Introduction:

Data mining is a valuable tool to analyze large amounts of data in health insurance companies and make informed decisions. It involves extracting meaningful patterns, trends, and insights from complex data sets. In a sector like health insurance, data mining is used for various purposes such as assessing risk, detecting fraud, analyzing claims, and segmenting customers. Risk assessment involves analyzing data related to a person's medical history, lifestyle, age, and other factors to determine appropriate premium rates and coverage options. Data mining is also used for fraud detection, identifying patterns of fraudulent activities like false claims or billing irregularities. Claims analysis involves optimizing claims processes, identifying potentially fraudulent claims, and improving operational efficiency. Additionally, data mining is used for customer segmentation, where data related to customers is analyzed to group them based on factors like age, gender, location, and health conditions for targeted marketing campaigns and personalized services.

Step 1: Data Source:

The source of the data for this health insurance analysis project is:
https://www.kaggle.com/mirichoi0218/insurance

Step2: Load the dataset and store it into a local R variable named.

```
> # STEP 2:Load the dataset
> insurance = read.csv("/Users/nirajkc/Downloads/insurance.csv")
>
>
```

The insurance dataset is loaded in the object names insurance in the Rstudio.

Step3: Explore your dataset.

```
> head(insurance)
  age    sex    bmi children smoker    region   charges
1  19 female 27.900        0    yes southwest 16884.924
2  18   male 33.770        1     no southeast  1725.552
3  28   male 33.000        3     no southeast  4449.462
4  33   male 22.705        0     no northwest 21984.471
5  32   male 28.880        0     no northwest  3866.855
6  31 female 25.740        0     no southeast  3756.622
```

The insurance dataset contains 7 variables:

Age: Refers to the age of the main policyholder.

Sex: Refers to the gender of the person who is taking the insurance policy - can be male or female.

BMI: The Body Mass Index is an indicator of body weight relative to height, which gives an idea of whether a person's weight is relatively high or low. The ideal range for BMI is between 18.5 and 24.9 kg/m².

Children: Refers to the number of dependents or children covered under the health insurance policy.

Smoker: Refers to whether the policyholder is a smoker or not.

Region: Refers to the geographical region where the policyholder resides in the United States. The regions are Northeast, Southeast, Southwest, and Northwest.

Charges: Refers to the medical expenses charged by the health insurance company for each individual policyholder.

```
> dim(insurance)
[1] 1338    7
>
```

The dataset contains 7 variables and 1338 rows or observations.

```
> summary(insurance)
      age            sex                 bmi           children        smoker             region             charges
 Min.   :18.00   Length:1338        Min.   :15.96   Min.   :0.000   Length:1338        Length:1338        Min.   : 1122
 1st Qu.:27.00   Class :character   1st Qu.:26.30   1st Qu.:0.000   Class :character   Class :character   1st Qu.: 4740
 Median :39.00   Mode  :character   Median :30.40   Median :1.000   Mode  :character   Mode  :character   Median : 9382
 Mean   :39.21                      Mean   :30.66   Mean   :1.095                                         Mean   :13270
 3rd Qu.:51.00                      3rd Qu.:34.69   3rd Qu.:2.000                                         3rd Qu.:16640
 Max.   :64.00                      Max.   :53.13   Max.   :5.000                                         Max.   :63770
>
```

```
> str(insurance)
'data.frame':   1338 obs. of  7 variables:
 $ age     : int  19 18 28 33 32 31 46 37 37 60 ...
 $ sex     : chr  "female" "male" "male" "male" ...
 $ bmi     : num  27.9 33.8 33 22.7 28.9 ...
 $ children: int  0 1 3 0 0 0 1 3 2 0 ...
 $ smoker  : chr  "yes" "no" "no" "no" ...
 $ region  : chr  "southwest" "southeast" "southeast" "northwest" ...
 $ charges : num  16885 1726 4449 21984 3867 ...
>
```

As seen above, the variables age, children are integer data types, where as bmi, charges are numerical and sex, smoker, and region are character data types.

