**Programming with R**

**Module 5: Regression and Classification**

**Simple Linear Regression**

Simple Linear Regression (SLR) is a statistical technique for finding the existence of an association relationship between a dependent variable (response variable or outcome variable) and an independent variable (explanatory variable or predictor variable). Regression models do not establish a causal relationship between the dependent variable (Y) and the independent variable (X). In other words, using regression we cannot say that the value of Y depends on the value of X (or a change in the value of Y is caused due to a change in the value of X). We can only establish that the change in the value of Y is associated with the change in the value of X.

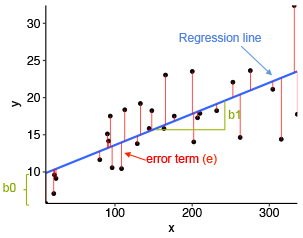
SLR implies that that there is only one independent variable in the model and the functional relationship between the dependent variable and the regression coefficient is linear. One of the functional forms of SLR is –

where:

b0and b1 are known as the regression *beta coefficients* or *parameters*:

* b0 is the *intercept* of the regression line; that is the predicted value when X = 0
* b1 is the *slope* of the regression line

e is the *error term* (also known as the *residual errors*), the part of Y that cannot be explained by the regression model



The regression parameters are estimated using the method of Ordinary Least Squares (OLS). In ordinary least squares, the objective is to find the optimal value of b0 and b1 that will minimise the Sum of Squares Error (SSE). The equation is used to predict a quantitative outcome Y on the basis of one single predictor variable X. Once, we build a statistically significant model, it is possible to use it for predicting future outcome on the basis of new X values.

library(dplyr)

library(ggplot2)

location <- "D:/Material\_CMS/R\_2020/Placement\_Data\_Full\_Class.csv"

placement.df <- read.csv(location)

# select only relevant columns

colnames(placement.df)

placement.reg <- select(placement.df, degree\_p, mba\_p)

str(placement.reg)

placement.reg %>% cor()

# Basic Visualisation

ggplot(placement.reg, aes(degree\_p, mba\_p)) +

geom\_point()

ggplot(placement.reg, aes(x = cut(degree\_p, breaks = 5), y = mba\_p)) +

geom\_boxplot()

ggplot(placement.reg, aes(degree\_p, mba\_p)) +

geom\_point() + geom\_smooth()

**SLR Model**

The function lm() can be used to determine the beta coefficients of the linear model.

# model

# mba\_p = b0 + b1\*degree\_p

model1 <- lm(mba\_p~degree\_p, data = placement.reg)

model1

From the output, the estimated values of b0 and b1 are: b0 = 41.109 and b1 = 0.319

**Regression equation:** The corresponding regression equation is given by-

mba\_p = 41.109 + 0.319\*degree\_p

**Interpretation of SLR coefficients:** The equation is interpreted as follows- For every one percentage increase in degree percent, the MBA percent will increase by 0.319% on an average.

**Plotting the regression line:**

# Regression line

ggplot(placement.reg, aes(degree\_p, mba\_p)) +

geom\_point() + geom\_smooth(method = "lm", se = FALSE)

**Validation of the SLR Model**

It is important to validate the regression model to ensure its validity and goodness of fit before it can be used for practical applications. The following measures are used to validate the SLR model:

1. Coefficient of determination (R-square)
2. Hypothesis test for the regression coefficient (b1)
3. Analysis of Variance for overall model validity (relevant for MLR)
4. Residual analysis to validate the regression model assumptions
5. Outlier analysis

We start by displaying the statistical summary of the model using the function summary().

**#** Model assessment

summary(model1)

**#** Characteristics

mba\_p.hat <- fitted.values(model1)

head(mba\_p.hat)

head(placement.reg$mba\_p)

mean(mba\_p.hat) == mean(placement.reg$mba\_p) #should be same

mod1residual <- residuals(model1)

head(mod1residual)

mean(mod1residual) #should be zero

sum(mod1residual) #should be zero

1. Coefficient of Determination (R-square or R2)

The coefficient of determination measures the percentage of variation in Y explained by the model. That is, R2 is the proportion of variation in the response variable explained by the regression model. The value of R2 lies between 0 and 1. Higher values of R2 implies better fit. There is no minimum threshold for R2, however a minimum value of R2 for a given significance value (α) can be derived using the relationship between the F-statistic and R2. Mathematically, for SLR the R-square is square of the Pearson correlation coefficient.

