# Topic: Naive Bayes

1.) Prepare a classification model using Naive Bayes for Salary data

**Ans:**

**Analyzing the input and output variables:**

* Output Variable(y) - Salary
* Input Variable (x) - Other Factors

**Importing data and Data Preprocessing:**

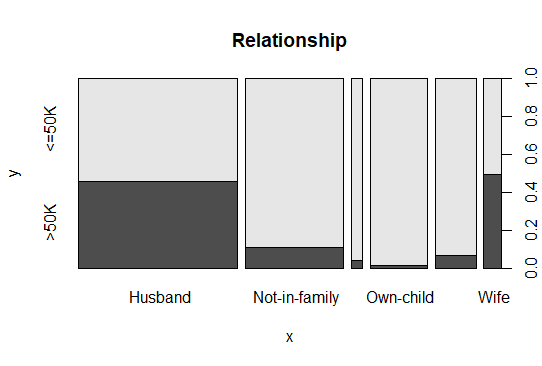
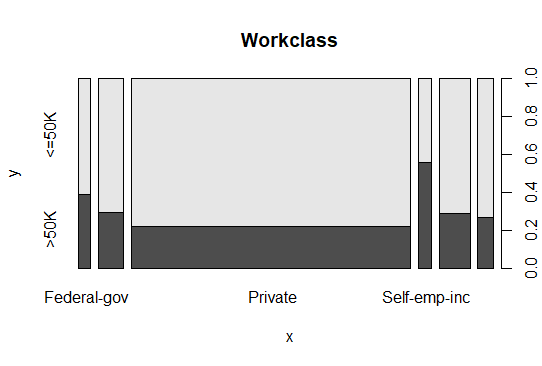
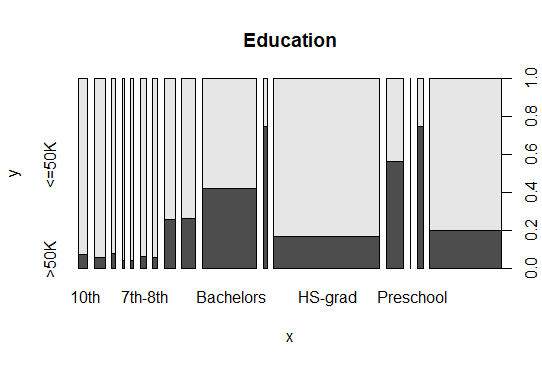
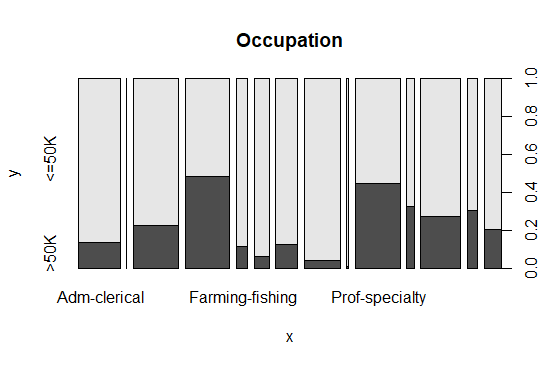
* Data file is imported by Text(base) to convert strings into factors.
* As there are no missing values and no imputation of data required.

**Normalizing the data:**

* Using the normalization function, the data is normalized and the data will come under same level.

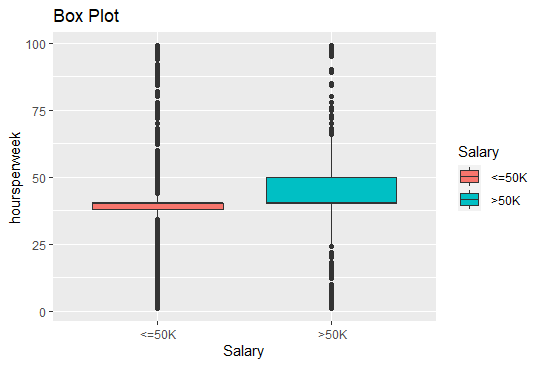
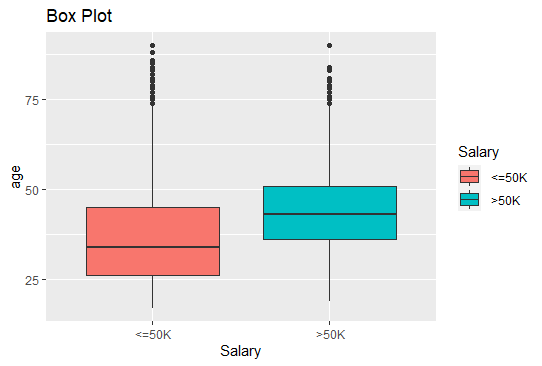
**Exploratory Data Analysis:**

