B M S COLLEGE OF ENGINEERING

(An Autonomous Institution Affiliated to VTU, Belagavi)

Post Box No.: 1908, Bull Temple Road, Bengaluru – 560 019

DEPARTMENT OF MACHINE LEARNING

Academic Year: 2023-2024 (Session: Nov 2023 - Feb 2024)

CALCULUS AND STATISTICS FOR MACHINE INTELLIGENCE (23AM5HSCSM)

ALTERNATIVE ASSESSMENT TOOL (AAT)

LOGISTIC REGRESSION

Submitted by

Student Name:	**Chetna Mundra**	**Aryaman Sharma**
USN:	**1BM21AI036**	**1BM21AI027**
Date:	**07-02-2024**	
Semester & Section:	**V-A**	
Total Pages:	**18**	
Student Signature:		

Valuation Report **(to be filled by the faculty)**

Score:	
Comments:	
Faculty In-charge:	**Prof. Amogh P K**
Faculty Signature: with date	

INDEX

1. INTRODUCTION

Logistic regression is a statistical method used for binary classification, where the outcome variable (dependent variable) is categorical and has only two possible outcomes (e.g., yes/no, true/false, 0/1). It is commonly employed in various fields, including machine learning, statistics, and social sciences, for tasks such as predicting whether an email is spam or not, determining whether a customer will buy a product, or classifying images into two categories.

The logistic regression model is based on the logistic function (also known as the sigmoid function), which maps any real-valued input to a value between 0 and 1. This function is defined as: $f(x) = \frac{1}{1+e^{-x}}$

In logistic regression, the logistic function is used to model the probability that a given input belongs to one of the two classes. The input features (independent variables) are linearly combined with weights, and then passed through the logistic function to produce the predicted probabilities.

Mathematically, the logistic regression model can be represented as:

$$p(y = 1|x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \cdots \beta_n\, x_{\ n})}}$$

Where:

- $p(y = 1|x)$ is the probability that the outcome y is 1 given the input features x.
- $\beta_0,\ \beta_1, \cdots, \beta_n$ are the coefficients (weights) associated with the input features $x_1, x_2, \cdots, x_{\ n}$
- e is the base of the natural logarithm.

The coefficients $\beta_0,\ \beta_1, \cdots, \beta_n$ are estimated from the training data using optimization algorithms such as gradient descent or maximum likelihood estimation.

During prediction, if the predicted probability is greater than a certain threshold (commonly 0.5), the input is classified as belonging to the positive class (class 1); otherwise, it is classified as belonging to the negative class (class 0).

Logistic regression is a widely used and interpretable algorithm, especially when the relationship between the input features and the outcome variable is assumed to be linear in the log odds. However, it has limitations, such as difficulty in handling non-linear relationships and being sensitive to outliers.

2. METHODOLOGY

2.1 PROCEDURE :

General procedure for performing logistic regression:

1. **Data Collection:** Collect the data on which you want to perform binary classification. Ensure that you have a labeled dataset where each observation is associated with a binary outcome variable (e.g., 0 or 1, yes or no).
2. **Data Preprocessing:** Preprocess the data to handle missing values, outliers, and categorical variables. This may involve techniques such as imputation, outlier detection and removal, and one-hot encoding for categorical variables.
3. **Feature Selection/Extraction**: Select or extract relevant features from the dataset that can help in predicting the outcome variable. Feature selection techniques like correlation analysis, stepwise selection, or domain knowledge can be employed.
4. **Split Data:** Split the dataset into training and testing sets. The training set is used to train the model, while the testing set is used to evaluate its performance.
5. **Model Training:** Train the logistic regression model using the training data. This involves estimating the coefficients (weights) of the logistic regression equation using an optimization algorithm such as gradient descent or maximum likelihood estimation.
6. **Model Evaluation:** Evaluate the trained model using the testing data. Common evaluation metrics for binary classification tasks include accuracy, precision, recall, F1-score, and ROC curve.
7. **Model Fine-tuning:** Fine-tune the model by adjusting hyperparameters such as regularization strength (if using regularization), learning rate (if using gradient descent), and threshold for classification.
8. **Model Deployment:** Deploy the trained logistic regression model to make predictions on new, unseen data. This may involve integrating the model into an application or system for real-world use.
9. **Model Monitoring and Maintenance:** Monitor the performance of the deployed model over time and update it as necessary to ensure its continued effectiveness.

