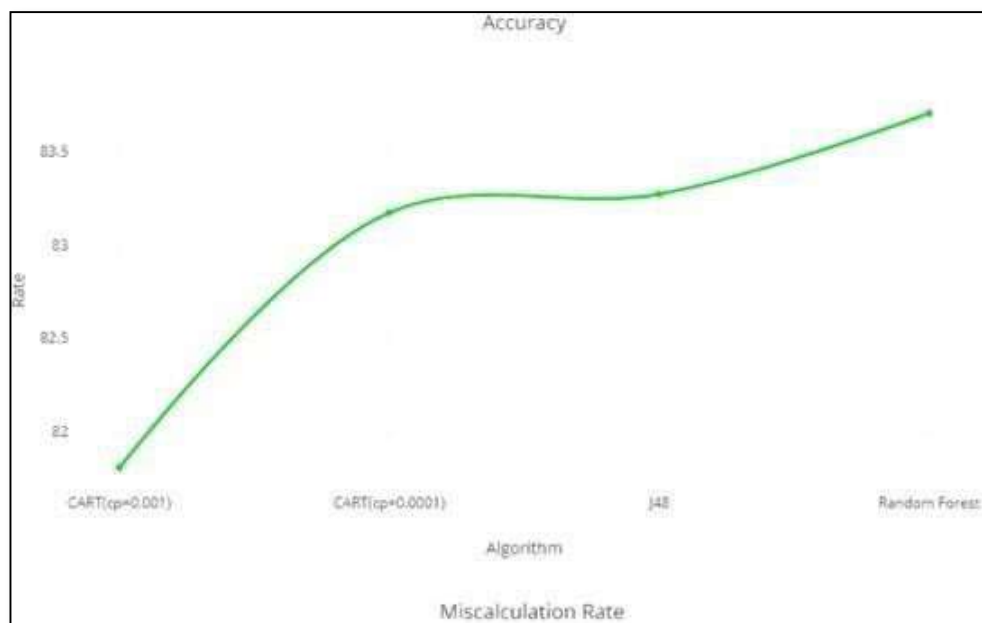


Figure16: Comparison of Algorithms 1

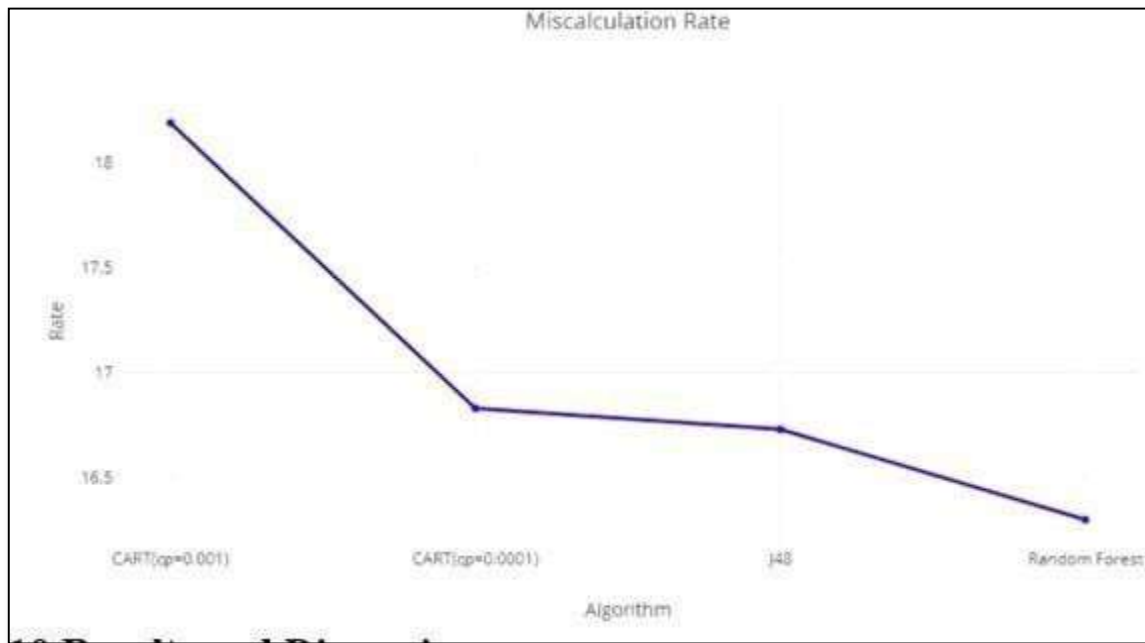


Figure17: Comparison of Algorithms 2

Algorithm	Accuracy	Miscalculation rate	Sensitivity	Specificity
CART(cp=0.001)	81.81%	18.19%	0.8344	0.7195
CART(cp=0.0001)	83.17%	16.83%	0.8589	0.7091
J48	83.27%	16.73%	0.8683	0.6911
Random Forest	83.7%	16.3%	0.8653	0.7153