In our example, the R-square value is 0.1619, meaning that the model explains 16.19% of the variation in MBA percentage. The remaining variation is due to other factors that were not included in the model.

Note: A high R-square value is not necessarily a good indicator of the correctness of the model; it could be a spurious relationship. The adjusted R-squared is a statistic that modifies R2 by incorporating the sample size and number of explanatory variables.

1. Hypothesis test for regression coefficient (t-Test)

The regression coefficient b1 captures the existence of a linear relationship between the response variable and the explanatory variable. If b1 = 0, we can conclude that there is no statistically significant linear relationship between the two variables.

The null and alternative hypothesis for the SLR model can be stated as follows:

Null: There is no relationship between X and Y (b1 = 0)

Alternative: There is a relationship between X and Y (b1 ≠ 0)

In our example, since the p-value for degree\_p is less than 0.05, we reject the Null and conclude that there is significant evidence suggesting a linear relationship between degree\_p (X) and mba\_p (Y).

1. Test for Overall Model: ANOVA (F-test)

Using ANOVA, we can test whether the overall model is statistically significant. However, for a SLR, the null and alternative hypothesis in ANOVA and t-test are exactly same and thus there will be no difference in the p-value.

The Null and Alternative hypothesis for F-test are given by-

Null: There is no statistically significant relationship between Y and any of the explanatory variables (i.e all regression coefficients are zero)

Alternative: Not all regression coefficients are zero

1. Residual analysis

The difference between the observed value of the dependent variable (Y) and the predicted value (, Y-hat) is called the residual (error). Each data point has one residual. Residual analysis is important to check whether the assumptions of regression models have been satisfied. It is performed to check the following-

1. The functional form of regression is correctly specified
2. The residuals are normally distributed
3. The variance of the residuals is constant (homoscedasticity)
4. If there are any outliers

Regression diagnostic plots are used to check on the above.

**Plotting the residuals:** Each vertical red segments represents the residual error between an actual value and the corresponding predicted (i.e. fitted) value.

**#** Highlighting the Residuals

ggplot(placement.reg, aes(degree\_p, mba\_p)) +

geom\_point() + geom\_smooth(method = "lm", se = FALSE) +

geom\_segment(aes(xend = degree\_p, yend = mba\_p.hat), color = "red", size = 0.3)

# Histogram of residuals for understanding

residual.df <- as.data.frame(mod1residual)

ggplot(residual.df,aes(mod1residual)) + geom\_histogram(bins = 8, fill='blue')

**Diagnostic plots:** Regression diagnostics plots can be created using the R base function plot(). The diagnostic plots show residuals in four different ways.

**#** Diagnostic Plots

plot(model1)

Note: The four plots show the top 3 most extreme data points labelled with the row numbers of the data in the data set. You might want to take a close look at them individually to check if there is anything special for the observation. The standardized residual is the residual divided by its standard deviation.

1. **Residuals vs Fitted:** Used to check the linear relationship assumption. An approximately horizontal line (red line), without distinct patterns is an indication for a linear relationship. Any pattern in the residual plot would indicate incorrect specification of the model.

In our example, there is no pattern in the residual plot. This suggests that we can assume linear relationship between the predictor (degree\_p) and the outcome variable (mba\_p).

Note: If the residual plot indicates a non-linear relationship in the data, then a simple approach is to use non-linear transformations of the predictors, such as log(X), sqrt(X) and X2, in the regression model.

1. **Normal Q-Q:** Used to examine whether the residuals are normally distributed. It is good if residuals points follow the straight dashed line.

In our example, most of the points fall approximately along the reference line, so we can assume normality. Thus the hypothesis tests (t and F) are valid.