Step4: Perform Summary Statistics.

```
> # For numerical variables
> summary(insurance[c("age", "bmi", "children", "charges")])
      age             bmi           children        charges
 Min.   :18.00   Min.   :15.96   Min.   :0.000   Min.   : 1122
 1st Qu.:27.00   1st Qu.:26.30   1st Qu.:0.000   1st Qu.: 4740
 Median :39.00   Median :30.40   Median :1.000   Median : 9382
 Mean   :39.21   Mean   :30.66   Mean   :1.095   Mean   :13270
 3rd Qu.:51.00   3rd Qu.:34.69   3rd Qu.:2.000   3rd Qu.:16640
 Max.   :64.00   Max.   :53.13   Max.   :5.000   Max.   :63770
>
```

There are four numeric variables as shown above. There are no much differences between the mean and median for the variables age, bmi, and children whereas for the charges variable mean is greater than the median means it is right skewed.

```
> # For categorical variables
> table(insurance$sex)

female   male
   662    676
> table(insurance$smoker)

  no  yes
1064  274
> table(insurance$region)

northeast northwest southeast southwest
      324       325       364       325
>
```

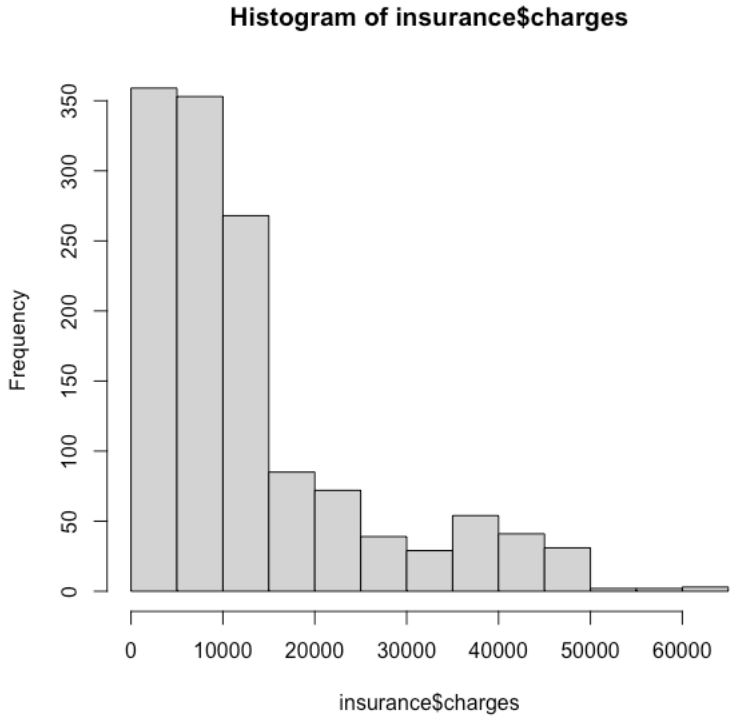
For the categorical variables like sex, smoker, and region, the distribution of these variables are shown above by using the table() function. The distribution of values of the variables sex and region are equally distributed whereas, in the smoker variables, the number of people who do not smoke is 1064 whereas smoker is only 274.

Step 5: Visualization.

➢Show the distribution of the charges, and give your analysis.

```
> summary(insurance$charges)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   1122    4740    9382   13270   16640   63770
>
```