Table 3: Results and Observations

CHAPTER 7 – CONCLUSION

Conclusion

In this study, we aimed to predict a person's income based on variables like habitat, education, marital status, age, race, sex and others. We found in exploring this particular dataset that, higher education is no guarantee to high income. This pattern could be attributed the uneven sample distribution. Several classification models were tested for prediction accuracy and we determined that the Random Forest model (83.7%) gives the highest accuracy among others.

As a future work, we will extend this study to include feature engineering methods, to measure if the predictive power of the models could be increased or not

REFERENCES

1. Bekena, S. M. (2017). Using decision tree classifier to predict income levels.
2. Khongchai, P., & Songmuang, P. (2016, November). Random Forest for Salary Prediction System to Improve Students' Motivation. In 2016 12th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS) (pp. 637-642). IEEE.
3. Chakrabarty, N., & Biswas, S. (2018). A Statistical Approach to Adult Census Income Level Prediction. arXiv preprint arXiv:1810.10076.
4. Mandot, M., & Soni, R. (2012). Decision Tree of Behavioral Model for Income of Indian Adults at USA. *International Journal of Advanced Research in Computer Science*, 3(3).
5. Ishibashi, K., Iwasaki, T., Otomasa, S., & Yada, K. (2016). Model selection for financial statement analysis: Variable selection with data mining technique. *Procedia Computer Science*, 96, 1681-1690.
6. Bollen, K. A., Glanville, J. L., & Stecklov, G. (2007). Socio-economic status, permanent income, and fertility: A latent-variable approach. *Population studies*, 61(1), 15-34.
7. Singh, S., Mittal, H., & Purwar, A. (2014). Prediction of investment patterns using data mining techniques. *International Journal of Computer and Communication Engineering*, 3(2), 145.
8. Soumya, S. B., & Deepika, N. (2016). Data Mining With Predictive Analytics for Financial Applications. *International Journal of Scientific Engineering and Applied Science*, 1, 310-317.
9. Koskinen, L., Nurminen, T., & Salonen, J. (2005). Modelling and predicting individual salaries: a study of Finlands unique dataset. *Eläketurvakeskus*.
10. Anderson, R. E., & Darkenwald, G. G. (1979). Participation and Persistence in American Adult Education. Implications for Public Policy and Future Research from a Multivariate Analysis of a National Data Base. *Direction Papers in Lifelong Learning*.
11. Stojanova, D., Panov, P., Kobler, A., Dzeroski, S., & Taskova, K. (2006, October). Learning to predict loan with different data mining techniques. In *Conference on data mining and data warehouses (SiKDD 2006), Ljubljana, Slovenia* (pp. 255-258).
12. Karouni, A., Daya, B., & Chauvet, P. (2014). APPLYING DECISION TREE ALGORITHM AND NEURAL NETWORKS TO PREDICT OCCURRENCE OF LOAN IN LEBANON. *Journal of Theoretical & Applied Information Technology*, 63(2).
13. Chakrabarty, N., & Biswas, S. (2018). A Statistical Approach to Adult Census Income Level Prediction. arXiv preprint arXiv:1810.10076.

14. Bekena, S. M. (2017). Using decision tree classifier to predict income levels.
15. Sharath, R., Nirupam, K. N., Sowmya, B. J., & Srinivasa, K. G. (2016, October). Data analytics to predict the income and economic hierarchy on census data. In 2016 International Conference on Computation System and Information Technology for Sustainable Solutions (CSITSS) (pp. 249-254). IEEE.
16. Chet Lemon, Chris Zelazo and Kesav Mulakaluri: "Predicting if income exceeds \$50,000 per year based on 1994 US Census Data with Simple Classification Techniques
17. Chakrabarty, N., & Biswas, S. (2018). A Statistical Approach to Adult Census Income Level Prediction. arXiv preprint arXiv:1810.10076.
18. Rashid, N. K. A., Nasir, A., Mustapha, N. H. N., & Kamil, N. F. (2011). Analysis of income and expenditure of households in the east coast of Peninsular Malaysia. *Journal of Global Business and Economics*, 2(1), 59-72.
19. Chockalingam, V., Shah, S., & Shaw, R. Income Classification using Adult Census Data
20. Sharath, R., Nirupam, K. N., Sowmya, B. J., & Srinivasa, K. G. (2016, October). Data analytics to predict the income and economic hierarchy on census data. In *2016 International Conference on Computation System and Information Technology for Sustainable Solutions (CSITSS)* (pp. 249-254). IEEE.