Note: If the residual plot does not follow approximately normal distribution it puts a doubt on the validity of the model itself since we cannot trust the outcome of the hypothesis tests (p-value).

1. **Scale-Location:** (or Spread-Location). Used to check the homogeneity of variance of the residuals (homoscedasticity). This plot shows if residuals are spread equally along the ranges of the predictors. Failure to meet this assumption will result in unreliability of the hypothesis tests. It is good if you see a horizontal line with equally spread points.  Horizontal line with equally spread points is a good indication of homoscedasticity.

Note: If there is heteroscedasticity then we can expect a funnel type shape in the residual plot. A funnel shape indicates that the variance of the residuals depends on the value of independent variable (X).

1. **Residuals vs Leverage:** Used to identify influential cases, that is, extreme values that might influence the regression results when included or excluded from the analysis.

**Outliers:** Outliers are observations whose values show a large deviation from the mean value, that is () is large. That is, an outlier is a data point whose response Y does not follow the general trend of the rest of the data. Presence of an outlier can have a significant influence on values of regression coefficients. Observations whose standardized residuals are greater than 3 in absolute value are possible outliers. In our example, there are no outliers which exceed 3 standard deviations.

**High Leverage points:** A data point has high leverage if it has extreme predictor X values. Leverage value of an observation measures the influence of that observation on the overall fit of the regression function. Leverage value of more than 3(p+1)/n is treated as highly influential. In our example, there are a few high leverage points (leverage greater than 0.028). Mostly, the observations with high leverage and high residuals can be removed from the data set.

Outliers and high leverage data points have the potential to be influential, but we generally have to investigate further to determine whether or not they are actually influential. Not all outliers (or extreme data points) are influential in linear regression analysis. A metric called *Cook’s distance* is used to determine the influence of a value. This metric defines influence as a combination of leverage and residual size.

In our example, the data don’t present any influential points. Cook’s distance lines (a red dashed line) are not shown on the Residuals vs Leverage plot because all points are well inside of the Cook’s distance lines. When data points have high Cook’s distance scores they are generally located at the upper right or lower right corner of the leverage plot. We can also see the cook’s distance plot using the below code. By default, the top 3 most extreme values are labelled on the Cook’s distance plot.

**#** Cook's distance

plot(model1, 4)

**Multiple Linear Regression**

Multiple Linear Regression (MLR) is a statistical technique for finding existence of an association relationship between a dependent variable (response variable or outcome variable) and several independent variables (explanatory variable or predictor variable). The functional form of MLR is given by-

Y = b0 + b1\*X1 + b2\*X2 + …. + bp\*Xp + e

where,

Y is the dependent variable, X1, X2 … Xp are independent variables, b0 is a constant, b1, b2 … bp are the partial regression coefficients corresponding to the explanatory variables and e is the error term (residual).

In MLR the regression coefficients are called partial regression coefficients since the relationship between an explanatory variable and the response variable is calculated after removing (partial out) the effect of all the other explanatory variables in the model.

**#** select only relevant columns

colnames(placement.df)

placement.mlr <- placement.df %>% select(ends\_with("\_p"), -etest\_p)

colnames(placement.mlr)

str(placement.mlr)

**#** Correlation among numeric columns

placement.mlr %>% cor()

# Correlation Visualisation

GGally::ggcorr(placement.mlr)

GGally::ggpairs(placement.mlr)

**Dividing the data into training and validation data sets**

The data is randomly divided into mostly two subsets- training and validation/test. The proportion of training dataset is usually between 70% to 80%. The training data is used for developing the model and the validation data is used for model validation and selection.

**#** Train and Test data

library(caTools) # to split data into train and test

set.seed(1001)

sample <- sample.split(placement.mlr$mba\_p, SplitRatio = 0.80)

train = subset(placement.mlr, sample == TRUE)

test = subset(placement.mlr, sample == FALSE)

**MLR Model**

The function lm() can be used to determine the partial regression coefficients of the linear model. The statistical summary of the model can be displayed using the function summary().

