**Practical No:-1**

**Aim:-practical to demonstrate Simple Linear Regression.**

**Code:-**

height<-c(102,117,105,141,135,115,138,144,137,100,131,119,115,121,113)

weight<-c(61,46,62,54,60,69,51,50,46,64,48,56,64,48,59)

student<-lm(weight~height)

student

predict(student,data.frame(height=199),interval="confidence")

plot(student)

**Practical No:-2**

**Aim:- practical to demonstrate multiple linear regression**

Code:-

data(mtcars)

head(mtcars)

summary(mtcars)

model<-lm(mpg~wt+hp,data=mtcars)

summary(model)

new\_data<-data.frame(wt=c(3,2.5),hp=c(110,150))

predictions<-predict(model,newdata=new\_data)

print(predictions)

**PRACTICAL NO:-03**

**Aim :-** Practical to Demonstrate KNN Model

CODE:-

install.packages("ggplot2")

library(class)

library(ggplot2)

data(iris)

iris\_subset <- iris[,c("Sepal.Length","Sepal.Width","Species")]

set.seed(123)

iris\_subset <- iris\_subset[sample(nrow(iris\_subset)),]

train\_index <- 1:round(0.7 \* nrow(iris\_subset))

train\_data <- iris\_subset[train\_index, 1:2]

train\_labels <- iris\_subset[train\_index, 3]

test\_data <- iris\_subset[-train\_index, 1:2]

test\_labels <- iris\_subset[-train\_index, 3]

k <- 3

predicated\_lables<-knn(train=train\_data,test=test\_data,cl=train\_labels,k=k)

test\_data<-as.data.frame(test\_data)

test\_data$predicated<-predicated\_lables

test\_data$Actual<-test\_labels

ggplot(data=test\_data,aes(x=Sepal.Length,y=Sepal.Width))+geom\_point(aes(color=predicated,shape = Actual ),size=3)+scale\_color\_manual(values = c("red","green","blue"))+labs(title="KNN Classification Results",x="Sepal Length",y="Sepal Width")+theme\_minimal()

**Practical 4**

**Aim: Practical to demonstrate Logistic Regression.**

**Code1:-**

input <- mtcars[,c("am","cyl","hp","wt")]

(head(input))

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

print(summary(am.data))

**Code2:-**

install.packages("dplyr")

library(dplyr)

summary(mtcars)

install.packages("caTools")

library(caTools)

split <- sample.split(mtcars, SplitRatio = 0.8)

split

train\_reg <- subset(mtcars, split == "TRUE")

test\_reg <- subset(mtcars, split == "FALSE")

logistic\_model <- glm(vs ~ wt + disp,

data = train\_reg,

family = "binomial")

logistic\_model

summary(logistic\_model)

predict\_reg <- predict(logistic\_model,

test\_reg, type = "response")

predict\_reg

predict\_reg <- ifelse(predict\_reg >0.5, 1, 0)

table(test\_reg$vs, predict\_reg)

missing\_classerr <- mean(predict\_reg != test\_reg$vs)

print(paste('Accuracy =', 1 - missing\_classerr))

**Practical 5**

**Aim: Practical to Decision tree classifier.**

**Code:-**

install.packages("party")

library(party)

print(head(readingSkills))

input.dat<-readingSkills[c(1:105),]

png(file="decision\_tree.png")

output.tree<-ctree(nativeSpeaker~age+shoeSize+score,data=input.dat)

plot(output.tree)

dev.off()

**PRACTICAL NO: - 6**

**Aim: -** Practical to demonstrate Principal Component Analysis.

**Code:-**

data("iris")

X<-iris[,1:4]

y<-iris$Species

X\_scaled<-scale(X)

species\_colors<-c("setosa"="red","versicolor"="green","virginica"="blue")

pca<-prcomp(X\_scaled,center=TRUE,scale.=TRUE)

pca\_data<-data.frame(pca$x[,1:2])

colnames(pca\_data)<-c("PC1","PC2")

pca\_data$Species<-y

par(mfrow=c(2,1))

plot(X\_scaled,

col=species\_colors[as.character(y)],

pch=19,

xlab="Feature 1",

ylab="Feature 2",

main = "Before PCA of Iris Dataset")

plot(pca\_data$PC1,pca\_data$PC2,

col=species\_colors[as.character(pca\_data$Species)],

pch=19,

xlab="Principle Component 1",

ylab="Principle Component 2",

main = "After PCA of Iris Dataset")

par(mfrow=c(1,1))

