**R code for H1B petition project**

**# Import regression\_datamodel.csv**

reg\_datamodel=read.csv("regression\_datamodel.csv",header = TRUE, sep = ",")

# Print the structure of regression\_datamodel.csv

reg\_datamodel

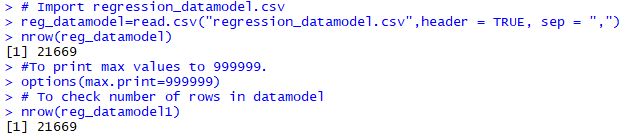
#To print max values to 999999.

options(max.print=999999)

# To check number of rows in datamodel

nrow(reg\_datamodel1)

**Screenshot**



# selection of 80% training data and 20% Test data

split=sample.split(reg\_datamodel$STATUS,SplitRatio=0.8)

# training data

train.data=subset(reg\_datamodel,split==TRUE)

# testing data

test.data=subset(reg\_datamodel,split==FALSE)

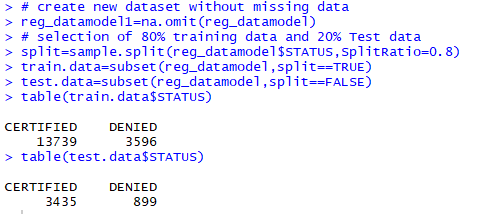
# to specify number of STATUS in training data.

table(train.data$STATUS)

#to specify number of STATUS in testing data.

table(test.data$STATUS)

**Screenshot**



#Defining variables.

STATUS=train.data$STATUS

SOC\_NAME=train.data$SOC\_NAME

POSITION\_TYPE=train.data$POSITION\_TYPE

WAGE=train.data$WAGE

YEAR=train.data$YEAR

REGION=train.data$REGION

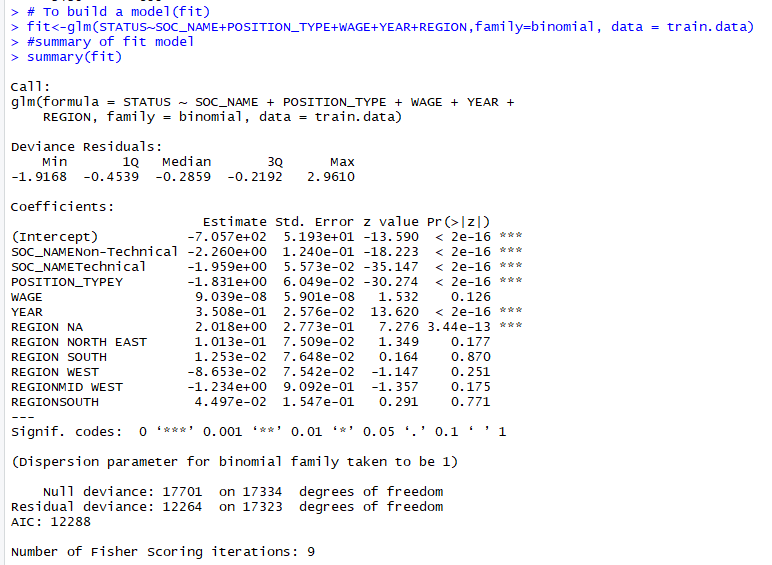
**Building a model**

**# To build a model(fit)**

fit<-glm(STATUS~SOC\_NAME+POSITION\_TYPE+WAGE+YEAR+REGION,family=binomial, data = train.data)

#summary of fit model

summary(fit)



**# Base model on taking one variable wage**

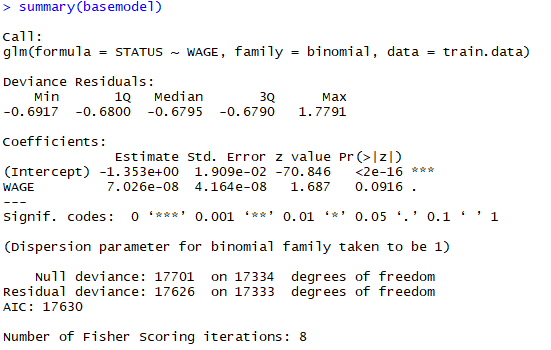
basemodel=glm(STATUS~WAGE,data=train.data,family = binomial)

basemodel

**# summary of Base model.**

summary(basemodel)

**Screenshot**



**# Forward model on Base model**

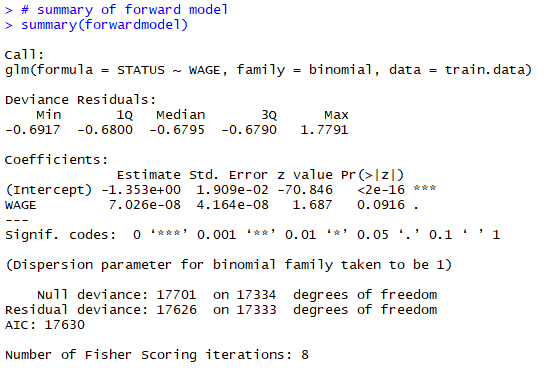
forwardmodel<-step(basemodel,direction = "forward",)

#,trace = F)

forwardmodel

# summary of forward model

summary(forwardmodel)