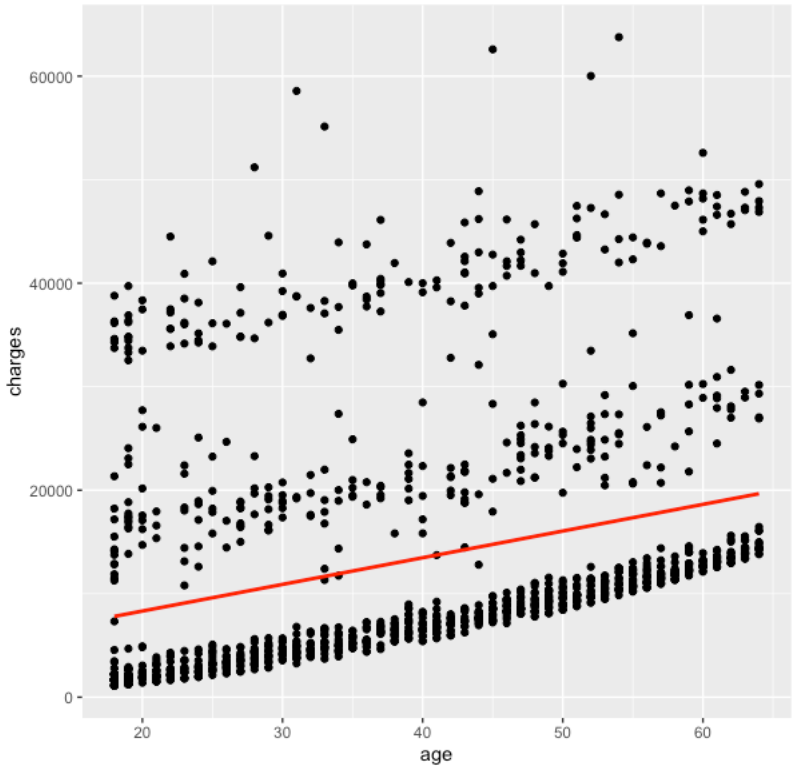
The summary statistics reveal that the insurance charges dataset has a minimum charge of 1,122 USD, a median charge of 9,382 USD, and a maximum charge of 63,770 USD. The mean charge is 13,270 USD, which is greater than the median charge, indicating a positively skewed distribution of charges.



The visualization of the charges distribution in the insurance dataset indicates that the data is positively skewed or right-skewed. This is due to the longer tail on the right side of the distribution, as shown in the histogram. A right-skewed distribution implies that the mean is greater than the median, indicating that there are more data points on the right side of the distribution. In the case of insurance charges, most policyholders have charges below $20,000, with a few policyholders having much higher charges, resulting in the long tail on the right side.

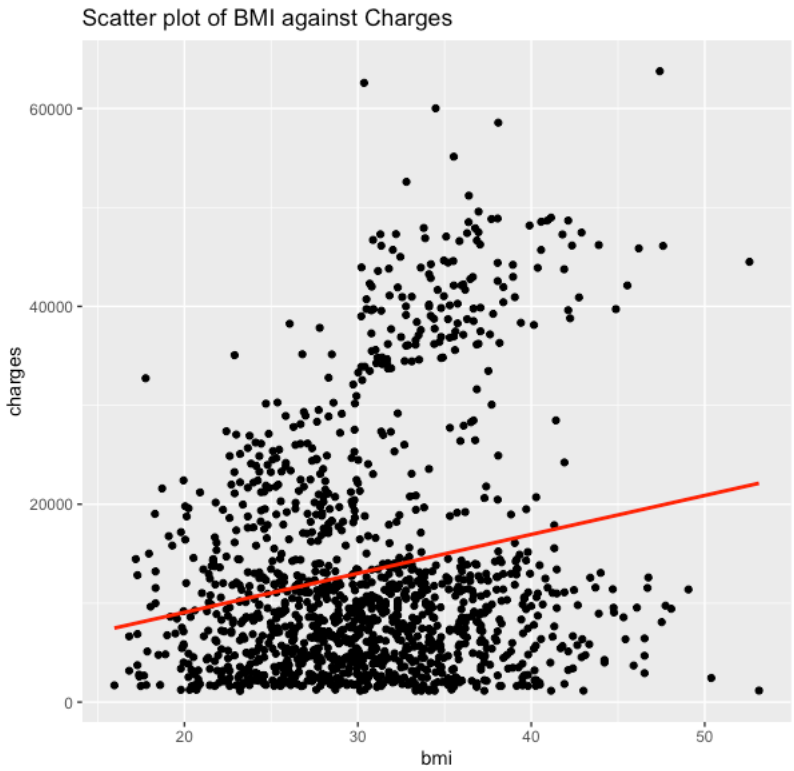
➢Show the effect of the other variables like (age and BMI on the charges and give your analysis)

```
> ggplot(insurance, aes(x = age, y = charges)) + geom_point()+
+ geom_smooth(method = "lm", se = FALSE, color = "red")
`geom_smooth()` using formula = 'y ~ x'
>
```



