Shuvam Sanyal

PRN:-19070243015

Linear Regression Assignment

**R Code:-**

library(ggplot2)

library(readr)

library(tidyverse)

library(dplyr)

library(gridExtra)

library(grid)

library(ggthemes)

library(RColorBrewer)

data1=read.csv('Sales\_Data.csv',header=TRUE)

view(data1)

head(data)

dim(data)

library("Metrics")

library("DAAG")

#Exploring the Linear Regression Asummption

#Visualize the linear relationship between the predictor and response

scatter.smooth(x=data1$TV, y=data1$Sales, main="Sales ~ TV")

scatter.smooth(x=data1$Radio, y=data1$Radio, main="Sales ~ Radio")

#Box Plot To spot any outlier observations in the variable. Having outliers in your predictor can drastically affect the predictions as they can easily affect the direction/slope of the line of best fit.

par(mfrow=c(1, 2)) # divide graph area in 2 columns

boxplot(data1$TV, main="TV", sub=paste("Outlier rows: ", boxplot.stats(data1$TV)$out))

boxplot(data1$Radio, main="Radio", sub=paste("Outlier rows: ", boxplot.stats(data1$Radio)$out))

#Density Plot To see the distribution of the predictor variable. Ideally, a close to normal distribution (a bell shaped curve), without being skewed to the left or right is preferred.

library(e1071)

par(mfrow=c(1, 2)) # divide graph area in 2 columns

plot(density(data1$TV), main="Density Plot: Speed", ylab="Frequency", sub=paste("Skewness:", round(e1071::skewness(data1$TV), 2)))

polygon(density(data1$TV), col="red")

plot(density(data1$Radio), main="Density Plot: Distance", ylab="Frequency", sub=paste("Skewness:", round(e1071::skewness(data1$Radio), 2)))

polygon(density(data1$Radio), col="red")

cor(data1$TV, data1$Radio)

#Creating the Multiple Linear Regression Model

The function used for building linear models is lm(). The lm() function takes in two main arguments, namely: 1. Formula 2. Data. The data is typically a data.frame and the formula is a object of class formula:-

linearmod=lm(Sales~TV+Radio+Newspaper, data=data1)

summary(linearmod)

## # The p Value: Checking for statistical significance

We can consider a linear model to be statistically significant only when both these p-Values are less that the pre-determined statistical significance level, which is ideally 0.05. This is visually interpreted by the significance stars at the end of the row. The more the stars beside the variable’s p-Value, the more significant the variable.

Null and alternate hypothesis

When there is a p-value, there is a hull and alternative hypothesis associated with it. In Linear Regression, the Null Hypothesis is that the coefficients associated with the variables is equal to zero. The alternate hypothesis is that the coefficients are not equal to zero (i.e. there exists a relationship between the independent variable in question and the dependent variable).

t-value

We can interpret the t-value something like this. A larger *t-value* indicates that it is less likely that the coefficient is not equal to zero purely by chance. So, higher the t-value, the better.

*Pr(>|t|)* or *p-value* is the probability that you get a t-value as high or higher than the observed value when the Null Hypothesis (the *β* coefficient is equal to zero or that there is no relationship) is true. So if the *Pr(>|t|)* is low, the coefficients are significant (significantly different from zero). If the *Pr(>|t|)* is high, the coefficients are not significant.

t−Statistic=β−coefficient/Std.Error, When the model co-efficients and standard error are known.

R-Squared tells us is the proportion of variation in the dependent (response) variable that has been explained by this model.

R^2=1−SSE/SST

R square value of greater than 0.85 is considered as goodness of fit.

The Akaike’s information criterion - AIC and the Bayesian information criterion - BIC are measures of the goodness of fit of an estimated statistical model and can also be used for model selection. Both criteria depend on the maximized value of the likelihood function L for the estimated model. The lower the value of both, better the model.

modelSummary <- summary(linearmod) # capture model summary as an object

modelCoeffs <- modelSummary$coefficients # model coefficients

beta.estimate <- modelCoeffs["TV", "Estimate"] # get beta estimate

std.error <- modelCoeffs["TV", "Std. Error"] # get std.error

t\_value <- beta.estimate/std.error # calc t statistic

t\_value

p\_value <- 2\*pt(-abs(t\_value), df=nrow(data1)-ncol(data1)) # calc p Value

p\_value

f\_statistic <- linearMod$fstatistic[1] # fstatistic

#f\_statistic

f <- summary(linearMod)$fstatistic # parameters for model p-value calc

#f

model\_p <- pf(f[1], f[2], f[3], lower=FALSE)

AIC(linearmod)

BIC(linearmod)

# Create Training and Test data –

Then I have divided ,y dataset into a 80:20 sample (training:test), then, built the model on the 80% sample and then use the model thus built to predict the dependent variable on test data.