**#** MLR model

mlrmodel <- lm(mba\_p~., train)

mlrmodel

**#** Model assessment

summary(mlrmodel)

From the output, the estimated values of the parameters are

b0 = 36.016; b1 = 0.063; b2 = 0.122; and b3 = 0.209

**Regression equation:** The regression model is given by-

mba\_p = b0 + b1\*ssc\_p + b2\*hsc\_p + b3\*degree\_p

mba\_p = 36.016 + 0.063\*ssc\_p + 0.122\*hsc\_p + 0.209\*degree\_p

**Interpretation of the coefficients:** The partial regression coefficient provides the change in the response variable for a unit change in the explanatory variable, when all other explanatory variables are kept constant or controlled.

For every one percentage increase in SSC, the MBA percent will increase by 0.063% provided all other variables are held constant.

Since the unit of measurement of all the explanatory variables is same, we can say that the Degree percentage has a higher impact on the MBA percentage as compared to others.

Note: If the unit of measurement of the explanatory variables is different, we have to derive the standardised regression coefficients to compare the impact.

**Validation of the Multiple Regression Model**

1. Validation of the overall regression model: F-Test

The research question that we are answering here is, does the regression model contain at least one predictor variable useful in predicting the response variable.

Analysis of Variance (ANOVA) is used to validate the overall regression model. The null and alternative hypotheses are stated as-

Null: b0 = b1 = b2 = b3 = 0

Alternative: Not all coefficients are zero

In our example, it can be seen that p-value of the F-statistic is < 0.05, which is highly significant and hence we reject the null hypothesis. This means that, at least one of the predictor variables is significantly related to the response variable.

The statement in alternative hypothesis is that not all the beta’s are zero, that is, some of the coefficients may be zero. This is the reason why we have to do the t-Test to check the existence of statistically significant relationship between the individual explanatory variables and the response variable.

Note: We usually don’t worry about the *p*-value for Constant. It has to do with the “intercept” of the model and seldom has any practical meaning unless it makes sense for all the independent variables to be zero simultaneously.

1. Statistical significance of individual variables in MLR: t-Test

Within a MLR model, we may want to know whether a particular independent variable is making a useful contribution to the model. That is, given the presence of the other independent variables in the model, does a particular variable help us predict or explain the dependent variable?

For a given independent variable, the t-statistic evaluates whether or not there is significant association between the independent and the dependent variable, that is, whether the beta coefficient of the independent variable is significantly different from zero.

The null and alternative hypotheses in the case of individual independent variables (Xi) and the dependent variable (Y) is stated as-

Null: There is no relationship between independent variable (Xi) and dependent variable Y (bi = 0)

Alternative: There is a relationship between independent variable (Xi) and dependent variable Y (bi ≠ 0)

When we cannot reject the null hypothesis for a particular independent variable in the above, we should say that we do not need that variable (say X1) in the model given that variables X2 and X3 will remain in the model.

In our example, we find that SSC percentage is not significant at 0.05 alpha. This means that for a fixed HSC percentage and Degree percentage, changes in the SSC percentage will not significantly affect MBA percentage.

If we remove ssc\_p from the model, we will have the regression equation as below.

**#** Remove insignificant independent variable

mlrmodel1 <- lm(mba\_p ~ hsc\_p + degree\_p, train)

summary(mlrmodel1)

The regression model is given by-

mba\_p = 36.439 + 0.142\*hsc\_p + 0.246\*degree\_p

A good regression model should include only significant independent variables. It is not always clear what will happen when we add or remove variables from a model. Therefore, we should not consider dropping all insignificant variables at one time, but rather take a more structured approach. Automated methods like forward selection, backward elimination, stepwise regression, and best subsets methods facilitate this process. Such procedures ensure that only statistically significant variables for a given alpha are include in the model.

