Application Development Laboratory (CS 33002)

KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY

School of Computer Engineering

Strictly for internal circulation (within KIIT) and reference only. Not for outside circulation without permission

2 Credit

Analytics Application Development using R

Lab Contents

Sr #	Major and Detailed Coverage Area	Lab#
Predictive Analytics		9
1	Linear Regression	
2	Multiple Regression	
3	Logistic Regression	
4	Non-Linear Regression	
5	Support Vector Machine	
6	Naïve Bayes Classifier	

Regression

- Regression analysis is a very widely used statistical tool to establish a relationship model between two variables.
- Regression helps investment and financial managers to value assets and understand the relationships between variables, such as commodity prices and the stocks of businesses dealing in those commodities.
- The two basic types of regression are **linear regression** and **multiple linear regression**, although there are non-linear regression methods for more complicated data and its analysis.
- **Linear regression** uses one independent variable to explain or predict the outcome of the dependent variable.
- **Multiple linear regression** uses two or more independent variables to predict the outcome of the dependent variable.

Regression cont…

The general form of each type of regression is:

- **Linear regression:** $Y = a + b * X + e$
- **Multiple regression:** $Y = a + b_1 * X_1 + b_2 * X_2 + b_3 * X_3 + \ldots + b_t * X_t + e$

where:

Y = the variable that you are trying to predict (dependent variable).

$X, X_1 \ldots$ = the variable that you are using to predict (independent variable).

a = the intercept.

$b, b_1, b_2 \ldots$ = the slope.

e = the regression residual.

Linear Regression

Year	Sales (Million Euro)	Advertising (Million Euro)
1	651	23
2	762	26
3	856	30
4	1,063	34
5	1,190	43
6	1,298	48
7	1,421	52
8	1,440	57
9	1,518	58

Source: Italian clothing company Benetton

- Each row in the table shows Benetton's sales for a year and the amount spent on advertising that year.
- Outcome of interest is sales i.e. it is what we want to predict.
- Dependent variable = **sales** and independent variable = **advertising**
- Let's assume **Sales = 168 + 23 * Advertising**
  - a = the intercept = 168
  - b = the slope = 23

Interpret

- If advertising expenditure is increased by one Euro, then sales will be expected to increase by 23 million Euro.
- If there was no advertising we would expect sales of 168 million Euro.

Residual Analysis in Regression

- Because a linear regression model is not always appropriate for the data, you should assess the appropriateness of the model by defining residuals and examining residual plots.
- The difference between the observed value of the dependent variable (y) and the predicted value ($\hat{y}$) is called the residual (e). Each data point has one residual.

  Residual = Observed value - Predicted value

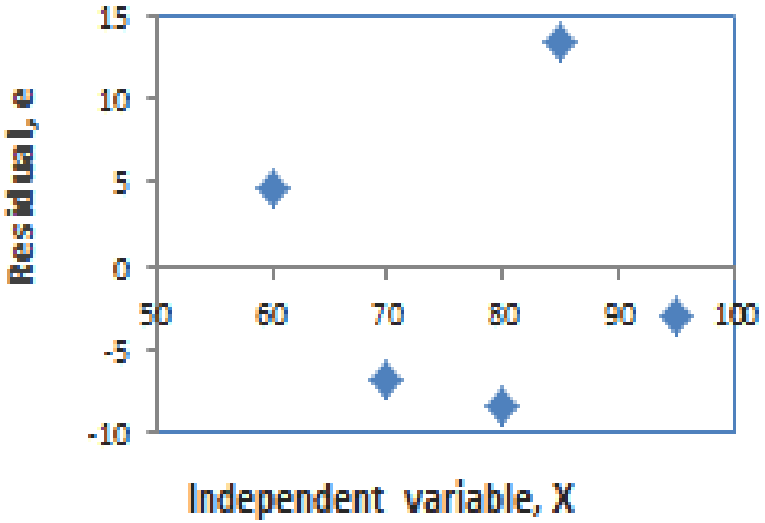
  $e = y - \hat{y}$
- Both the sum and the mean of the residuals are equal to zero.