**Plot Representation:**

****

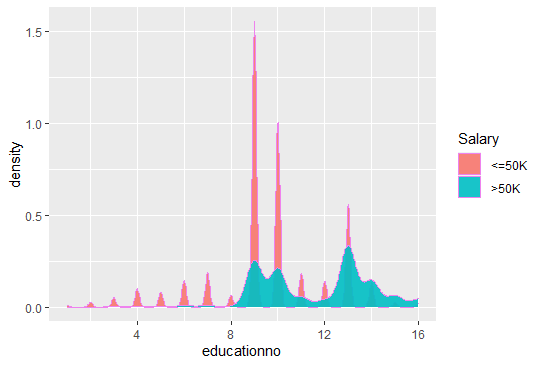
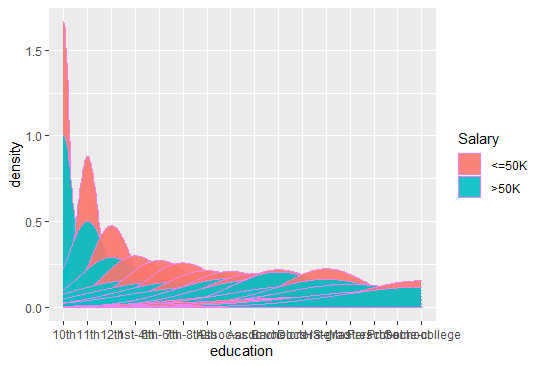
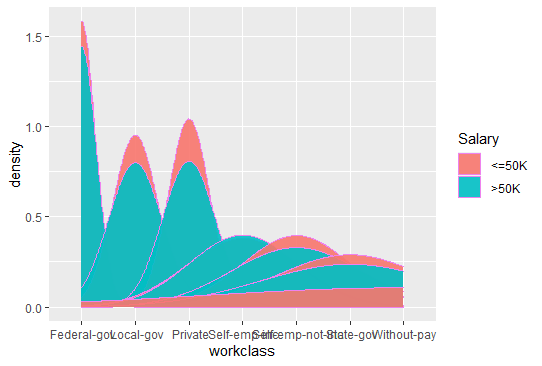
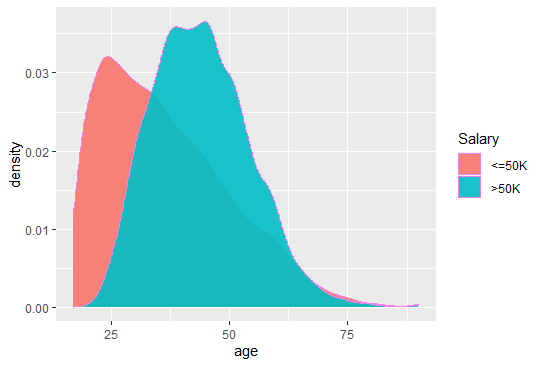
* From the above graphical representation, we can clearly see how the data is spread in each individual variable.

**Box Plot Representation:**

****

* The box plots represent that there exist outliers in the input and output variables.

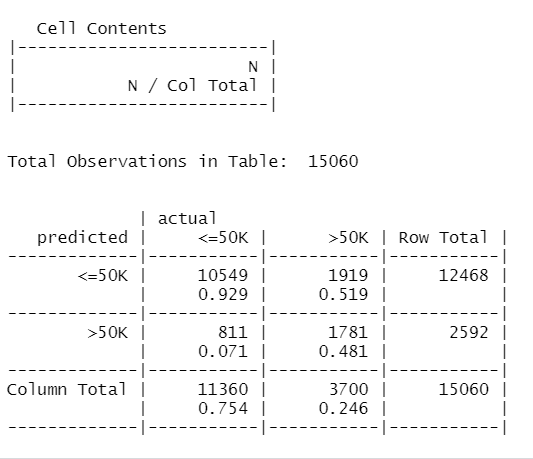
**Density Plots:**

****

* The density plot observes the distribution of data of a variable.