This ggplot(scatter plot) shows a positive correlation between charges and age, indicating that charges increase as age increases. Additionally, outliers are visible, possibly due to other variables such as pre-existing medical conditions.

```
> #insurance vs bmi
> library(ggplot2)
> ggplot(insurance, aes(x = bmi, y = charges)) +geom_point()+
+ ggtitle("Scatter plot of BMI against Charges")+
+ geom_smooth(method = "lm", se = FALSE, color = "red")
`geom_smooth()` using formula = 'y ~ x'
>
```



And, also we can see that there is a positive correlation between BMI and charges, meaning that as BMI increases, so do the charges. We can also see that there are some outliers with very high charges, which could be due to other factors such as lifestyle or pre-existing medical conditions.

Also, we can see the correction of these variables with the charges using cor() function. As we can see below, the correction between variable charges with age is more while "charges" has least correction with children.

```
> # correlation of age, bmi, children on charges
> cor(insurance[c("age", "bmi", "children", "charges")])
               age       bmi   children    charges
age      1.0000000 0.1092719 0.04246900 0.29900819
bmi      0.1092719 1.0000000 0.01275890 0.19834097
children 0.0424690 0.0127589 1.00000000 0.06799823
charges  0.2990082 0.1983410 0.06799823 1.00000000
>
```

#Step 6: Data Preprocessing:
Missing values.

One of the data preprocessing steps is checking the missing values in the dataset. In R, we can check the missing values using is.na() function and colSums() function will add all the values according to column wise.

```
> # 6.1 Missing values.
> colSums(is.na(insurance))
     age      sex      bmi children   smoker   region  charges
       0        0        0        0        0        0        0
>
```

There are no missing values in the dataset as seen above code execution.

Encoding Categorical Data.

Encoding categorical data is another data pre-processing step. Encoding categorical data is the process of converting categorical or qualitative variables into numerical or quantitative representations that can be used as input for the data mining algorithms or statistical analysis. There are several common methods for encoding categorical data, including: Label Encoding, One-Hot Encoding, Binary Encoding, Count Encoding, Target Encoding.

The encoding method that we are using in this project is One-hot encoding- It is used to convert categorical features into binary columns, with 1 indicating presence and 0 indicating absence. The caret package is imported, and dummyVars() is used to create dummy variables for categorical variables, with n-1 columns for n unique levels. The resulting transformed data confirms successful encoding.

```
> # one-hot encoding function
> library(caret)
> dmy <- dummyVars(" ~ .", data = insurance, fullRank = T)
> insurance_encoded <- data.frame(predict(dmy, newdata = insurance))
> head(insurance_encoded)
  age sexmale    bmi children smokeryes regionnorthwest regionsoutheast regionsouthwest   charges
1  19       0 27.900        0         1               0               0               1 16884.924
2  18       1 33.770        1         0               0               1               0  1725.552
3  28       1 33.000        3         0               0               1               0  4449.462
4  33       1 22.705        0         0               1               0               0 21984.471
5  32       1 28.880        0         0               1               0               0  3866.855
6  31       0 25.740        0         0               0               1               0  3756.622
>
```

For using categorical variables in machine learning algorithms or statistical models, we first need to convert them into numerical format through a process called encoding. Categorical variables contain labels or categories instead of numeric values, so they cannot be used directly for analysis. One popular encoding method is one-hot encoding, which creates binary variables for each category in the categorical variable. The binary variable takes the value 1 if the category is present in the observation and 0 otherwise. This allows for the categorical variable to be represented as a set of binary variables that can be used for analysis or modeling.

In the given code, the data.frame() function from the dummies library is used to perform one-hot encoding on the "sex", "smoker", and "region" columns of the "insurance" data frame. The names argument specifies the columns that need to be one-hot encoded.

#Step 7: Regression

Definition of Regression as a data mining techniques: Regression analysis is a data mining technique that helps us understand how different variables are related in a dataset. By looking at historical data, we can identify patterns and trends to make predictions about the future. Regression techniques establish a mathematical relationship between a dependent variable (the one we want to predict) and one or more independent variables (the ones we use for predictions). The dependent variable is usually a numeric value, like sales revenue or temperature, while the independent variables can be numeric or categorical. The goal is to find the best line or curve that fits the data well and minimizes prediction errors. There are two main types of regression: simple regression, which involves one independent variable, and multiple regression, which involves multiple independent variables.