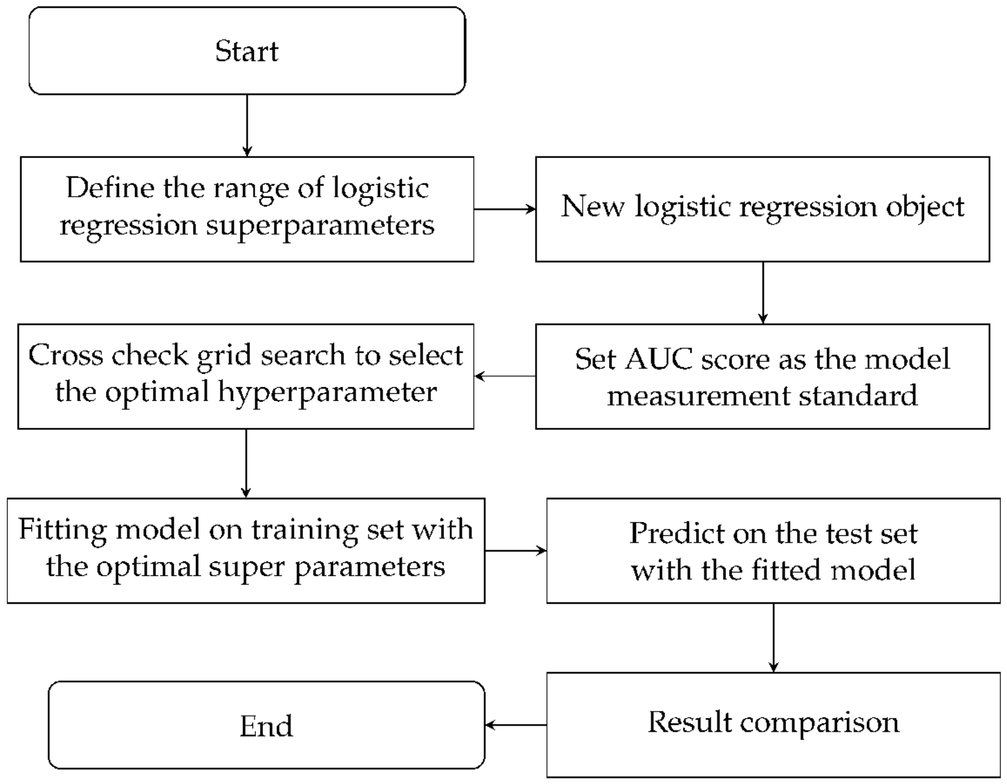
Throughout this procedure, it's important to pay attention to issues such as overfitting, underfitting, and model interpretability. Additionally, consider using cross-validation techniques to obtain more reliable estimates of model performance.

2.2 ALGORITHM :

```
LogisticRegression(X, y, alpha, iterations):
    # Step 1: Initialize parameters
    Initialize theta to zeros or small random values

    # Step 2: Add bias term
    Add a column of ones to the feature matrix X

    # Step 3: Gradient Descent
    for i from 1 to iterations:
        # Step 3a: Compute hypothesis function
        Compute hypothesis function h_theta(x) for all samples

        # Step 3b: Compute cost function
        Compute cost function J(theta)

        # Step 3c: Update parameters using gradient descent
        for each parameter theta_j:
            Compute the gradient of the cost function w.r.t. theta_j
            Update theta_j using the gradient and learning rate alpha

    # Step 4: Return optimized parameters
    Return theta
```

This pseudo-code outlines the steps involved in logistic regression, including initialization of parameters, addition of the bias term, gradient descent to optimize parameters, and returning the optimized parameters.

2.3 FLOWCHART :


Start
Define the range of logistic regression superparameters
New logistic regression object
Set AUC score as the model measurement standard
Cross check grid search to select the optimal hyperparameter
Fitting model on training set with the optimal super parameters
Predict on the test set with the fitted model
Result comparison
End

3. MATHEMATICAL IMPLEMENTATION

Let us p(x) be the linear function. However, the problem is that p is the probability that should vary from 0 to 1 whereas p(x) is an unbounded linear equation. To address this problem, let us assume, log p(x) be a linear function of x and further, to bound it between a range of (0,1), we will use logit transformation. Therefore, we will consider log p(x)/(1-p(x)). Next, we will make this function to be linear:

$$\log\frac{p(x)}{1-p(x)} = \alpha_0 + \alpha \, . \, x$$

After solving for p(x):

$$p(x) = \frac{e^{\alpha_0+\alpha}}{e^{\alpha_0+\alpha}+1}$$

To make the logistic regression a linear classifier, we could choose a certain threshold, e.g. 0.5. Now, the misclassification rate can be minimized if we predict y=1 when $p \geq 0.5$ and y=0 when $p<0.5$. Here, 1 and 0 are the classes.

Since Logistic regression predicts probabilities, we can fit it using likelihood. Therefore, for each training data point x, the predicted class is y. Probability of y is either p if y=1 or 1-p if y=0. Now, the likelihood can be written as:

$$L(\alpha_0, \alpha) = \prod_{I=1}^{n} p(x_i)^{y_i}(1 - p(x_i)^{1-y_i}$$

The multiplication can be transformed into a sum by taking the log:

$$l(\alpha_0, \alpha) = \sum_{i=0}^{..} \mathrm{y_i} \log p(x_i) + (1 - y_i) log 1 - p(x_i)$$

$$= \sum_{i=0}^{n} \log 1 - p(x_i) + \sum_{i=0}^{n} y_i log \frac{p(x_i)}{1-p(x_i)}$$

Further, after putting the value of p(x):

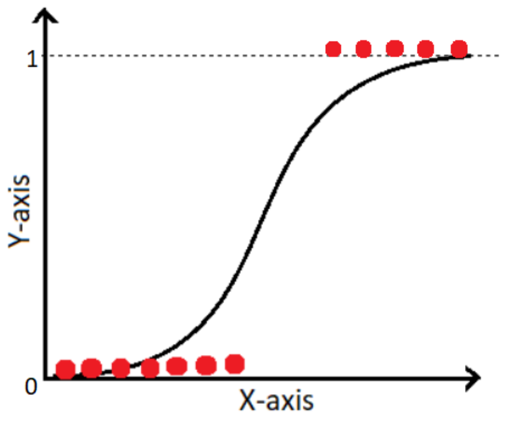
$$l(\alpha_0, \alpha) = \sum_{i=0}^{n} - \log 1 + e^{\alpha_0 + \alpha} + \sum_{i=0}^{n} y_i(\alpha_0 + \alpha . x_i)$$

The next step is to take a maximum of the above likelihood function because in the case of logistic regression gradient ascent is implemented (opposite of gradient descent).