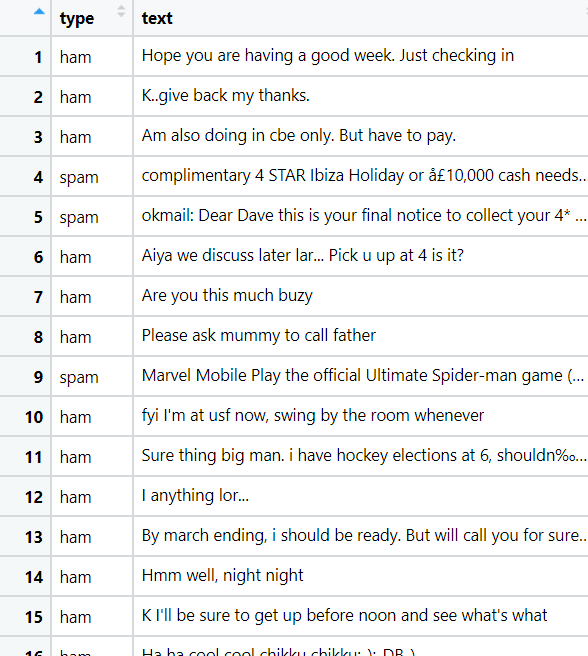
**Building a Model using Naïve Bayes Analysis:**

* Now the e1071 package is loaded that holds the Naive Bayes function. This is an in-built function provided by R.
* With the help of library(gmodels) a cross table built for the test data.



* After loading the package, the below code snippet will create Naive Bayes model by using the training data set:
* To check the efficiency of the model, we are now going to run the testing data set on the model, after which we will evaluate the accuracy of the model by using a Confusion matrix.
* The final output shows that we built a Naive Bayes classifier that can predict salary, with an accuracy of approximately 82%.

2.) Build a Naive Bayes model on the data set for classifying the ham and spam



**Ans:**

**Business Problem:**

**Objective**: To identify the spam text

**Constraints**: To minimize the spam text

**Data Preprocessing:**

* Loading the raw data set of SMS
* The type of SMS (ham/spam) is factorized to 2 level
* We need examine the “type” variable
* The proportion of ham and spam are 4812 and 747.

**Building a corpus using text mining package:**

* Creating a corpus vector source for the text
* Now the corpus is cleaned by using text mining function called tm\_map ()
* Converting all the upper-case letter to lower case letters in the corpus
* Removing the punctuation marks from the text data
* Removing the numeric values from the data
* Removing the stop words and the white spaces generated after removing stop words from corpus
* Once the corpus is cleaned, document term matrix for cleaned corpus

**Creating the Train and Test Data:**

* Creating the train and test data for raw data, cleaned corpus data and dtm data.
* Checking the proportion of spam in raw data, train and test data is similar or not
* prop.table(table(sms\_raw$type))

ham spam

0.8656233 0.1343767

* prop.table(table(sms\_raw\_train$type))

ham spam

0.8647158 0.1352842

* prop.table(table(sms\_raw\_test$type))

ham spam

0.8683453 0.1316547

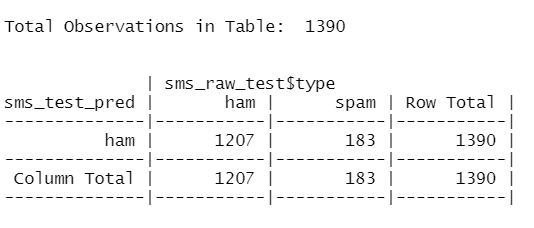
* From the above proportion table, the spam data is divided in the similar proportion

**Indicator features for frequent words:**

* Creating the matrix for which the words are repeated more than 5 times for both train and test data using the function list(dictionary = sms\_dict)
* There are nearly 1230 words are repeated more that 5 times
* Using custom function converting counts to factors
* Here if the word is repeated more than “0 “times then it will be written as “1” else will be mentioned as “0”
* Now the converted counts are applied to train and test data as Margin = 2 for columns and margin = 1 for rows

**Training the model on the data:**

* Using the library(e1071) the Naive Bayes model is built on the train data.
* Now evaluating the model performance in the test data
* With the help of library(gmodels) a cross table built for the test data.



* The test accuracy is 0.87 and training accuracy is 0.86, which shows the model is the best fit.

**Hints:**

1. Business Problem
   1. Objective
   2. Constraints (if any)
2. Data Pre-processing

2.1 Data cleaning, Feature Engineering etc.

1. Model Building
   1. Partition the dataset
   2. Model(s) - Reasons to choose any algorithm
   3. Model(s) Improvement steps
   4. Model Evaluation
   5. Python and R codes
2. Deployment

4.1 Deploy solutions using R shiny and Python Flask.

1. Result Share the benefits/impact of the solution - how or in what way the business (client) gets benefit from the solution provided.

**Note:**

1. For each assignment the solution should be submitted in the format
2. Research and Perform all possible steps for improving the model(s) accuracy Ex: Feature Engineering, Hyper Parameter tuning etc.
3. All the codes (executable programs) are running without errors
4. Documentation of the module should be submitted along with R & Python codes, elaborating on every step mentioned here