1. Split data into Training and Test set

```
> # 7.1 split the dataset into training and testing sets
> set.seed(123)
> trainIndex <- createDataPartition(y = insurance_encoded$charges, p = 0.8, list = FALSE)
> train <- insurance_encoded[trainIndex, ]
> test <- insurance_encoded[-trainIndex, ]
> dim(train)
[1] 1072    9
> dim(test)
[1] 266   9
> head(train)
  age sexmale    bmi children smokeryes regionnorthwest regionsoutheast regionsouthwest   charges
1  19       0 27.900        0         1               0               0               1 16884.924
2  18       1 33.770        1         0               0               1               0  1725.552
3  28       1 33.000        3         0               0               1               0  4449.462
4  33       1 22.705        0         0               1               0               0 21984.471
6  31       0 25.740        0         0               0               1               0  3756.622
7  46       0 33.440        1         0               0               1               0  8240.590
>
```

For the reproducibility of results, we are using set.seed() function and then creating a data partition using the createDataPartition() function, dividing it into a training set (80%) and test data set (20%). As we can see the dimension of the training dataset after the split is (1072, 9) and for the test dataset (266, 9).

2. K-Fold Cross Validation

```
> # create a training control object for k-fold cross-validation
> trainControl <- trainControl(method = "cv", number = 10)
>
```

In k-fold cross-validation, the training dataset is divided into k equally sized folds. The models are trained k times, with each fold used as a validation set exactly once,

and the remaining k-1 folds used as training data. The performance of the model is evaluated on each validation set, and the performance metrics are typically averaged across the k folds to obtain an overall estimate of the model's performance. Here we have divided the dataset into 10 folds.

3. Use Multiple Linear Regression to show which are the variables have high statically relationships with the charges, and which are not have significant impact on charges.

```
> lmFit <- train(charges ~.,
+                data = train,
+                method = "lm",
+                trControl = trainControl)
```

```
> # print the model summary
> summary(lmFit)

Call:
lm(formula = .outcome ~ ., data = dat)

Residuals:
     Min       1Q   Median       3Q      Max
-11749.9  -2741.8   -918.2   1454.3  29607.7

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)     -11225.41    1095.37 -10.248  < 2e-16 ***
age                247.10      13.24  18.667  < 2e-16 ***
sexmale           -116.86     368.84  -0.317  0.75143
bmi                327.59      31.72  10.326  < 2e-16 ***
children           437.55     154.27   2.836  0.00465 **
smokeryes        24243.48     455.32  53.245  < 2e-16 ***
regionnorthwest   -514.25     532.52  -0.966  0.33442
regionsoutheast   -981.16     523.90  -1.873  0.06137 .
regionsouthwest  -1042.92     536.25  -1.945  0.05206 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 6010 on 1063 degrees of freedom
Multiple R-squared:  0.7594,    Adjusted R-squared:  0.7576
F-statistic: 419.4 on 8 and 1063 DF,  p-value: < 2.2e-16

>
```

Project Question 1:

What is the medical cost of the person based on health and demographic features?

From the multiple linear regression model, the variables age, bmi and smokeryes, children have high statistically relationship with the charges whereas the sexmale, regionorthwest, regionsoutheast, regionsouthwest their relationship with charges is not statistically significant at the 5% level as they have larger p-values (> 0.05). The r2 value for the multiple regression model is 76%. An R-squared (R2) value of 76%

in the context of a multiple regression model indicates that the model explains approximately 76% of the variance in the dependent variable.

```
> # make predictions
> predictions <- predict(lmFit, newdata = test)
> # calculate the mean absolute error on the test set
> MAE <- mean(abs(predictions - test$charges))
> print(paste("Mean absolute error on test set:", MAE))
[1] "Mean absolute error on test set: 4184.85679997598"
>
```

Mean absolute error after predicting the test dataset is 4184, which is good since the median value of the charges were 9382.

Now, after the model is built, we can predict the medical cost of the person based on his/her health or demographic features.