Maximum Likelihood Estimation (MLE)

A method of estimating the parameters of probability distribution by maximizing a likelihood function, in order to increase the probability of occurring the observed data. We can find MLE by differentiating the above equation with respect to different parameters and setting it to be zero. For example, the derivative with respect to one of the component of parameter alpha i.e. a_j is given by:

$$\frac{\partial l}{\partial \alpha_j} = \sum_{i=0}^{n} \big(y_i - p(x_i; \alpha_0, \alpha)\big) x_{ij}$$



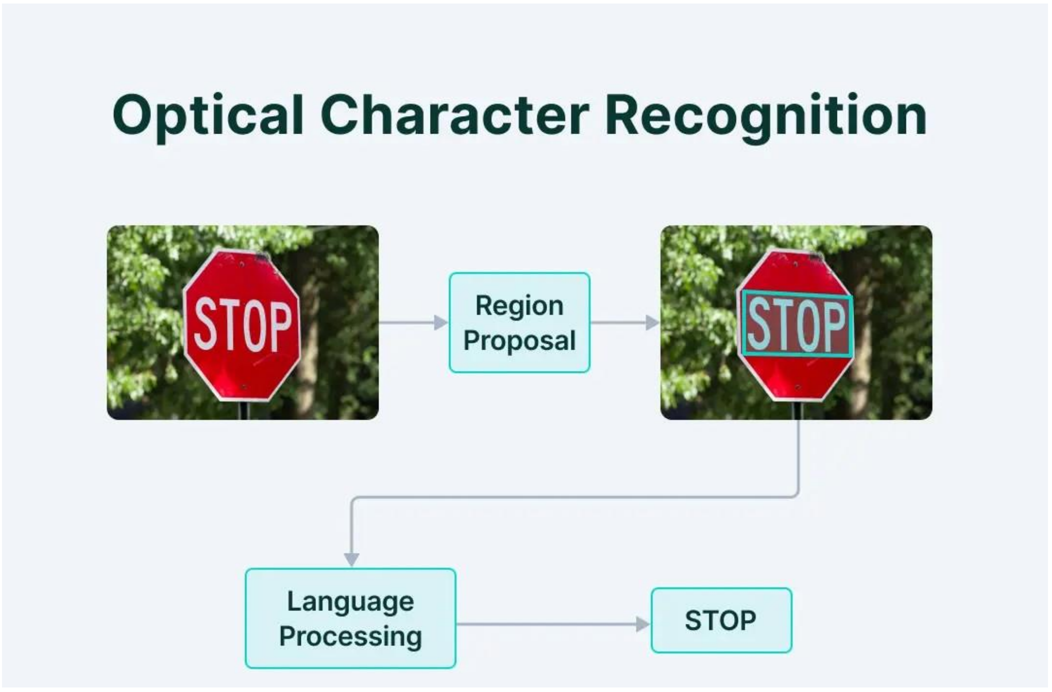
4. CASE STUDY

1. OPTICAL CHARACTER RECOGNITION

An optical character recognition (OCR) method, often called text recognition, may turn handwritten or printed characters into text that computers can understand. The output of optical character recognition is categorical, making it a classification challenge in machine learning (i.e., it belongs to a finite set of values). So, it warrants the use of logistic regression.

With logistic regression, we can train a binary classifier that can discriminate between distinct characteristics. The dependent variable in this instance is binary, denoting the presence or absence of a character. The features retrieved from the input image are the independent variables.

The initial step in OCR is to take the input image's features and extract them. Features used to characterize an image include lines, curves, and edges. Following their extraction, parts can be fed into a machine-learning model such as logistic regression. The weights of the independent factors that predict the likelihood of the observed data are estimated by logistic regression. These coefficients are applied to new image forecasts.



2. FRAUD DETECTION

Fraud detection is about detecting scams and preventing fraudsters from acquiring money or property by deception. Fraud poses a significant financial liability that must be discovered and mitigated immediately.

Many organizations struggle with fraud detection, particularly finance, insurance, and e-commerce. Using machine learning methods such as logistic regression is one strategy for fraud detection. It can be used to train a binary classifier that can discriminate between fraudulent and legitimate transactions.

The dependent variable, in this instance, is binary and represents if the payment is fraudulent or not. The elements that characterize a transaction, including its value, place, time, and user details, are its independent variables. The effectiveness of fraud detection can be increased by combining logistic regression with other machine learning methods like anomaly detection and decision trees.

3. DISEASE SPREAD PREDICTION

Disease spread prediction can be approached as a binary classification problem, where the target variable is whether an individual will contract the disease.