1. Coefficient of Multiple Determination (R-Square) and adjusted R-square

R-square measures the proportion of variation in the dependent variable explained by the model. A problem with R-square is that it will always increase when more variables are added to the model even when there is no statistically significant relationship between the independent variable and the dependent variable. A solution to this is to adjust the value of R-square by taking into account the number of predictors. The Adjusted R-square reflects both the number of independent variables and the sample size and may increase or decrease when an independent variable is added or dropped, thus giving an indication of the value of adding or removing the independent variables in the model. An increase in Adjusted R-square indicates that the model has improved. The Adjusted R-square value will always be less than or equal to R-square value.

In our example, the Adjusted R-square of 0.2328 means that 23.28% of the variability in MBA percentage is explained by the model. (This is better than the R-square of approximately 16% in the SLR model)

1. Residual analysis

Residual analysis is important for checking assumptions about the functional form of the regression model, normal distribution of the residuals and homoscedasticity.

**#** Residual analysis

plot(mlrmodel1)

In our example, based on the plots we can infer that all the assumptions are satisfied and the data don’t present any outliers and influential observations in the MLR model.

1. Multi-collinearity and Variance Inflation Factor

When the data set has a large number of independent variables, it is possible that few of these independent variables may be highly correlated. Existence of high correlation between independent variables is called multi-collinearity. Presence of multi-collinearity can destabilise the MLR model. Due to presence of multi-collinearity it is possible that a statistically significant variable may be labelled as insignificant on account of inflated p-value or the sign of a regression coefficient may change. Thus, it is necessary to identify the presence of multi-collinearity and take corrective action.

Multi-collinearity can be assessed by computing a score called the variance inflation factor (VIF) for each independent variable. The smallest possible value of VIF is one (absence of multi-collinearity). As a rule of thumb, a VIF value that exceeds 5 or 10 indicates a problem. VIF greater than this requires further investigation to assess the impact of multi-collinearity. Remedies to handle multi-collinearity include using Principal Component Analysis (PCA), Ridge regression and LASSO regression.

The R function vif() [car package] can be used to detect multi-collinearity in a regression model.

In our example, the VIF score is within limit.

**#** VIF

car::vif(mlrmodel1)

**Making predictions on the validation dataset**

After the model is built and has passed all diagnostic tests, we can apply the model on the validation dataset to predict.

**#** Prediction on test data

mbapred <- predict(mlrmodel1, test)

data.frame(test$mba\_p, mbapred)

**Cross-validation**

How do we know that an estimated regression model is generalizable beyond the sample data used to fit it? One way is to partition the sample data into a training (or model-building) set, which we can use to develop the model, and a validation (or prediction) set, which is used to evaluate the predictive ability of the model. This is called cross-validation. Cross-validation refers to a set of methods for measuring the performance of a given predictive model on new test data sets.

The different cross-validation methods for assessing model performance are

* The validation set approach
* Leave-one-out cross-validation
* k-fold cross-validation
* Repeated k-fold cross-validation

Validation set approach involves randomly dividing the available set of observations into two parts, a training set and a validation set (hold-out set). The model is fit on the training set, and the fitted model is used to predict the responses for the observations in the validation set. The resulting validation set error rate – typically assessed using Root Mean Square Error (RMSE) provides an estimate of the test error rate.

Sum of Square Errors (SSE) =

Mean Square Error (MSE) =

Root Mean Square Error (RMSE) =

(Note: R calls this as Residual Standard Error (RSE). RSE, also known as the model sigma, is a variant of the RMSE adjusted for the number of predictors in the model. Instead of n in the denominator in the MSE formula, it will be the degrees of freedom which is n-p-1 where p is the number of predictors.

The lower the RSE, the better the model. In practice, the difference between RMSE and RSE is very small, particularly for large multivariate data.)

If the predictions obtained using the model are very close to the actual values in the validation set, then RMSE will be small, and we can conclude that the model fits the validation set very well. The lower the RMSE, the better the model.

Dividing the RMSE by the average value of the outcome variable will give us the prediction error rate, which should be as small as possible

**#** cross-validation

mse <- mean((test$mba\_p - mbapred)^2)

rmse <- sqrt(mse)

rmse

prederror <- rmse/mean(test$mba\_p)

prederror

**Logistic Regression**

Classification problems are an important category of problems in analytics in which the response variable takes a discrete value. In classification problems, the primary objective is to predict the class (or class probability) based on the values of explanatory variables or predictors. There are several techniques used for solving classification problems such as logistic regression, classification trees, discriminant analysis, K-nearest neighbour etc.