Step8: Classification

Definition of classification as a data mining techniques: Classification in data mining involves grouping or categorizing data based on patterns and relationships found in historical data. It analyzes a dataset to identify characteristics that can be used to classify new data into predefined categories. The goal is to build a model that can automatically classify data based on its attributes. In classification, historical data is used to train the model, which learns the patterns and relationships in the data. The model can then be used to classify new data based on the learned patterns. The dependent variable in classification is typically a categorical variable, such as spam/ham email, customer churn, disease diagnosis, or sentiment analysis.

Few examples of algorithms used in classification, including decision trees, logistic regression, support vector machines, k-nearest neighbors, and random forests. These algorithms analyze the data and create a model that can be used for classification tasks.

By using Classification technique or algorithms, you will be able to answer the second question.

➢ Use Decision Tree - Cross Validation Method.

First, I will take one hot encoded insurance dataset.

```
> # 8.1  Use Decision Tree - Cross Validation Method.
> head(insurance_encoded)
  age sexmale    bmi children smokeryes regionnorthwest regionsoutheast regionsouthwest   charges
1  19       0 27.900        0         1               0               0               1 16884.924
2  18       1 33.770        1         0               0               1               0  1725.552
3  28       1 33.000        3         0               0               1               0  4449.462
4  33       1 22.705        0         0               1               0               0 21984.471
5  32       1 28.880        0         0               1               0               0  3866.855
6  31       0 25.740        0         0               0               1               0  3756.622
>
```

Now, creating one new column in the dataset named charges_category which consists of categories- “Low”, “Medium”, “High”. If the charges are less than 10000, than low, if between 10000 – 20000$ than Medium, else High. I have used sapply() function to perform this operation.

```
> # Creating the categorize_charge variable  from the variable charges
> categorize_charges <- function(charges) {
+   if (charges < 10000) {
+     return("Low")
+   } else if (charges >= 10000 & charges < 20000) {
+     return("Medium")
+   } else {
+     return("High")
+   }
+ }
> insurance_encoded$charges_category <- sapply(insurance_encoded$charges, categorize_charges)
> head(insurance_encoded)
  age sexmale    bmi children smokeryes regionnorthwest regionsoutheast regionsouthwest   charges charges_category
1  19       0 27.900        0         1               0               0               1 16884.924           Medium
2  18       1 33.770        1         0               0               1               0  1725.552              Low
3  28       1 33.000        3         0               0               1               0  4449.462              Low
4  33       1 22.705        0         0               1               0               0 21984.471             High
5  32       1 28.880        0         0               1               0               0  3866.855              Low
6  31       0 25.740        0         0               0               1               0  3756.622              Low
>
```

Now, removing the original charges column and changing new column charges_category as factor.

I will use this new column as my target column for the classification problem.

```
> # Removing original charges column from dataset
> insurance <-  subset(insurance_encoded, select = -c(charges))
> # changing charges_category column as factor
> insurance$charges_category <- as.factor(insurance$charges_category)
```

Split the dataset.

```
> #Split data
> library(caTools)
> sample_data = sample.split(insurance, SplitRatio = 0.8)
> train <- subset(insurance, sample_data == TRUE)
> test <- subset(insurance, sample_data == FALSE)
> dim(train)
[1] 1041    9
> dim(test)
[1] 297   9
> library(party)
> set.seed(123) # for reproducibility
```

Building the decision tree classification model with cross validation method. I have done 10 folds cross-validation. The target variable is charges_category. The train data is used to train the decision tree model for classification purposes. From the model, we will classify whether the medical charges will be low, medium, or high, ,tuneLength = 10: This specifies the number of tuning parameter values to be evaluated during model training, which is set to 10 in this case. Tuning parameters are used to optimize the model performance. The purpose this tuning hyperparameters is to find the optimal combination of hyperparameter values that result in the best performance of the model.

```
> # for cross validation
> ctrl <- trainControl(method = "cv", number = 10)
> model <- train(charges_category ~ ., data = train, method = 'rpart', trControl = ctrl, tuneLength = 10)
> print(model)
CART

1041 samples
   8 predictor
   3 classes: 'High', 'Low', 'Medium'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 936, 936, 937, 937, 937, 937, ...
Resampling results across tuning parameters:

  cp          Accuracy   Kappa
  0.00000000  0.8905554  0.8125414
  0.03668981  0.8481998  0.7377192
  0.07337963  0.8395458  0.7237046
  0.11006944  0.8395458  0.7237046
  0.14675926  0.8395458  0.7237046
  0.18344907  0.8395458  0.7237046
  0.22013889  0.8395458  0.7237046
  0.25682870  0.8395458  0.7237046
  0.29351852  0.8395458  0.7237046
  0.33020833  0.7100804  0.4193150

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was cp = 0.
>
```