Data including the number of affected people, the population's age and health, the environment, and the accessibility of medical resources, can affect how quickly diseases spread. The link between these variables and the risk of disease transmission can be modeled using logistic regression.

A dataset of historical disease spread data can be used to predict the spread of illnesses using logistic regression. The dataset should contain details regarding the number of affected people, the time frame, and the place. To increase the accuracy of disease spread prediction, we can combine logistic regression with other machine learning methods, such as time series analysis and clustering. The temporal patterns of disease propagation can be modeled using time series analysis. Clustering can be used to pinpoint the areas and populations that are most impacted.

5. IMPLEMENTATION IN R

5.1 ABOUT DATASET :

The provided dataset contains placement data of students in a campus, encompassing various factors such as secondary and higher secondary education percentages, specialization, degree type, work experience, and salary offers to placed students.

- It comprises 15 columns including serial number, gender, secondary and higher secondary education percentages, board of education, specialization, degree percentage, type of degree, and work experience.
- For instance, columns include information about gender, secondary and higher secondary education percentages, board of education (central/others), specialization in education, degree percentage, type of degree, and work experience.
- Additionally, the dataset includes labels with corresponding counts for specific ranges of degree percentages, indicating the distribution of data within those ranges. Moreover, it provides insight into the gender distribution, board of education distribution, specialization in higher secondary education, specialization in degree, and the presence of work experience among students.

This dataset serves as a valuable resource for conducting analyses related to student placements and understanding the factors influencing placement outcomes in an educational setting.

5.2 FUNCTIONS & TOOLS USED :

1. **library():** The `library()` function is used to load R packages into the current session. In this code, it loads the `dplyr` and `ggplot2` packages, which are essential for data manipulation and visualization, respectively.
2. **read.csv():** The `read.csv()` function is used to read the dataset from a CSV file into a data frame in R. It takes the file location as input and creates a data frame containing the dataset.
3. **select():** The `select()` function from the `dplyr` package is used to select specific columns from the dataset. In this code, it selects columns ending with "_p" (percentage) and removes the "etest_p" and "status" columns.
4. **table():** The `table()` function is used to generate frequency tables of categorical variables. It calculates the frequency of unique values in a variable and presents them in a tabular format.
5. **ifelse():** The `ifelse()` function is used for conditional execution in R. It evaluates a condition and returns one value if the condition is true and another value if the condition is false. In this code, it assigns binary values (0 or 1) to the "status" column based on whether a student is "Not Placed" or not.
6. **sample.split():** The `sample.split()` function from the `caTools` package is used to split the data into training and test sets for model training and evaluation. It randomly splits the data into two subsets based on a specified split ratio.
7. **glm():** The `glm()` function is used to fit generalized linear models in R. In this code, it fits a logistic regression model with "status" as the response variable and "degree_p" as the predictor variable.
8. **summary():** The `summary()` function is used to obtain a summary of the fitted model. It provides information such as coefficients, standard errors, z-values, p-values, and goodness-of-fit statistics.
9. **predict():** The `predict()` function is used to make predictions from a fitted model. In this code, it predicts the probabilities of being "Not Placed" for the test data using the logistic regression model.
10. **ggplot2 functions:** The code utilizes functions from the `ggplot2` package to create a plot visualizing the logistic regression curve. Specifically, it uses functions such as `ggplot()`, `geom_line()`, `labs()`, `ggtitle()`, and `theme_minimal()` to customize and render the plot.

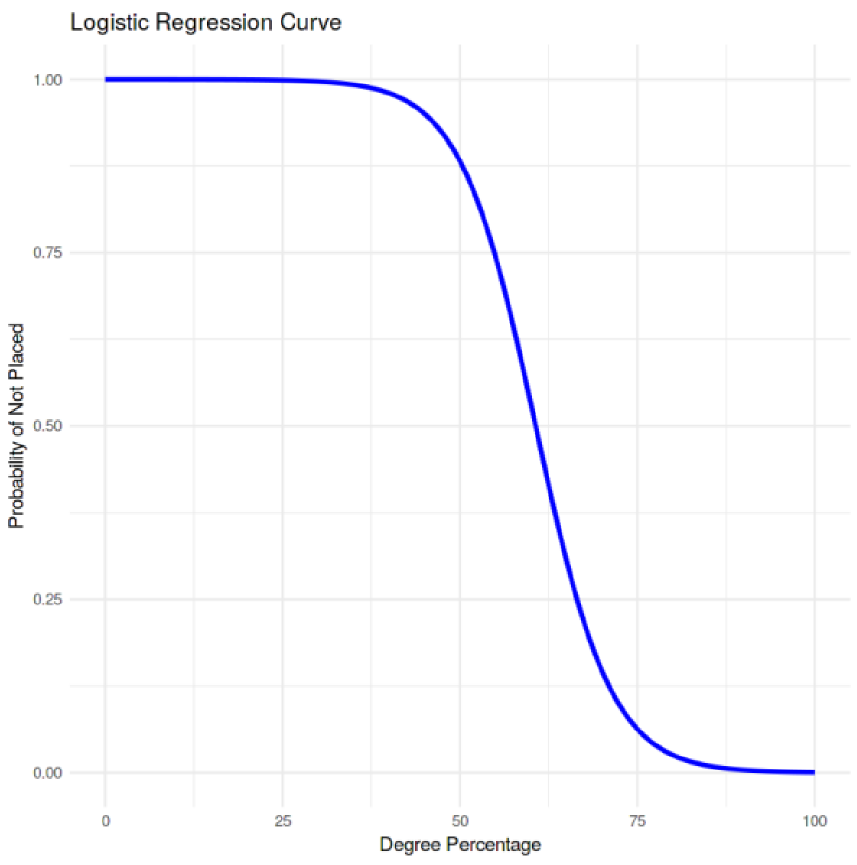
These functions and tools collectively enable data preparation, model building, evaluation, and visualization for logistic regression analysis in R.