Doing it this way, we will have the model predicted values for the 20% data (test) as well as the actuals (from the original dataset). By calculating accuracy measures (like min\_max accuracy) and error rates (MAPE or MSE), we can find out the prediction accuracy of the model.

set.seed(100) # setting seed to reproduce results of random sampling

trainingRowIndex <- sample(1:nrow(data1), 0.8\*nrow(data1)) # row indices for training data

trainingData <- data1[trainingRowIndex, ] # model training data

testData <- data1[-trainingRowIndex, ] # test data

# Build the model on training data -

lmMod <- lm(Sales~TV+Radio+Newspaper, data=trainingData) #build the model

SalesPred <- predict(lmMod, testData) # predict sales

#Calculating the Accuracy of the regression model

summary (lmMod)

AIC (lmMod) # Calculate akaike information criterion

actuals\_preds <- data.frame(cbind(actuals=testData$Sales, predicteds=SalesPred))

correlation\_accuracy <- cor(actuals\_preds)

head(actuals\_preds)

min\_max\_accuracy <- mean(apply(actuals\_preds, 1, min) / apply(actuals\_preds, 1, max))

min\_max\_accuracy #Higher the better

mape <- mean(abs((actuals\_preds$predicteds - actuals\_preds$actuals))/actuals\_preds$actuals)

mape #Lower the better

K-Fold Cross Vlaidation

It is important to rigorously test the model’s performance as much as possible. One way is to ensure that the model equation you have will perform well, when it is ‘built’ on a different subset of training data and predicted on the remaining data.

How to do this is? Split your data into ‘k’ mutually exclusive random sample portions. Keeping each portion as test data, we build the model on the remaining (k-1 portion) data and calculate the mean squared error of the predictions. This is done for each of the ‘k’ random sample portions. Then finally, the average of these mean squared errors (for ‘k’ portions) is computed. We can use this metric to compare different linear models.

By doing this, we need to check two things:

1. If the model’s prediction accuracy isn’t varying too much for any one particular sample, and
2. If the lines of best fit don’t vary too much with respect the the slope and level.

install.packages("DAAG")

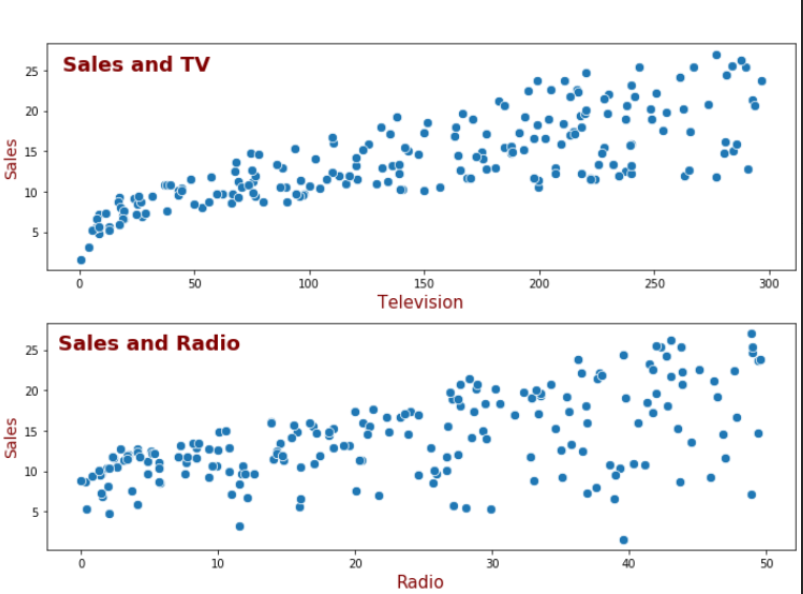
library(DAAG)

cvResults <- suppressWarnings(CVlm(data=data1, form.lm=Sales ~ TV, m=5, dots=FALSE, seed=29, legend.pos="topleft", printit=FALSE, main="Small symbols are predicted values while bigger ones are actuals.")); # performs the CV