**Practical 7**

**Aim: Practical to demonstrate K-mean Clustering.**

**Code:**

data("iris")

names(iris)

new\_data<-subset(iris,select = c(-Species))

new\_data

cl<-kmeans(new\_data,3)

cl

data <- new\_data

wss <- sapply(1:15,

function(k){kmeans(data, k )$tot.withinss})

wss

plot(1:15, wss,

type="b", pch = 19, frame = FALSE,

xlab="Number of clusters K",

ylab="Total within-clusters sum of squares")

install.packages("cluster")

library(cluster)

clusplot(new\_data, cl$cluster, color=TRUE, shade=TRUE,

labels=2, lines=0)

cl$cluster

cl$centers

**Practical No 8**

**Aim:** Practical to demonstrate Hierarchical clustering.

**Code:-**

"agglomarative clustering "

clusters <- hclust(dist(iris[, 3:4]))

plot(clusters)

clusterCut <- cutree(clusters, 3)

table(clusterCut, iris$Species)

ggplot(iris, aes(Petal.Length, Petal.Width, color = iris$Species)) +

geom\_point(alpha = 0.4, size = 3.5) + geom\_point(col = clusterCut) +

scale\_color\_manual(values = c('black', 'red', 'green'))

clusters <- hclust(dist(iris[, 3:4]), method = 'average')

clusterCut1 <- cutree(clusters, 3)

table(clusterCut1, iris$Species)

plot(clusters)

ggplot(iris, aes(Petal.Length, Petal.Width, color = iris$Species)) +

geom\_point(alpha = 0.4, size = 3.5) + geom\_point(col = clusterCut1) +

scale\_color\_manual(values = c('black', 'red', 'green'))

**Practical 9**

**Aim:** Practical to demonstrate Time Series Analysis and apply the various visualization

Code:-

# Load the AirPassengers dataset

data("AirPassengers")

# Convert it into a time series object

ts\_data <- AirPassengers

# Basic Summary

print("Summary of the dataset:")

summary(ts\_data)

# 1. Plot the Original Time Series

plot(ts\_data,

     main = "Monthly Air Passengers (1949-1960)",

     xlab = "Year",

     ylab = "Number of Passengers",

     col = "blue")

# 2. Seasonal Boxplot

boxplot(ts\_data ~ cycle(ts\_data),

        main = "Seasonal Boxplot of Air Passengers",

        xlab = "Month",

        ylab = "Number of Passengers",

        col = "lightblue")

# 3. Forecasting using ARIMA

library(forecast)

# Fit an ARIMA model and forecast

fit <- auto.arima(ts\_data)

forecasted <- forecast(fit, h = 12)  # Forecast for next 12 months

# Plot the Forecast

plot(forecasted,

     main = "Forecast of Air Passengers",

     xlab = "Year",

     ylab = "Number of Passengers")

**Practical 10**

**Aim: Practical to demonstrate Support Vector Machine (SVM).**

**Code:-**

library(e1071)

data(iris)

head(iris)

svm\_model<-svm(Species~Sepal.Length+Sepal.Width,data=iris)

summary(svm\_model)

predictions<-predict(svm\_model,iris)

conf\_matrix<-table(predictions,iris$Species)

print(conf\_matrix)

accuracy<-sum(diag(conf\_matrix))

cat("Accuracy:",accuracy,"\n")

library(ggplot2)

ggplot(iris,aes(x=Sepal.Length,y=Sepal.Width,color=Species))+

geom\_point()+

geom\_abline(slope=-svm\_model$coefs[1]/svm\_model$coefs[2],

intercept = -svm\_model$rho/svm\_model$coefs[2],

color="red")+

labs(title="SVM Classifier-Sepal Length vs Sepal Width")

**Practical 11**

**Aim:** Given a dataset with missing values and multiple features, perform data normalization and data transformation.

**Code1:-**

df1= data.frame(

A1 = c(NA, 10, NA, 7, 8, 11,20),

A2 = c("A", 9, 3, "B", "C", "D","E"),

A3 = c(1, 0, NA, 1, 1, NA,3)

)

print(df1)

print("After removing the NA values ")

result=na.omit(df1)

print(result)

df1 <- data.frame(

A1 = c(NA, 10, NA, 7, 8, 11,20),

A2 = c("A", 9, 3, "B", "C", "D","E"),

A3 = c(1, 0, NA, 1, 1, NA,3)

)

print(df1)

print("After removing the NA values ")

result=df1[complete.cases(df1),]

print(result)