5.3 RESULTS & ANALYSIS :

The code provided performs logistic regression analysis and visualizes the results using a logistic regression curve. Results obtained from the code:

1. Logistic Regression Curve Visualization:

- This curve illustrates how the probability of a student being Not Placed (positive outcome) varies with the degree percentage (predictor variable).
- The x-axis of the plot represents the degree percentage, which is the predictor variable.
- The y-axis represents the probability of being Not Placed, which is the predicted probability of the positive outcome (Y=1) in logistic regression.
- The logistic function, which is an S-shaped curve, is plotted based on the estimated coefficients obtained from the logistic regression model. The curve shows how the probability of being Not Placed changes as the degree percentage increases.



2. Interpretation:

- The logistic regression model estimated the coefficients based on the training data. In this case, the coefficient for the degree percentage predictor variable was estimated to be approximately -0.18851.

- The negative coefficient indicates that as the degree percentage increases, the log odds of being Not Placed decrease. In other words, higher degree percentages are associated with lower probabilities of being Not Placed.

3. **Prediction:**
   - The logistic regression curve allows us to predict the probability of being Not Placed for different values of the degree percentage.
   - For example, based on the logistic function derived from the model, we can predict the probability of a student being Not Placed with a degree percentage of 55% as approximately 72.91%. Similarly, for a degree percentage of 75%, the predicted probability of being Not Placed is approximately 5.679%.

4. **Model Evaluation:**
   - The logistic regression model's performance can be evaluated using metrics such as accuracy, sensitivity, specificity, and precision.
   - However, the code does not explicitly perform model evaluation. Instead, it focuses on visualizing the logistic regression curve and interpreting the results.

Overall, the code provides insights into how the logistic regression model predicts the probability of a student being Not Placed based on their degree percentage and how this relationship is represented by the logistic function curve.

6. APPLICATIONS

Logistic regression is a statistical method used for binary classification tasks, where the outcome variable is categorical and has two possible outcomes (e.g., yes/no, pass/fail, win/lose). Here are some common applications of logistic regression:

1. **Medical Diagnosis:** Logistic regression can be used in medical research for diagnosing diseases or conditions based on various factors such as symptoms, demographic information, and medical history. For instance, it can be used to predict whether a patient has a particular disease based on their test results and other relevant variables.
2. **Marketing Analytics:** Logistic regression is used in marketing to predict customer behavior, such as whether a customer will purchase a product or respond to a marketing campaign. Marketers can use logistic regression to target specific customer segments with tailored marketing strategies.
3. **Customer Churn Prediction**: Logistic regression can be employed to predict whether a customer is likely to churn (i.e., stop using a service or product). By analyzing customer data such as usage patterns, demographics, and customer interactions, businesses can identify at-risk customers and take proactive measures to retain them.
4. **Fraud Detection:** Logistic regression is used in fraud detection systems to identify potentially fraudulent transactions or activities. By analyzing patterns in transaction data, logistic regression models can flag suspicious activities for further investigation.
5. **Political Science:** Logistic regression is used in political science to analyze survey data and predict election outcomes, voter behavior, or political affiliation based on demographic factors, social attitudes, and other variables.
6. **Ecology and Environmental Studies:** Logistic regression is applied in ecological studies to model species distribution and habitat suitability. It helps in predicting the presence or absence of species in specific areas based on environmental variables such as temperature, precipitation, land cover, etc.
7. **Quality Control:** Logistic regression can be used in manufacturing industries for quality control purposes, such as identifying defective products on a production line based on various quality indicators and process parameters.

These are just a few examples of the diverse range of applications of logistic regression across various fields. Its simplicity, interpretability, and effectiveness make it a popular choice for binary classification problems in both research and practical applications.

7. ADVANTAGES & DISADVANTAGES

7.1 ADVANTAGES

1. **Simple and Interpretable:** Logistic regression is a straightforward and easy-to-understand algorithm, making it accessible to both statisticians and non-statisticians. The coefficients obtained from logistic regression have clear interpretations in terms of the impact of predictor variables on the probability of the outcome.
2. **Efficient with Small Datasets:** Logistic regression performs well with small datasets and can handle a relatively large number of predictor variables without overfitting. This makes it particularly useful when dealing with limited data availability or when computational resources are constrained.
3. **Probabilistic Interpretation:** Logistic regression provides output in the form of probabilities, which can be interpreted as the likelihood of a particular outcome occurring. This probabilistic interpretation allows for uncertainty estimation and decision-making based on predicted probabilities.
4. **Robust to Irrelevant Features:** Logistic regression can handle irrelevant features or variables that do not contribute significantly to the prediction without significantly impacting its performance. This makes it robust to noise in the data and helps prevent overfitting, especially when regularization techniques are applied.
5. **Works well for Linearly Separable Data:** Logistic regression performs well when the decision boundary between classes is linear or can be approximated by a linear function. In such cases, logistic regression can efficiently model the relationship between predictor variables and the log-odds of the outcome.