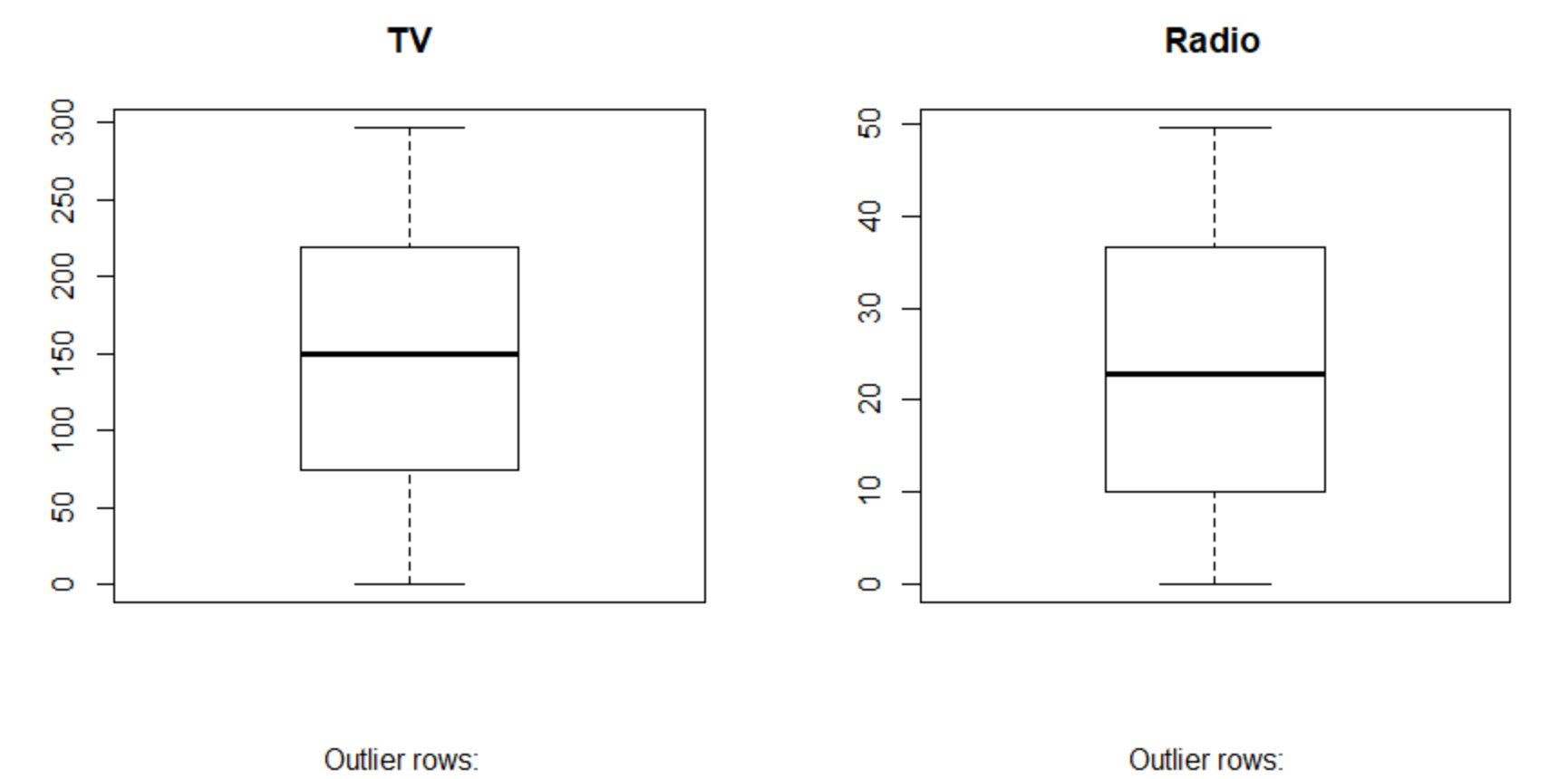
attr(cvResults, 'ms')

