LINEAR REGRESSION:

```
rm(list=ls())
library(dplyr)
data<-mtcars
train=sample_n(data,15)
plot(train$wt,train$mpg,main='Scatter plot MPG VS WT', xlab ="wt",ylab = "mpg")
cor.test(train$wt,train$mpg)
lm1<-lm(mpg~wt,data=train)
summary(lm1)
abline(lm1,col='blue')
```

Conclusion: if p-value<0.05 then null hypothesis is rejected (null hypothesis: significant linear relationship between the independent variable X and the dependent variable Y, the slope will not equal zero)

TIME-SERIES FORECASTING:

```
library(forecast)
library(tseries)

gold<-read.csv("gold.csv")
gold_ts<-ts(gold$Price,start = min(gold$Month),end=max(gold$Month),frequency = 1)
class(gold_ts)
plot(gold_ts)
acf(gold_ts)
pacf(gold_ts)
adf.test(gold_ts)
gold_model=auto.arima(gold_ts,ic="aic",trace = TRUE)
gold_f = forecast(gold_model,level = c(95),h=24)
gold_f
plot(gold_f)
```

Conclusion:

Dickey-Fuller = -2.3526, Lag order = 3, p-value = 0.4359

alternative hypothesis: stationary

Since p-value > 0.05, we accept the null hypothesis. Hence the time series is non-stationary (data is dependent of time).

From the auto.arima function we get the best fit model with P,D,Q values as : 0,1,0

- frequency = 12 pegs the data points for every month of a year.
- frequency = 4 pegs the data points for every quarter of a year.
- frequency = 6 pegs the data points for every 10 minutes of an hour.

- frequency = 24*6 pegs the data points for every 10 minutes of a day.

Regression and Forecasting:

```
df<-read.csv("weatherHistory2016.csv")
library(dplyr)
library(corrplot)
library(forecast)
library(tseries)
head(df)

ml1<-lm(dt$df.Temperature..C. ~ dt$df.Apparent.Temperature..C. + dt$df.Humidity +
dt$df.Wind.Speed..km.h.+dt$df.Wind.Bearing..degrees.+dt$df.Visibility..km.,data = dt)
summary(ml1)

ml2<-lm(dt$df.Temperature..C. ~ dt$df.Apparent.Temperature..C. + dt$df.Humidity
+dt$df.Wind.Speed..km.h.+dt$df.Wind.Bearing..degrees.,data =dt)summary(ml2)

dt<-
data.frame(df$Temperature..C.,df$Apparent.Temperature..C.,df$Humidity,df$Wind.Speed..km.h.,df$Wi
nd.Bearing..degrees.)

corr<-cor(dt)
corrplot(corr)

date<-df$Formatted.Date
dt<-cbind(dt,date)
dt<-na.omit(dt)
tseries<-ts(dt$df.Temperature..C.,start = as.Date("2016-01-01 00:00"),end=as.Date("2016-12-31
22:59"),frequency = 24)
plot(tseries)

acf(tseries)
pacf(tseries)

adf.test(tseries)
model=auto.arima(tseries,ic="aic",trace = TRUE)

forc = forecast(model,level = c(95),h=24)
plot(forc)
```

ANOVA:

```
df<-read.csv("color-anova-example.csv")
group_by(df,color)%>%
summarise(count=n(),mean=mean(response, na.rm=TRUE))

anova<-aov(response~color,data=df)
summary(anova)

TukeyHSD(anova)
```

Conclusion:
Pr value < 0.05 so Rejecting null hypothesis [=> not all group means are equal]
According to TukeyHSD test, column having least p-adjust value (<0.05) have the most significant difference

LOGISTIC REGRESSION:

```
ad<-read.csv("Social_network_Ads.csv")
ad$Gender<-as.factor(ad$Gender)
ad$Purchased<-as.factor(ad$Purchased)

model<-glm(Purchased~Age+Gender+EstimatedSalary,data = ad,family = 'binomial')
summary(model)
res<-predict(model,ad,type='response')
cfmatrix<-table(Act=ad$Purchased,pred=res>0.6)
cfmatrix
acc=(cfmatrix[[1,1]]+cfmatrix[[2,2]])/sum(cfmatrix)
acc
```