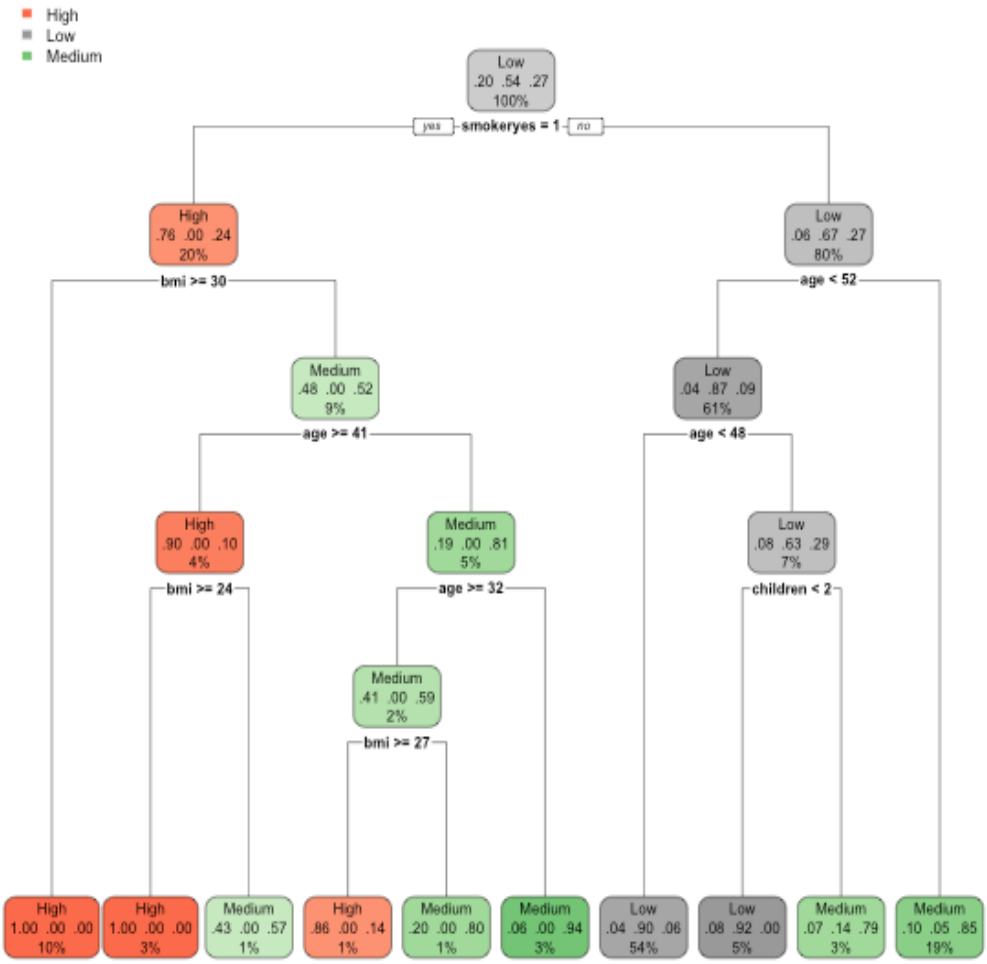
CART (Classification and Regression Trees) model that was trained and evaluated using 1041 samples, 8 predictors, and 3 classes ('High', 'Low', 'Medium'). The

decision tree model with a cp (Complexity Parameter) value of 0 was selected as the final model based on the largest accuracy value.