Logistic regression is a statistical technique for finding existence of a relationship between a qualitative (discrete) dependent variable (outcome or response variable) and several independent variables (explanatory variables or predictors). Logistic regression is one of the powerful supervised learning algorithms for solving classification problems. Logistic regression does not return directly the class of observations. It allows us to estimate the probability (P) of class membership. The probability will range between 0 and 1. We need to decide the threshold probability at which the category flips from one to the other. By default, this is set to p = 0.5 in many software’s, but in reality it should be set based on the purpose of analysis. We will focus on binary logistic regression in which the response variable (Y) takes only two values – either 1 (positive outcome) or 0 (negative outcome). The binary logistic regression model is given by-

where, Z = b0 + b1\*X1 + b2\*X2 + …. + bp\*Xp

Here X1, X2 … Xp are independent variables.

One of the objectives of classification problems is to predict the class probability, that is the probability that an observation will belong to a particular class. The above equation gives the class probability of an observation belonging to class which is labelled as 1 (i.e P(Y=1)). The logistic function will produce an S-shaped curve.

There are other algebraically equivalent ways to write the logistic regression model. The first is,

The quantity is known as odds. The odds reflect the likelihood that the event will occur. It can be seen as the ratio of “successes” to “non-successes”. For example, if you are at a racetrack and there is an 80% chance that a certain horse will win the race, then his odds are 0.80 / (1 - 0.80) = 4, or 4:1.

The second is,

which states that the (natural) logarithm of the odds is a linear function of the *X* variables (and is often called the log odds). The equation is known as logit (**log**istic probability un**it**) function.

In case of logistic regression, the regression parameters are estimated using Maximum Likelihood Estimator (MLE). We cannot use the OLS method since the residuals in logistic regression will not follow a normal distribution.

**Preparing the data**

**#** select only relevant columns

colnames(placement.df)

placement.lr <- placement.df %>% select(ends\_with("\_p"), -etest\_p, status)

colnames(placement.lr)

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)

**Dividing the data into training and validation data sets**

**#** 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))

**Building the model - Simple logistic regression**

The simple logistic regression is used to predict the probability of class membership based on one single predictor variable. The following code builds a model to predict the probability of a student being Not Placed (Positive or 1) based on the Degree percentage.

The R function glm(), for generalized linear model, is used to compute logistic regression. We need to specify the option family = binomial, which tells R that we want to fit logistic regression.

**#** Train the model

model.lr <- glm(status ~ degree\_p, family = binomial, data = train.lr)

summary(model.lr)

The summary() function is used to access particular aspects of the fitted model such as the coefficients and their p-values.

The estimated values of the coefficients are b0 = 11.43688 and b1 = –0.18851. the logistic function is written as

**Interpretation of logistic regression parameters**

(Remember in linear regression, b1 gives the average change in Y associated with one-unit increase in X).

In logistic regression model, increasing X by one-unit changes the log odds by b1. In general, the interpretation of the regression coefficient b1 in logistic regression is that-

* When b1 is positive, the probability (P(Y=1)) increases as the value of the predictor X1 increases
* When b1 is negative, the probability (P(Y=1)) decreases as the value of the predictor X1 increases

From the output, the coefficient for degree percentage is negative (– 0.18851). Thus the probability of student being Not Placed (P(Y=1)) will decrease as the Degree percentage increases.

**Making predictions**

Once the coefficients have been estimated, we can compute the probability of a student being Not Placed for any given degree percentage.

For example, using the logistic function arrived at, we can predict the probability of a student being Not Placed with a degree percentage of 55% as 0.7291 i.e. 72.91%

In contrast, the predicted probability of a student being Not Placed who has scored 75% in degree is much lower, and equals 0.05679 i.e. 5.679%.