Conclusion: Model summary, confusion matrix, accuracy

KNN:

```
library(class)
library(caTools)

data(iris)
summary(iris)

splitd<-sample.split(iris,SplitRatio
= 0.8)
train <-
```

```
subset(iris,splitd=="TRUE")
test <- subset(iris,splitd=="FALSE")
View(train)
View(test)

norm<- function(x){((x-
min(x))/(max(x)-min(x)))}
norm_train <-
as.data.frame(lapply(train[,1:4],n
orm))
norm_test <-
as.data.frame(lapply(test[,1:4],no
rm))
View(norm_test)

pred<-knn(train = norm_train,
test = norm_test, cl =
train$Species,k=5)
cf <- table(test$Species,pred)
cf

ACC <-
(cf[[1,1]]+cf[[2,2]]+cf[[3,3]])/sum(
cf)
ACC
```

Conclusion: accuracy

K MEANS:

```
dt1<-read.csv("iris.csv")
df<-scale(dt1)
fit<-kmeans(df,centers=2)      #2 clusters
fit$cluster
fit$size
fit$withinss
fit$tot.withinss               # Within Cluster Sum of Squares (WCSS)

Kmax <- 15
WCSS <- rep(NA,Kmax)
nClust <- list()
for (i in
  1:Kmax){ fit<-
```

```
  kmeans(df,i)
  WCSS[i] <- fit$tot.withinss
  nClust[[i]] <- fit$size
}

plot(1:Kmax,WCSS,type="b",pch=19)
library(factoextra)
fviz_nbclust(df, kmeans, method = "wss")
fviz_cluster(fit, dt1)

library(cluster)
fit <- pam(df, 3, metric = "manhattan")          # K-Medoids
print(fit)
```

HEIRARCHICAL CLUSTERING:

```
dt<-read.csv("iris.csv",row.names = 1)
df<-scale(dt)

ed<-dist(df,method = "euclidean")
hier_clust <- hclust(ed, method = 'complete')
hier_clust
plot(hier_clust)

cluster <- cutree(hier_clust, k = 3)
cluster
rect.hclust(hier_clust, k = 3, border = 2:4)
```

GRADIENT DESCENT

```
gd <- function(x, y, m, c, alpha, conv_thr, iter)
 {plot(x, y, col = "blue", pch = 20)
 iterations <- 0
 hf <- 0

while(iterations <= iter)
   {y_p = m*x+c
   hf_new <- sum(y_p-y)^2
   m = m-alpha*sum((y_p-y)*x)
   c = c- alpha*sum(y_p-y)
   if(abs(hf-hf_new) < conv_thr)
     {break
   }
   hf <- hf_new
   iterations = iterations + 1
 }
 return(paste("Optimal intercept:", c, "Optimal slope:", m," Loss funtion", hf," iterations",iterations))
}

data1 <- mtcars
gd(data1$wt, data1$mpg, -0.2, 32, 0.001, 0.00001, 2000)
reg <- lm(data1$mpg~data1$wt)
reg
```

Conclusion: the value obtained from linear regression model and grad_desc are same, at iteration, etc.

MOMENTUM GRADIENT DESCENT:

```
data_mtcars <- mtcars
rm(list = ls())

mgd <- function(x1,x2, y, m1,m2, c, alpha, gamma, iter)
 {iterations <- 0
 u_m1<-0
 u_m2<-0
 u_c<-0

 while(iterations<=iter){ y_pred=m1
  *x1+m2*x2+c loss_new<-
  0.5*sum((y_pred-y)^2)
  nu_m1<-gamma*u_m1+alpha*sum((y_pred-y)*x1)
  nu_m2<-gamma*u_m2+alpha*sum((y_pred-y)*x2)
  nu_c<-gamma*u_c+alpha*sum(y_pred-y)
  m1<-m1-nu_m1
  m2<-m2-nu_m2
  c<-c-nu_c
  u_m1<-nu_m1
  u_m2<-nu_m2
  u_c<-nu_c
  loss<-loss_new
  iterations<-iterations+1
 }
 return(paste("Optimal intercept: ", c, " Optimal slope m1: ", m1, " Optimal slope m2: ", m2," Loss
funtion: ", loss," iterations: ",iterations))
}

mgd(data_mtcars$wt, data_mtcars$hp,data_mtcars$mpg, -0.2, -0.2, 32, 0.000002,0.45 ,20000)

model<- lm(data_mtcars$mpg~data_mtcars$wt+data_mtcars$hp)
model
```

Conclusion: Hence with appropriate alpha, gamma and iteration values we obtain the optimal slope and intercept using the momentum gradient function for the given multi linear regression model