Plotting decision tree using rpart.plot packages.

```
> # Plot the decision tree
> install.packages("rpart.plot")
trying URL 'https://cran.rstudio.com/bin/macosx/contrib/4.2/rpart.plot_3.1.1.tgz'
Content type 'application/x-gzip' length 1013546 bytes (989 KB)
==================================================
downloaded 989 KB


The downloaded binary packages are in
        /var/folders/r7/32bm9v6164j0k8z2qvjcfrq40000gn/T//Rtmpo3IU6X/downloaded_packages
> library(rpart.plot)
Loading required package: rpart
> rpart.plot(model$finalModel)
>
```



Predicting test data and creating confusion matrix using confusionMatrix() function.

```
> predict <- predict(model, newdata = test)
> confusionMatrix <- confusionMatrix(predict, test$charges_category)
> confusionMatrix
Confusion Matrix and Statistics

          Reference
Prediction High Low Medium
    High     48   0      2
    Low       5 146      7
    Medium   16   5     68

Overall Statistics

               Accuracy : 0.8822
                 95% CI : (0.8399, 0.9165)
    No Information Rate : 0.5084
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.8077

 Mcnemar's Test P-Value : 0.001021

Statistics by Class:

                     Class: High Class: Low Class: Medium
Sensitivity               0.6957     0.9669        0.8831
Specificity               0.9912     0.9178        0.9045
Pos Pred Value            0.9600     0.9241        0.7640
Neg Pred Value            0.9150     0.9640        0.9567
Prevalence                0.2323     0.5084        0.2593
Detection Rate            0.1616     0.4916        0.2290
Detection Prevalence      0.1684     0.5320        0.2997
Balanced Accuracy         0.8434     0.9423        0.8938
>
```

As seen above, the model has an overall accuracy of 88.22% for test data.

➢ Use k-Nearest Neighbors - Split Train-Test method.

First, splitting the dataset that we used for the decision tree. I have taken the same dataset after adding the column charges_category, the dataset contains charges_category and will be used as the target variable for the KNN.

```
> # Split the data into training and testing sets
> head(insurance)
  age sexmale    bmi children smokeryes regionnorthwest regionsoutheast regionsouthwest charges_category
1  19       0 27.900        0         1               0               0               1           Medium
2  18       1 33.770        1         0               0               1               0              Low
3  28       1 33.000        3         0               0               1               0              Low
4  33       1 22.705        0         0               1               0               0             High
5  32       1 28.880        0         0               1               0               0              Low
6  31       0 25.740        0         0               0               1               0              Low
> set.seed(123)
> sample_data <- sample(nrow(insurance), 0.8 * nrow(insurance))
> train <- insurance[sample_data, ]
> test <- insurance[-sample_data, ]
> dim(train)
[1] 1070    9
> dim(test)
[1] 268   9
>
```

Train KNN model:

For knn model, first, we need to install the class library, k = 35 set the numbers of neighbors.

```
> install.packages("class")
trying URL 'https://cran.rstudio.com/bin/macosx/contrib/4.2/class_7.3-21.tgz'
Content type 'application/x-gzip' length 96047 bytes (93 KB)
==================================================
downloaded 93 KB


The downloaded binary packages are in
        /var/folders/r7/32bm9v6164j0k8z2qvjcfrq40000gn/T//Rtmpwoc50u/downloaded_packages
> library(class)
> k <- 35  # set the number of neighbors
> knn_model <- knn(train = train[, -9], test = test[, -9], cl = train[, 9], k = k)
>
```

Accuracy of the KNN model:

```
> # Evaluate the accuracy of the model
> accuracy <- sum(knn_model == test[,9]) / length(knn_model)
> cat("Accuracy of KNN:", accuracy)
Accuracy of KNN: 0.7201493
>
```

Project Question 2:

What type of insurance package that can offer to a certain person?

After creating the classification decision tree and KNN model, now we can predict the insurance package that can be offered to a certain person based on his/her health and demographic features as we did for the test data set.

➢Make a comparison between both, based on the accuracy, which algorithm best in classifying the customer.

The accuracy of the decision tree model is 88.22% whereas the accuracy of the KNN model is 72% on classifying the test dataset. Seeing the accuracy result, we can say that the decision tree is the best classification model for the insurance dataset.