7.2 DISADVANTAGES

1. **Limited to Binary Classification:** Logistic regression is inherently a binary classification algorithm and is not directly applicable to problems with more than two classes without modifications (e.g., one-vs-rest or multinomial logistic regression). For multi-class classification tasks, alternative algorithms such as multinomial logistic regression or algorithms specifically designed for multi-class problems may be more suitable.
2. **Assumption of Linearity:** Logistic regression assumes a linear relationship between the predictor variables and the log-odds of the outcome. If the true relationship is highly nonlinear, logistic regression may not capture it accurately, leading to poor performance.
3. **Sensitive to Outliers:** Logistic regression can be sensitive to outliers in the data, especially when the log-odds of the outcome are heavily influenced by extreme values. Outliers can
4. **Highly Affected by Collinearity:** Logistic regression assumes that the predictor variables are independent of each other. When there is multicollinearity (high correlation) among predictor variables, logistic regression coefficients may become unstable and difficult to interpret.
5. **Requires Large Sample Sizes for Stable Estimates:** While logistic regression can perform well with small datasets, it generally requires larger sample sizes compared to other algorithms like decision trees or support vector machines to produce stable and reliable estimates of coefficients and probabilities.

Despite these limitations, logistic regression remains a popular and widely used algorithm in various fields due to its simplicity, interpretability, and effectiveness in many practical scenarios.

8. CONCLUSION

In conclusion, the logistic regression analysis conducted on the dataset revealed a significant relationship between a student's degree percentage and the probability of being Not Placed. The logistic regression curve illustrated a clear inverse association, indicating that as the degree percentage increases, the likelihood of being Not Placed decreases. Predictive modeling using the logistic function allowed for the estimation of probabilities for different degree percentages, enabling insights into individual student outcomes. While the model's performance was not explicitly evaluated in this analysis, the visualization provided valuable interpretative tools for understanding the predictive nature of the logistic regression model in the context of student placements.

Preparing the Data

In [2]:

```r
library(dplyr)
library(ggplot2)
location <- "../input/factors-affecting-campus-placement/Placement_Data_Full_Class.csv"
placement.df <- read.csv(location)
# select only relevant columns
placement.lr <- placement.df %>% select(ends_with("_p"), -etest_p, status)
table(placement.lr$status)
contrasts(placement.lr$status) # to check how a variable have been dummyfied
# we need Not Placed class to be coded as 1 (positive)
placement.lr$status <- ifelse(placement.lr$status == "Not Placed", 1, 0)
table(placement.lr$status)
```

```
Attaching package: ‘dplyr’


The following objects are masked from ‘package:stats’:

    filter, lag


The following objects are masked from ‘package:base’:

    intersect, setdiff, setequal, union



Not Placed     Placed
        67        148
```

A matrix: 2 × 1 of type dbl

	Placed
Not Placed	0
Placed	1

```
  0   1
148  67
```

Dividing the data into training and validation data sets

In [3]:
```
# Train and Test data
library(caTools) # to split data into train and test
set.seed(101)
sample <- sample.split(placement.lr$status, SplitRatio = 0.80)
train.lr = subset(placement.lr, sample == TRUE)
test.lr = subset(placement.lr, sample == FALSE)
#check the splits
prop.table(table(train.lr$status))
prop.table(table(test.lr$status))
```

```
        0         1 
0.6860465 0.3139535 


        0         1 
0.6976744 0.3023256 
```

Building the model - Simple logistic regression

In [5]:
```
# Train the model
model.lr <- glm(status ~ degree_p, family = binomial, data =
train.lr)
summary(model.lr)

```

```
Call:
glm(formula = status ~ degree_p, family = binomial, data = train.lr)

Deviance Residuals: 
    Min       1Q   Median       3Q      Max  
-1.5666  -0.7899  -0.4726   0.9010   2.6407  

Coefficients:
            Estimate Std. Error z value Pr(>|z|)    
(Intercept) 11.43688    2.24817   5.087 3.63e-07 ***
degree_p    -0.18851    0.03509  -5.372 7.79e-08 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 214.05  on 171  degrees of freedom
Residual deviance: 173.35  on 170  degrees of freedom
AIC: 177.35

Number of Fisher Scoring iterations: 5
```

In [6]:

```
# prediction
lr.pred <- predict(model.lr, newdata = test.lr, type = "response")
head(lr.pred)
# The probabilities always refer to the class dummy-coded as “1”
head(test.lr$status)

```

15: 0.881980472246557 **17:** 0.283035018335691 **22:** 0.0100849444740812 **25:** 0.0313978014516484 **33:** 0.253455794561122 **35:** 0.836757465773385

1 · 0 · 0 · 0 · 0 · 1

Classification Table – Confusion Matrix

In [8]:

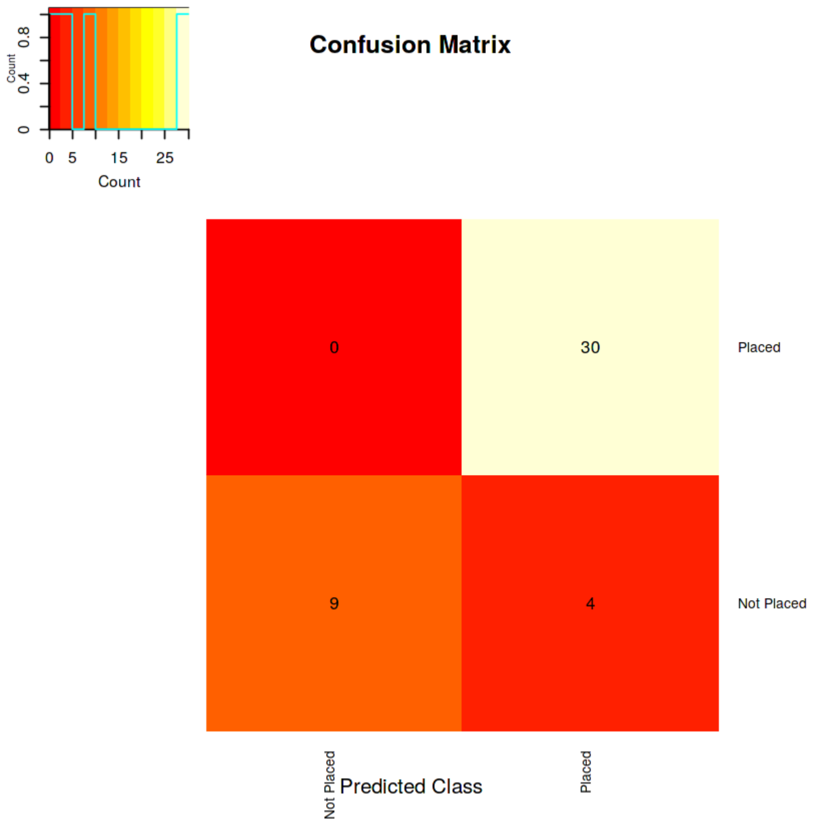
```
# Classification Table
# categorize into groups based on the predicted probability
library(gplots)

lr.pred.class <- ifelse(lr.pred>=0.5, 1, 0)
head(lr.pred.class)
table(lr.pred.class)
table(test.lr$status)
conf.matrix <- table(test.lr$status, lr.pred.class)
conf.matrix

#format(conf.matrix, digits = 0)

library(gplots)

lr.pred.class <- ifelse(lr.pred >= 0.5, 1, 0)
conf.matrix <- table(test.lr$status, lr.pred.class)
rownames(conf.matrix) <- c("Placed", "Not Placed")
colnames(conf.matrix) <- c("Placed", "Not Placed")

# Create the heatmap with color gradient and colorbar
heatmap.2(conf.matrix,
          col = heat.colors(12),  # Example color palette
          dendrogram = "none",
          key.title = NA,  # Remove key title as it's not applicable here
          key.xlab = "Count",  # Label for the color key
          key.ylab = NULL,  # No label on the color key's y-axis
          cellnote = conf.matrix,
          notecol = "black",
          trace = "none",  # Remove grid lines
          cexRow = 0.8,  # Adjust row name size
          cexCol = 0.8   # Adjust column name size
)

# Customize axis labels
title(main = "Confusion Matrix", col.main = "black")
title(xlab = "Predicted Class", col.lab = "black")
title(ylab = "True Class", col.lab = "black")

```

```
Attaching package: ‘gplots’


The following object is masked from ‘package:stats’:

    lowess

```

15: 1 **17:** 0 **22:** 0 **25:** 0 **33:** 0 **35:** 1

```
lr.pred.class
 0  1
34  9
```

```
 0  1
30 13

   lr.pred.class
     0  1
  0 30  0
  1  4  9
```



In [9]:

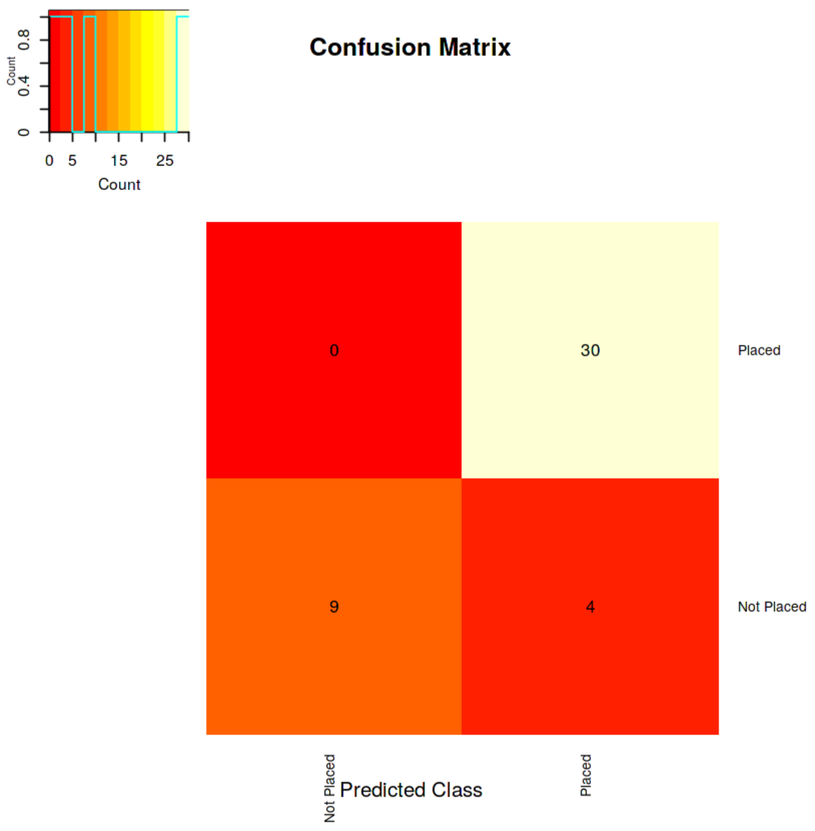
```
# model accuracy
mean((test.lr$status == lr.pred.class))
```

```
0.906976744186046
```