**Code Output**

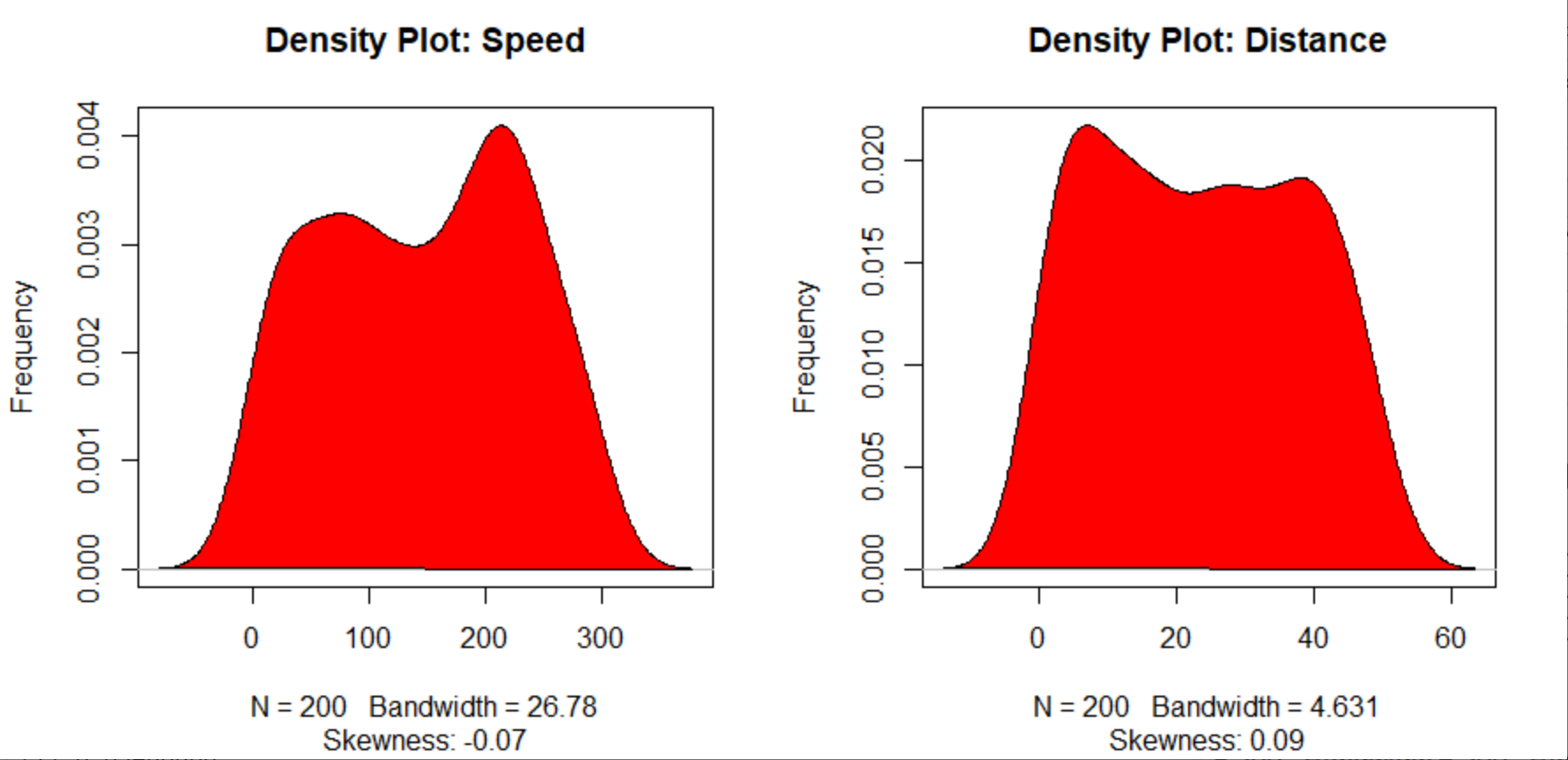
**#Scatter Plot**



**#Box Plot:-**



**#Density Plot**



linearmod=lm(Sales~TV+Radio+Newspaper, data=data1)

> summary(linearmod)

Call:

lm(formula = Sales ~ TV + Radio + Newspaper, data = data1)

Residuals:

Min 1Q Median 3Q Max

-8.8277 -0.8908 0.2418 1.1893 2.8292

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 2.938889 0.311908 9.422 <2e-16 \*\*\*