Residual Plots

- A residual plot is a graph that shows the residuals on the vertical axis and the independent variable on the horizontal axis.
- If the points in a residual plot are randomly dispersed around the horizontal axis, a linear regression model is appropriate for the data; otherwise, a non-linear model is more appropriate.
- **Example:**

x	y	$\hat{y}$	e
60	70	65.411	4.589
70	65	71.849	-6.849
80	70	78.288	-8.288
85	95	81.507	13.493
95	85	87.945	-2.945

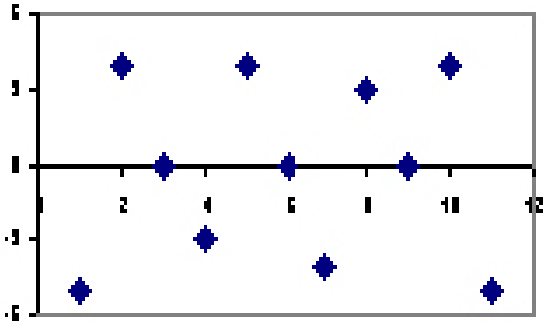
Residual Plots cont…



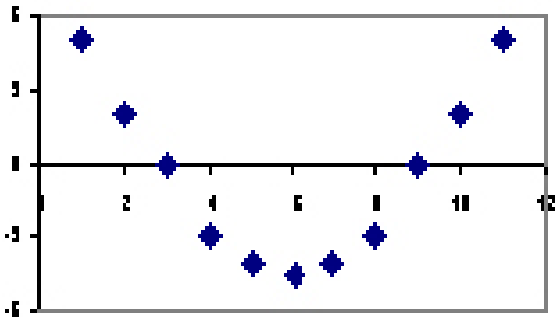
- The residual plot shows a fairly random pattern - the first residual is positive, the next two are negative, the fourth is positive, and the last residual is negative.
- This random pattern indicates that a linear model provides a decent fit to the data.

Residual Plots cont…

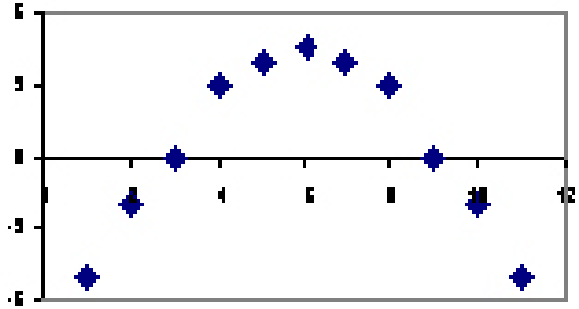
- Below, the residual plots show three typical patterns.
- The first plot shows a random pattern, indicating a good fit for a linear model.
- The other plot patterns are non-random (U-shaped and inverted U), suggesting a better fit for a non-linear model.



Random pattern



Non-random: U-shaped



Non-random: Inverted U-shaped

Steps to Establish a Regression

Consider a simple example of regression is predicting weight of a person when his height is known. To do this we need to have the relationship between height and weight of a person. The steps to create the relationship is

- Carry out the experiment of gathering a sample of observed values of height and corresponding weight.
- Create a relationship model using the **lm()** functions in R.
- Find the coefficients (i.e. intercept and slope) from the model created and create the mathematical equation using these.
- Get a summary of the relationship model to know the average error (residuals) in prediction.
- To predict the weight of new persons, use the **predict**() function in R.

Linear Regression Demonstration

Values of height

151, 174, 138, 186, 128, 136, 179, 163, 152, 131

Values of weight.

63, 81, 56, 91, 47, 57, 76, 72, 62, 48

lm function: This function creates the relationship model between the predictor and the response variable. The basic syntax for lm() function in linear regression is - **lm(formula, data)** where formula is a symbol presenting the relation between x and y, and data is the vector on which the formula will be applied.

Code:

```
x <- c(151, 174, 138, 186, 128, 136, 179, 163, 152, 131)
y <- c(63, 81, 56, 91, 47, 57, 76, 72, 62, 48)
relation <- lm(y~x) # Apply the lm() function.
print(relation)
```

Linear Regression Demonstration cont…

 # Get the Summary of the Relationship

print(summary(relation))

predict Function:

The basic syntax for predict() in linear regression is predict(object, newdata) where object is the formula which is already created using the lm() function, and newdata is the vector containing the new value for predictor variable.

Find weight of a person with height 170.

a <- data.frame(x = 170)

result <- predict(relation, a)

print(result)

Plot the chart.

plot(y, x, col = "blue", main = "Height & Weight Regression",

abline(lm(x~y)),cex = 1.3, pch = 16, xlab = "Weight in Kg", ylab = "Height in cm")

Multiple Regression

Multiple regression is an extension of linear regression into relationship between more than two variables. In simple linear relation we have one predictor and one response variable, but in multiple regression we have more than one predictor variable and one response variable.

Multiple regression: $Y = a + b_1 * X_1 + b_2 * X_2 + b_3 * X_3 + \ldots + b_t * X_t + e$

where:

Y = the variable that you are trying to predict (dependent variable).