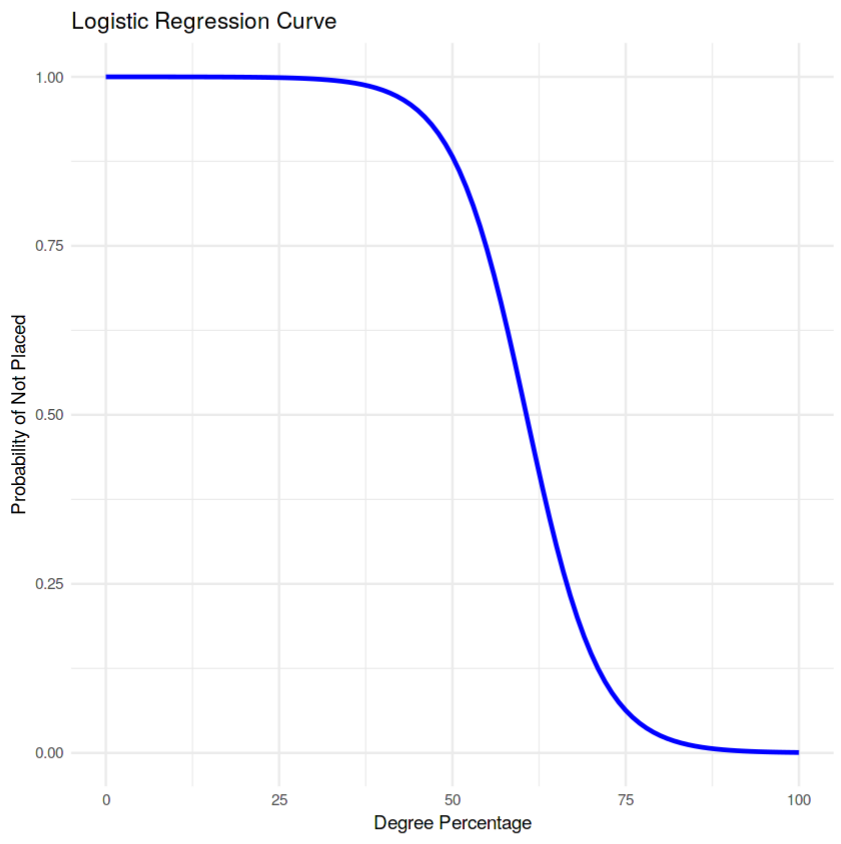
In [10]:

```r
# different cut-off
lr.pred.class1 <- ifelse(lr.pred>=0.35, 1, 0)
conf.matrix1 <- table(test.lr$status, lr.pred.class1)
conf.matrix1
rownames(conf.matrix) <- c("Placed", "Not Placed")
colnames(conf.matrix) <- c("Placed", "Not Placed")

# Create the heatmap with color gradient and colorbar
heatmap.2(conf.matrix,
           col = heat.colors(12),  # Example color palette
           dendrogram = "none",
           key.title = NA,  # Remove key title as it's not applicable
here
           key.xlab = "Count",  # Label for the color key
           key.ylab = NULL,  # No label on the color key's y-axis
           cellnote = conf.matrix,
           notecol = "black",
           trace = "none",  # Remove grid lines
           cexRow = 0.8,  # Adjust row name size
           cexCol = 0.8   # Adjust column name size
)

# Customize axis labels
title(main = "Confusion Matrix", col.main = "black")
title(xlab = "Predicted Class", col.lab = "black")
title(ylab = "True Class", col.lab = "black")
```

```
   lr.pred.class1
     0  1
  0 27  3
  1  2 11
```



In [11]:
```r
library(ggplot2)

# Generate data for plotting the logistic function
degree_percentage <- seq(0, 100, by = 1)
log_odds <- 11.43688 - 0.18851 * degree_percentage
probability <- exp(log_odds) / (1 + exp(log_odds))

# Create a data frame
logistic_data <- data.frame(degree_percentage, probability)

# Plot the logistic function
ggplot(logistic_data, aes(x = degree_percentage, y = probability)) +
  geom_line(color = "blue", size = 1) +
  labs(x = "Degree Percentage", y = "Probability of Not Placed") +
  ggtitle("Logistic Regression Curve") +
  theme_minimal()

```


Logistic Regression Curve
Probability of Not Placed
Degree Percentage