We will make predictions using the test data in order to evaluate the performance of our logistic regression model. The predict() function can be used to predict the probability for given values of predictors. The type = response option tells R to output probabilities of the form P(Y=1), as opposed to other information such as the logit.

Note: If no dataset is supplied to the predict() function, then the probabilities are computed for the training data that was used to build the logistic model.

**#** 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)

**Classification Table – Confusion Matrix**

The classification table in a logistic regression model output is a table that provides accuracy of the model, that is, the accuracy of classifying positives and negatives for a chosen classification cut-off probability. The classification table is also known as confusion matrix or error table.

**#** Classification Table

# categorize into groups based on the predicted probability

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

rownames(conf.matrix) <- c("Placed", "Not Placed")

colnames(conf.matrix) <- c("Placed", "Not Placed")

addmargins(conf.matrix)

We took the value of the cut-off probability as 0.5, it is a decision to be taken by the decision maker to enable the classification. There are different approaches for deriving the optimal classification cut-off probability. When the probability is greater than or equal to 0.5, the observation is classified as positive (coded as 1) and when the probability is less than 0.5, the observation is classified as negative (coded as 0).

In our example, the test data had 30 Placed (negative) cases and 13 Not Placed (positive) cases. The model has predicted 34 Placed (negative) cases and 9 Not Placed (positive) cases. The confusion matrix is as shown below.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | **Predicted** | | **Total** |
| **Placed**  **(0)** | **Not Placed (1)** |
| **Actual** | **Placed (0)** | 30 | 0 | 30 |
| **Not Placed (1)** | 4 | 9 | 13 |
| **Total** | | 34 | 9 | 43 |

The diagonal elements of the confusion matrix indicate correct predictions. Hence, our model accuracy in classifying Placed (negatives) is 100%, whereas the accuracy of classifying Not Placed (positives) is 69.23%. The overall accuracy of the logistic regression model is 90.70%. The misclassification error rate is 9.3% (4 observations are misclassified out of 43 observations).

**#** model accuracy

mean((test.lr$status == lr.pred.class))

The classification accuracy will change when the cut-off probability is changed. For example, when the cut-off probability is 0.35, the confusion matrix is as shown below.

**#** different cut-off

lr.pred.class1 <- ifelse(lr.pred>=0.35, 1, 0)

conf.matrix1 <- table(test.lr$status, lr.pred.class1)

conf.matrix1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | | **Predicted** | | **Total** |
| **Placed**  **(0)** | **Not Placed (1)** |
| **Actual** | **Placed (0)** | 27 | 3 | 30 |
| **Not Placed (1)** | 2 | 11 | 13 |
| **Total** | | 29 | 14 | 43 |

The overall accuracy has decreased from the earlier 90.70% to 88.37%. However, the accuracy of classifying Not Placed (positives) has increased from the earlier 69.23% to 84.61%. In the given context, we need higher accuracy in predicting the Not Placed students than the Placed students. The classification cut-off of 0.35 is better than 0.50 even though the overall classification accuracy has reduced. In general, for a classification problem, the model selection cannot be based on overall accuracy. The accuracy paradox in classification problem states that a model with higher overall accuracy may not be a better model.

**Sensitivity, Specificity, and Precision**

The ability of a model to correctly classify positives and negatives are called sensitivity and specificity, respectively. Sensitivity is also called as recall. There is a trade-off between sensitivity and specificity in most classification problems. Precision measures how good the model is at assigning positives to the positive class.

|  |  |  |  |
| --- | --- | --- | --- |
|  | | **Predicted** | |
| **Negative**  **(0)** | **Positive**  **(1)** |
| **Actual** | **Negative (0)** | True Negative | False Positive |
| **Positive (1)** | False Negative | True Positive |

True Positive (TP) is the number of positives correctly classified by the model. True Negative (TN) is the number of negatives correctly classified by the model. False Negative (FN) is the number of positives misclassified as negatives by the model. False Positive (FP) is the number of negatives misclassified as positive by the model.

In our example, when the cut-off probability is 0.35, sensitivity is 84.61%, specificity is 90%, precision is 78.57%, and the overall accuracy is 88.37%.