DIGITAL ASSIGNMENT 2

Tool: Selenium Tool

Selenium is a portable framework for testing web applications. Selenium was originally developed by Jason Huggins in 2004 as an internal tool at ThoughtWorks. Huggins was later joined by other programmers and testers at ThoughtWorks, before Paul Hammant joined the team and steered the development of the second mode of operation that would later become "Selenium Remote Control" (RC). In 2008, Philippe Hanrigou (then at ThoughtWorks) made "Selenium Grid", which provides a hub allowing the running of multiple Selenium tests concurrently on any number of local or remote systems, thus minimizing test execution time. Grid offered, as open source, a similar capability to the internal/private Google cloud for Selenium RC. Pat Lightbody had already made a private cloud for "HostedQA" which he went on to sell to Gomez, Inc.

The name Selenium comes from a joke made by Huggins in an email, mocking a competitor named Mercury, saying that you can cure mercury poisoning by taking selenium supplements. The others that received the email took the name and ran with it.

Selenium is a portable framework for testing web applications. Selenium provides a playback tool for authoring functional tests without the need to learn a test scripting language (Selenium IDE). It also provides a test domain-specific language (Selenese) to write tests in a number of popular programming languages, including C#, Groovy, Java, Perl, PHP, Python, Ruby and Scala. The tests can then run against most modern web browsers. Selenium runs on Windows, Linux, and macOS. It is open-source software released under the Apache License 2.0.

Selenium Integrated Development Environment (IDE) is the simplest framework in the Selenium suite and is the easiest one to learn. It is a Firefox plugin that you can install as easily as you can with other plugins. However, because of its simplicity, Selenium IDE should only be used as a prototyping tool. If you want to create more advanced test cases, you will need to use either Selenium RC or WebDriver.

Application Names

- 1) Selenium WebDriver: Selenium WebDriver drives a browser natively, as a real user would, either locally or on remote machines
- 2) Selenium IDE: Selenium IDE is a Chrome and Firefox extension that makes it easy to record and playback tests in the browser.
- 3) Selenium Grid: Selenium Grid takes WebDriver to another level by running tests on many machines at the same time, cutting down on the time it takes to test on multiple browsers and operating systems.

Sample Demo

Demo1: Login Page

```
import unittest
import HtmlTestRunner
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
import time

class Login(unittest.TestCase):
    def setup(self):
        self.driver = webdriver.Firefox()
    def test_login_1(self):
        self.driver = webdriver.Firefox()
        self.driver.get("http://localhost/book_store/index.php")
        userName =
        self.driver.find_element_by_xpath("/html/body/div[3]/div[2]/ul/li[1]/form
        /input[1]")
        userName.send_keys("Hiren")
        password =
        self.driver.find_element_by_xpath("/html/body/div[3]/div[2]/ul/li[1]/form
        /input[2]")
        password.send_keys("hiru")
        self.driver.implicitly_wait(500)
        btn = self.driver.find_element_by_xpath('//*[@id="x"]')
        btn.submit()
        time.sleep(1)
        architecture =
        self.driver.find_element_by_xpath('//*[@id="menu"]/ul/li[4]/a')
        architecture.click()
    def test_login_2(self):
        self.driver = webdriver.Firefox()
        self.driver.get("http://localhost/book_store/index.php")
        userName =
        self.driver.find_element_by_xpath("/html/body/div[3]/div[2]/ul/li[1]/form
```

```

/input[1]")
userName.send_keys("shital")
password =
self.driver.find_element_by_xpath("/html/body/div[3]/div[2]/ul/li[1]/form
/input[2]")
password.send_keys("shital")
self.driver.implicitly_wait(500)
btn = self.driver.find_element_by_xpath('//*[@id="x"]')
btn.submit()
time.sleep(1)
architecture =
self.driver.find_element_by_xpath('//*[@id="sidebar"]/ul/li[3]/ul/li[1]/a')
architecture.click()
def test_login_3(self):
self.driver = webdriver.Firefox()
self.driver.get("http://localhost/book_store/index.php")
userName =
self.driver.find_element_by_xpath("/html/body/div[3]/div[2]/ul/li[1]/form
/input[1]")
userName.send_keys("shiefal")
password =
self.driver.find_element_by_xpath("/html/body/div[3]/div[2]/ul/li[1]/form
/input[2]")
password.send_keys("shiefefal")
self.driver.implicitly_wait(500)
btn = self.driver.find_element_by_xpath('//*[@id="x"]')
btn.submit()
time.sleep(1)
architecture =
self.driver.find_element_by_xpath('//*[@id="sidebar"]/ul/li[3]/ul/li[1]/a')
architecture.click()
if __name__=="main_":
unittest.main(testRunner=HtmlTestRunner.HTMLTestRunner(output
='reports'))