TV 0.045765 0.001395 32.809 <2e-16 \*\*\*

Radio 0.188530 0.008611 21.893 <2e-16 \*\*\*

Newspaper -0.001037 0.005871 -0.177 0.86

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.686 on 196 degrees of freedom

Multiple R-squared: 0.8972, Adjusted R-squared: 0.8956

F-statistic: 570.3 on 3 and 196 DF, p-value: < 2.2e-16

> modelSummary <- summary(linearmod) # capture model summary as an object

> modelCoeffs <- modelSummary$coefficients # model coefficients

> beta.estimate <- modelCoeffs["TV","Radio","Newspaper", "Estimate"] # get beta estimate for speed

> beta.estimate <- modelCoeffs["TV", "Estimate"] # get beta estimate for speed

> std.error <- modelCoeffs["TV", "Std. Error"] # get std.error for speed

> t\_value <- beta.estimate/std.error # calc t statistic

> p\_value <- 2\*pt(-abs(t\_value), df=nrow(cars)-ncol(cars)) # calc p Value

> f\_statistic <- linearMod$fstatistic[1] # fstatistic

> f <- summary(linearMod)$fstatistic # parameters for model p-value calc

> model\_p <- pf(f[1], f[2], f[3], lower=FALSE)

> t\_value

[1] 32.80862

> p\_value

[1] 1.569568e-34

> f\_statistic

> f <- summary(linearMod)$fstatistic # parameters for model p-value calc

> f

> f\_statistic <- linearMod$fstatistic[1] # fstatistic

> #f\_statistic

> f <- summary(linearMod)$fstatistic # parameters for model p-value calc

> #f

> model\_p <- pf(f[1], f[2], f[3], lower=FALSE)

> AIC(linearmod)

[1] 782.3622

> BIC(linearMod)

> BIC(linearmod)

[1] 798.8538

> # Create Training and Test data -

> set.seed(100) # setting seed to reproduce results of random sampling

> trainingRowIndex <- sample(1:nrow(data1), 0.8\*nrow(data1)) # row indices for training data

> trainingData <- data1[trainingRowIndex, ] # model training data

> testData <- data1[-trainingRowIndex, ] # test data

> # Build the model on training data -

> lmMod <- lm(Sales~TV+Radio+Newspaper, data=trainingData) #build the model

> distPred <- predict(lmMod, testData) # predict distance

> summary (lmMod)

Call:

lm(formula = Sales ~ TV + Radio + Newspaper, data = trainingData)

Residuals:

Min 1Q Median 3Q Max

-8.6571 -0.9273 0.2756 1.1875 2.9522

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 2.744135 0.356547 7.696 1.49e-12 \*\*\*

TV 0.045773 0.001566 29.235 < 2e-16 \*\*\*

Radio 0.188077 0.010113 18.597 < 2e-16 \*\*\*

Newspaper 0.003798 0.006958 0.546 0.586

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.705 on 156 degrees of freedom

Multiple R-squared: 0.8925, Adjusted R-squared: 0.8905

F-statistic: 431.9 on 3 and 156 DF, p-value: < 2.2e-16

> actuals\_preds <- data.frame(cbind(actuals=testData$TV, predicteds=TVPred)) # make actuals\_predicteds dataframe.

> correlation\_accuracy <- cor(actuals\_preds)

> head(actuals\_preds)

> actuals\_preds <- data.frame(cbind(actuals=testData$Sales, predicteds=SalesPred))

> AIC (lmMod) # Calculate akaike information criterion

[1] 630.7707

> actuals\_preds <- data.frame(cbind(actuals=testData$Sales, predicteds=distPred))

> correlation\_accuracy <- cor(actuals\_preds)

> head(actuals\_preds)

actuals predicteds

6 7.2 12.624147

9 4.8 3.536538

17 12.5 13.164084

28 15.9 16.961987

29 18.9 19.316205

33 9.6 7.589283

> min\_max\_accuracy <- mean(apply(actuals\_preds, 1, min) / apply(actuals\_preds, 1, max))

> min\_max\_accuracy

[1] 0.8983165

> mape <- mean(abs((actuals\_preds$predicteds - actuals\_preds$actuals))/actuals\_preds$actuals)

> mape

[1] 0.1135228

> library(DAAG)

Error in library(DAAG) : there is no package called ‘DAAG’