X_1, X_2 … = the variable that you are using to predict (independent variable).

a = the intercept.

b_1, b_2 … = the slope.

e = the regression residual.

Multiple Regression cont...

lm function: This function creates the relationship model between the predictor and the response variable and the basic syntax for multiple regression is – **lm(y ~ X_1+X_2+X_3..., data)** where formula is a symbol presenting the relation between the response variable and predictor variables, and data is the vector on which the formula will be applied.

Example: Consider the data set "mtcars" available in the R environment. It gives a comparison between different car models in terms of mileage per gallon (mpg), cylinder displacement("disp"), horse power("hp"), weight of the car("wt") and some more parameters. The goal of the model is to establish the relationship between "mpg" as a response variable with "disp", "hp" and "wt" as predictor variables. We create a subset of these variables from the mtcars data set for this purpose.

```
input <- mtcars[,c("mpg","disp","hp","wt")]
print(head(input))
```

Multiple Regression cont...


```
model <- lm(mpg~disp+hp+wt, data = input) # Create the relationship model.
print(model) # Show the model.

# Get the Intercept and coefficients as vector elements.
cat("# # # # The Coefficient Values # # # ","\n")
a <- coef(model)[1] # Get the intercept.
print(a) # Show the intercept.

bDisp <- coef(model)[2] # Get the slope for displacement.
bHp <- coef(model)[3] # Get the slope for horse power.
bWt <- coef(model)[4] # Get the slope for weight of the car.

print(bDisp) # Display the slope for displacement.
print(bHp) # Display the slope for horse power.
print(bWt) # Display the slope for weight of the car.
```

Multiple Regression cont...

Based on the above intercept and coefficient values, we create the mathematical equation.

$Y = a + bDisp * X_1 + bHp * X_2 + bWt * X_3$

or

$Y = 37.15 + (-0.000937) * X_1 + (-0.0311) * X_2 + (-3.8008) * X_3$

We can use the regression equation created above to predict the mileage when a new set of values for displacement, horse power and weight is provided.

For a car with disp = 221, hp = 102 and wt = 2.91 the predicted mileage is −

$Y = 37.15 + (-0.000937) * 221 + (-0.0311) * 102 + (-3.8008) * 2.91 = 22.7104$

Sessional Exercise

Predict the mileage when a new set of values for displacement = 259, horse power = 132 and weight = 4.52 using predict function.

Logistic Regression

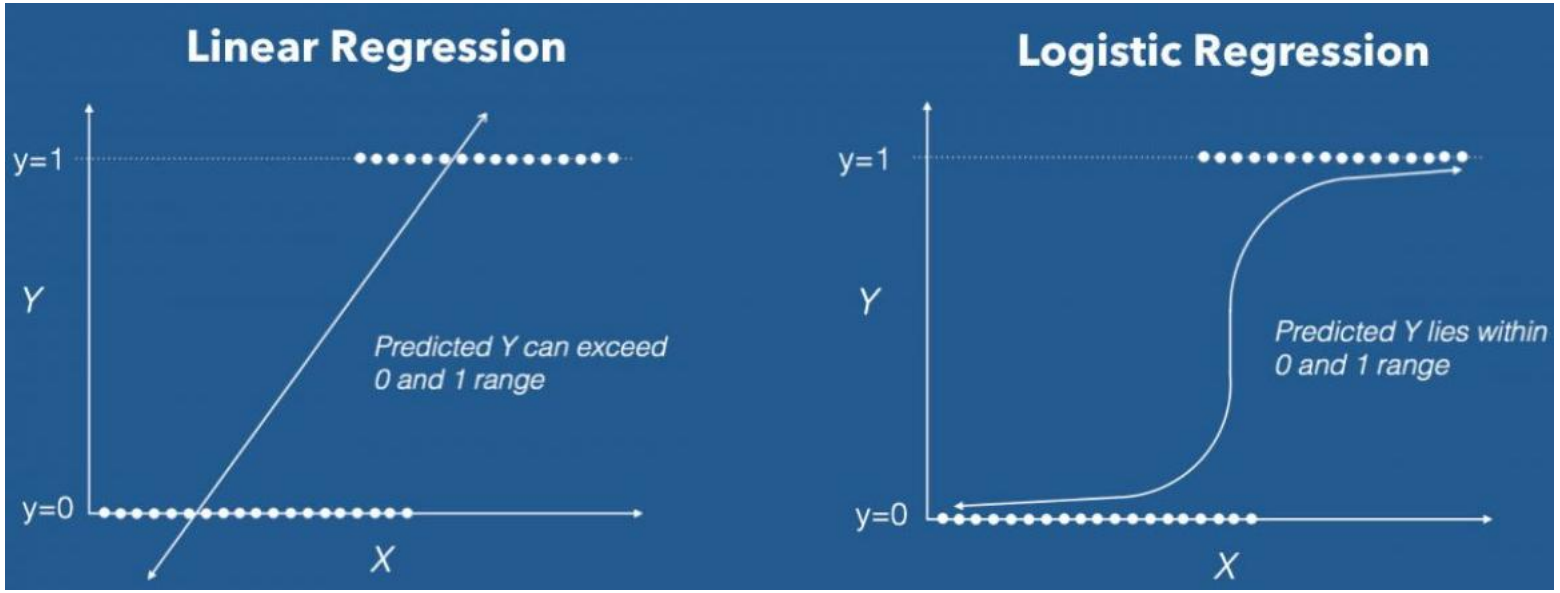
- In linear regression the Y variable is always a continuous variable. If suppose, the Y variable was categorical, you cannot use linear regression model it.
- So what would you do when the Y is a categorical variable with 2 classes?
- Logistic regression can be used to model and solve such problems, also called as binary classification problems.
- The Logistic Regression is a regression model in which the response variable (dependent variable) has categorical values such as True/False or 0/1.
- A key point to note here is that Y can have 2 classes only and not more than that. If Y has more than 2 classes, it would become a multi class classification and you can no longer use the vanilla logistic regression for that.