```


Output



Unittest Results

Start Time: 2020-04-06 11:11:45

Duration: 16.84 s

Summary: Total: 3, Pass: 2, Error: 1

__main__.Login	Status
test_login_1	Pass
test_login_2	Pass
test_login_3	Error View

Total: 3, Pass: 2, Error: 1 -- Duration: 16.84 s

Demo 2: Contact Us Page

```
import unittest
import HtmlTestRunner
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
import time
class Contact(unittest.TestCase):
    def setup(self):
        self.driver = webdriver.Firefox()
    def test_contact_1(self):
        self.driver = webdriver.Firefox()
        self.driver.get("http://localhost/book_store/index.php")
        contactMenu =
        self.driver.find_element_by_xpath('//*[@@id="menu"]/ul/li[3]/a')
        contactMenu.click()
        time.sleep(1)
        name =
        self.driver.find_element_by_xpath('//*[@@id="content"]/div/div/form/input
[1]')
        name.send_keys("Satin")
        email =
        self.driver.find_element_by_xpath('//*[@@id="content"]/div/div/form/input
```

```

[2]')
email.send_keys("satin@gmail.com")
query =
self.driver.find_element_by_xpath('//*[@id="content"]/div/div/form/texta
rea')
query.send_keys("This is some Query!!!")
self.driver.implicitly_wait(500)
btn =
self.driver.find_element_by_xpath('//*[@id="content"]/div/div/form/input
[3]')
btn.submit()
def test_contact_2(self):
self.driver = webdriver.Firefox()
self.driver.get("http://localhost/book_store/index.php")
contactMenu =
self.driver.find_element_by_xpath('//*[@id="menu"]/ul/li[3]/a')
contactMenu.click()
time.sleep(1)
name =
self.driver.find_element_by_xpath('//*[@id="content"]/div/div/form/input
[1]')
name.send_keys("sankalp")
email =
self.driver.find_element_by_xpath('//*[@id="content"]/div/div/form/input
[2]')
email.send_keys("sankalp@gmail.com")
query =
self.driver.find_element_by_xpath('//*[@id="content"]/div/div/form/texta
rea')
query.send_keys("This is some Query!!!")
self.driver.implicitly_wait(500)
btn =
self.driver.find_element_by_xpath('//*[@id="content"]/div/div/form/input
[3]')
btn.submit()
def test_contact_3(self):
self.driver = webdriver.Firefox()
self.driver.get("http://localhost/book_store/index.php")
contactMenu =
self.driver.find_element_by_xpath('//*[@id="menu"]/ul/li[3]/a')
contactMenu.click()

```

```

time.sleep(1)
email =
self.driver.find_element_by_xpath('//*[@@id="content"]/div/div/form/input
[2]')
email.send_keys("sankalp@gmail.com")
self.driver.implicitly_wait(500)
btn = self.driver.find_element_by_xpath('//*[@@id="x"]')
btn.submit()
time.sleep(1)
architecture =
self.driver.find_element_by_xpath('//*[@@id="sidebar"]/ul/li[3]/ul/li[1]/a')
architecture.click()
if __name__=="__main__":
unittest.main(testRunner=HtmlTestRunner.HTMLTestRunner(output
='reports'))

```

Output

Apps Vellore Institute of...

Unittest Results

Start Time: 2020-06-04 11:57:25

Duration: 42.42 s

Summary: Total: 3, Pass: 2, Error: 1

__main__.Contact	Status
test_contact_1	Pass
test_contact_2	Pass
test_contact_3	Error View

Total: 3, Pass: 2, Error: 1 -- Duration: 42.42 s