> install.packages(DAAG)

Error in install.packages : object 'DAAG' not found

> install.packages("DAAG")

WARNING: Rtools is required to build R packages but is not currently installed. Please download and install the appropriate version of Rtools before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/

Installing package into ‘C:/Users/Shuvam/Documents/R/win-library/3.6’

(as ‘lib’ is unspecified)

trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.6/DAAG\_1.24.zip'

Content type 'application/zip' length 2105646 bytes (2.0 MB)

downloaded 2.0 MB

package ‘DAAG’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in

C:\Users\Shuvam\AppData\Local\Temp\RtmpsBTFAq\downloaded\_packages

> library(DAAG)

Loading required package: lattice

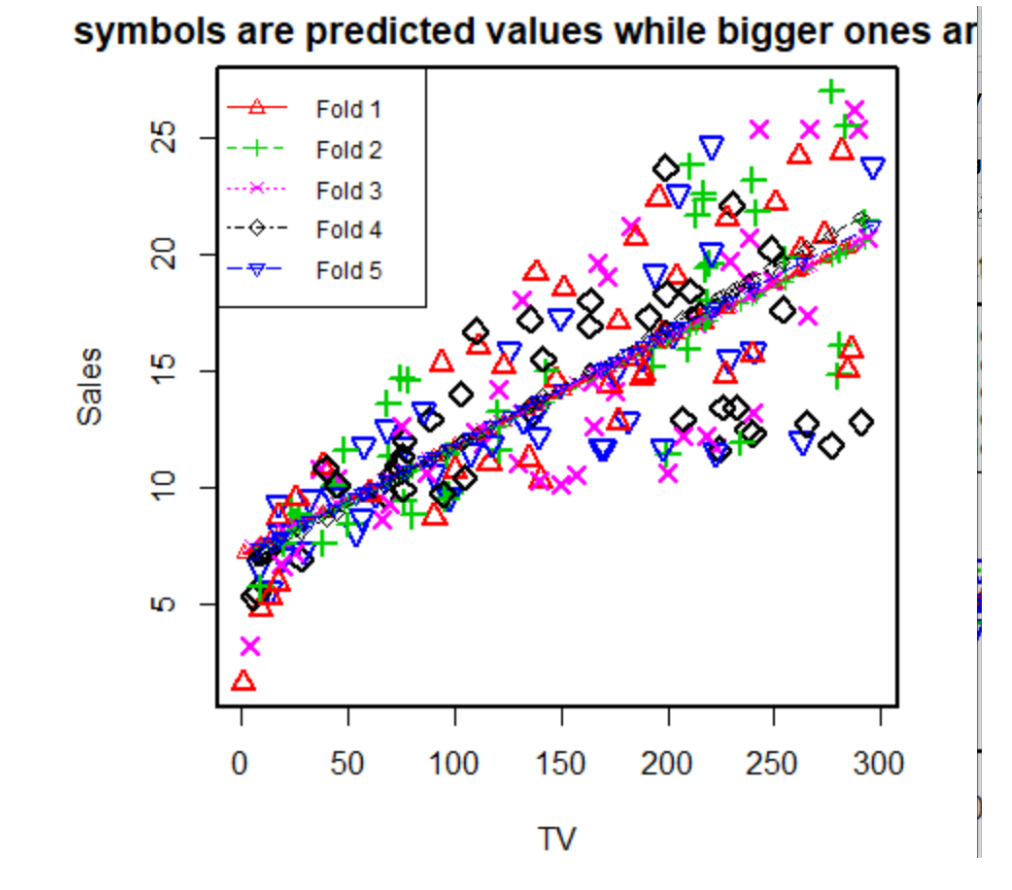
Warning message:

package ‘DAAG’ was built under R version 3.6.3

> results <- suppressWarnings(CVlm(data=data1, form.lm=Sales ~ TV, m=5, dots=FALSE, seed=29, legend.pos="topleft", printit=FALSE, main="Small symbols are predicted values while bigger ones are actuals."))

> attr(cvResults, 'ms')

[1] 10.9055 #MSE



So overcall we can see the linear regression model is performing well on test dataset after running 5 fold cross validation, fetching R Sqaured value close to 0.90. Also from the summart table , we can say out of three independent variable, the TV and Radio has better predictive capabilities of the Sales then Newspaper.