Real world examples:

- Spam Detection : Predicting if an email is spam or not
- Credit Card Fraud : Predicting if a given credit card transaction is fraud or not
- Health : Predicting if a given mass of tissue is benign or malignant
- Marketing : Predicting if a given user will buy an insurance product or not
- Banking : Predicting if a customer will default on a loan.

Logistic Regression cont…

- When the response variable has only 2 possible values, it is desirable to have a model that predicts the value either as 0 or 1 or as a probability score that ranges between 0 and 1.
- Linear regression does not have this capability. Because, If you use linear regression to model a binary response variable, the resulting model may not restrict the predicted Y values within 0 and 1.



Logistic Regression cont…

- This is where logistic regression comes into play. In logistic regression, you get a probability score that reflects the probability of the occurrence of the event. An event in this case is each row of the training dataset. It could be something like classifying if a given email is spam, or mass of cell is malignant or a user will buy a product and so on.

The general mathematical equation for logistic regression is –

$$Y = \frac{1}{1 + e^{-(a+b1*X1+b2*x2+b3*x3+\cdots)}}$$

Where
Y is the response variable
X_1, X_2 … = the variable that you are using to predict (independent variable)
a = the intercept.
b_1, b_2, … = the slope.

Logistic Regression cont...

Syntax:

The basic syntax for glm() function in logistic regression is – **glm(formula, data, family)** wherein formula is the symbol presenting the relationship between the variables, data is the data set giving the values of these variables, and family is R object to specify the details of the model. It's value is binomial for logistic regression.

Example:

The in-built data set "mtcars" describes different models of a car with their various engine specifications. In "mtcars" data set, the transmission mode (automatic or manual) is described by the column am which is a binary value (0 or 1). We can create a logistic regression model between the columns "am" and 3 other columns - hp, wt and cyl.

```
input <- mtcars[,c("am","cyl","hp","wt")] # Select some columns form mtcars.
print(head(input))
```

Logistic Regression cont…


```
am.data = glm(formula = am ~ cyl + hp + wt, data = input, family = binomial)
print(summary(am.data))
```

Sessional Exercise

- Develop the prediction function with new dataset.
- Install the package "mlbench" and build the logistic regression model for breast cancer.

Non-Linear Regression

Self-practice

Support Vector Machine

Self-study and self-practice

Naïve Bayes Classifier

Self-study and self-practice

Thank You
End of Lab 9

Lab Experiments

1. Search and download at least 3 dataset related to linear regression. Define the problem statement. WAP to demonstrate the linear regression model.
2. Search and download at least 3 dataset related to multiple regression. Define the problem statement. WAP to demonstrate the multiple regression model.
3. Search and download at least 3 dataset related to logistic regression. Define the problem statement. WAP to demonstrate the logistic regression model.
4. Search and download a dataset related to SVM. Define the problem statement. WAP to demonstrate the model.
5. Use IRIS dataset to demonstrate Naïve Bayes Classifier (NBC).
6. Search and download a dataset related to Non-Linear Regression. Define the problem statement. WAP to demonstrate the model.