**Code2:-**

data <- data.frame(

Name = c('Alice', 'Bob', NA, 'David', 'Eva'),

Age = c(24, NA, 22, 29, 25),

Salary = c('50000', '60000', '70000', '80000', NA),

Department = c('HR', NA, 'IT', 'Finance', 'HR'),

stringsAsFactors = FALSE # Ensure strings are not converted to factors

)

cat("Raw Data:\n")

print(data)

data$Name[is.na(data$Name)] <- 'Unknown'

data$Department[is.na(data$Department)] <- 'Unknown'

data$Age[is.na(data$Age)] <- mean(data$Age, na.rm = TRUE)

data$Salary[is.na(data$Salary)] <- '0'

data$Salary <- as.numeric(data$Salary)

cat("\nCleaned Data:\n")

print(data)

data$Salary\_in\_K <- data$Salary / 1000

cat("\nTransformed Data:\n")

print(data)

**PRACTICAL NO: - 12**

**Aim: -** Apply Cross – validation to evaluate the performance of predictive model, and use accuracy metrics such as ROC curves and precision-recall curves to assess the model’s effectiveness.

**Code:-**

library(caret)

library(pROC)

library(PRROC)

data(iris)

iris\_binary<-iris[iris$Species!="setosa",]

iris\_binary$Species<-factor(ifelse(iris\_binary$Species=="versicolor","Class1","Class0"))

set.seed(123)

train\_control<-trainControl(method="cv",number=10,classProbs=TRUE,summaryFunction=twoClassSummary)

model<-train(Species~.,data=iris\_binary,method="glm",family="binomial",metric="ROC",trControl=train\_control)

print(model)

predictions<-predict(model,iris\_binary,type="prob")

roc\_curve<-roc(iris\_binary$Species,predictions$Class1)

plot(roc\_curve,main="ROC Curve(10-Fold CV)",col="blue")

auc\_value<-auc(roc\_curve)

cat("AUC Value:",auc\_value,"\n")

pr<-pr.curve(scores.class0=predictions$Class1[iris\_binary$Species=="Class1"],scores.class1=predictions$Class1[iris\_binary$Species=="Class0"],curve=TRUE)

plot(pr,main="Precision-Recall Curve (10-Fold CV)")

**Practical No 13**

**Aim:-**Demonstrate Feature Selection Technique using python**.**

**Code1:-**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

df=pd.read\_csv(r"C:/Users/ckt/Desktop/employee/cs\_students.csv")

features=df.columns

print(features)

print(df.sample(5))

correlation=np.corrcoef(df['Age'],df['GPA'])

print(correlation)

plt.scatter(df['Age'],df['GPA'])

plt.show()

**Code2:-**

import numpy as npo

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

sns.set(style="whitegrid")

import warnings

warnings.filterwarnings('ignore')

from sklearn.datasets import load\_iris

feature\_names=load\_iris().feature\_names

x\_data=pd.DataFrame(load\_iris().data,columns=feature\_names)

y\_data=load\_iris().target

x\_data.head()

from sklearn.feature\_selection import f\_classif

f\_value=f\_classif(x\_data,y\_data)

for feature in zip(feature\_names,f\_value[0]):

  print(feature)

plt.figure(figsize=(4,4))

plt.bar(x=feature\_names,height=f\_value[0],color='tomato')

plt.xticks(rotation='vertical')

plt.ylabel('F-value')

plt.title('F-value Comparison')

plt.show()

**Aim:** Practical to demonstrate One Way Anova (Analysis of Variance).

**Code:-**

weight\_loss <- c(5.2, 4.8, 6.1, 4.0, 3.8, 4.5, 5.5, 5.8, 6.0)

diet\_type <- factor(rep(c("Low Carb", "Mediterranean", "High Protein"), each = 3))

anova\_result <- aov(weight\_loss ~ diet\_type)

summary(anova\_result)

boxplot(weight\_loss ~ diet\_type,

main = "Weight Loss by Diet Type",

xlab = "Diet Type",

ylab = "Weight Loss (kg)",

col = c("lightblue", "lightgreen", "pink"))

**Practical 15**

**Aim:** Practical to demonstrate Two Way Anova (Analysis of Variance).

**Code:-**

diet <- factor(rep(c("Low Carb", "Mediterranean", "High Protein"), each = 6))

exercise <- factor(rep(c("Cardio", "Strength Training"), each = 3, times = 3))

weight\_loss <- c(5.2, 4.8, 6.1, 5.5, 5.8, 6.0,

4.0, 3.8, 4.5, 4.2, 4.1, 4.3,

5.8, 6.0, 6.2, 6.1, 6.3, 6.4)

data <- data.frame(Diet = diet, Exercise = exercise, WeightLoss = weight\_loss)

anova\_result <- aov(WeightLoss ~ Diet \* Exercise, data = data)

summary(anova\_result)

interaction.plot(data$Exercise, data$Diet, data$WeightLoss,

col = c("blue", "green", "red"),

main = "Interaction Plot",

xlab = "Exercise",

ylab = "Weight Loss")