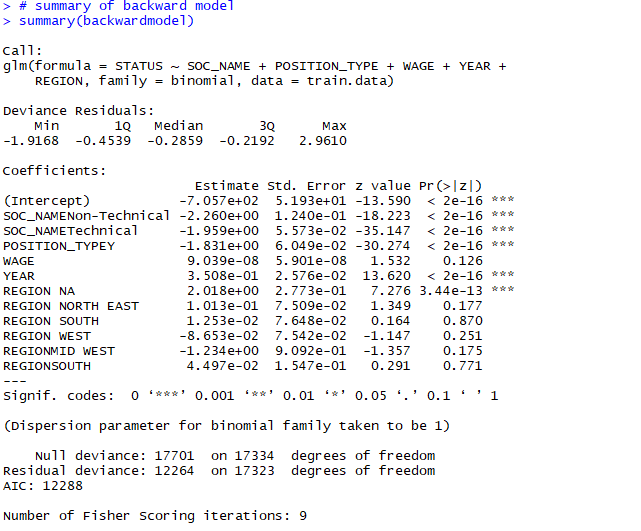
**# Backward model on fit model**

backwardmodel<-step(fit,direction = "backward",trace = T)

backwardmodel

# summary of backward model

summary(backwardmodel)



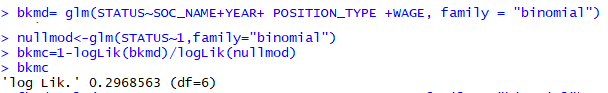
**# To Calculate McFadden's R squared Backward model**

bkmd= glm(STATUS~SOC\_NAME+YEAR+ POSITION\_TYPE +WAGE, family = "binomial")

nullmod<-glm(STATUS~1,family="binomial")

bkmc=1-logLik(bkmd)/logLik(nullmod)

bkmc



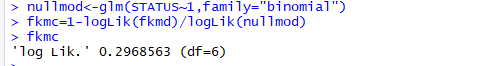
**#To Calculate McFadden's R squared Forward model**

fkmd= glm(STATUS~WAGE+SOC\_NAME+POSITION\_TYPE+YEAR, family = "binomial")

nullmod<-glm(STATUS~1,family="binomial")

fkmc=1-logLik(fkmd)/logLik(nullmod)

fkmc



# **TO predict values based on test data using confusion matrix for backward model**

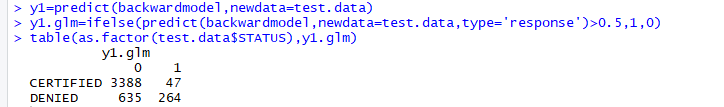
#cut off value as **0.5**

y1=predict(backwardmodel,newdata=test.data)

**#using cut-off value as 0.50**

y1.glm=ifelse(predict(backwardmodel,newdata=test.data,type='response')>0.5,1,0)

table(as.factor(test.data$STATUS),y1.glm)



By adding left diagonal values: 3338+ 264 = **3,652**

Total number of test data: **4334**

Accuracy = left diagonal values/Total number of test data

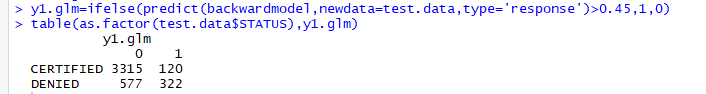
=3652/4334 = **84.26 %**

**# by taking cut off value as 0.45**

**#using cut-off value as 0.45**

y1.glm=ifelse(predict(backwardmodel,newdata=test.data,type='response')>0.45,1,0)

table(as.factor(test.data$STATUS),y1.glm)



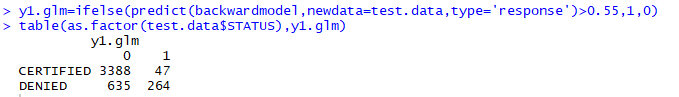
|  |
| --- |
|  |
|  |
| |  | | --- | |  | |

Accuracy is **83.91 %** by taking cut off value as .45

#By taking cutoff value as **.55**

y1.glm=ifelse(predict(backwardmodel,newdata=test.data,type='response')>0.55,1,0)

table(as.factor(test.data$STATUS),y1.glm)



Accuracy is **84.26 %** by taking cut off value as **.55**

Hence we will compare the models with cutoff value at **.50**

# **TO predict values based on test data using confusion matrix for forward model**

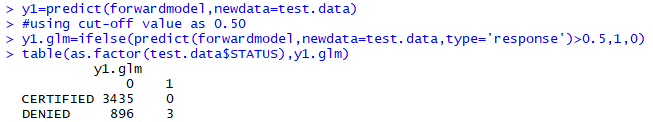
# TO predict values based on test data using confusion matrix forward model

y1=predict(forwardmodel,newdata=test.data)

#using cut-off value as 0.50

y1.glm=ifelse(predict(forwardmodel,newdata=test.data,type='response')>0.5,1,0)

table(as.factor(test.data$STATUS),y1.glm)



Accuracy for forward at cutoff value 0.5 is **79.32%**

By comparison we can say that Model having high accuracy value is the best model in our case Backward model is having high accuracy value compare to forward model hence it is considered as best model.

**Hyposthesis Code**

**# For Electronic Engineer**

# Import electronic engineer.csv

m1 = read.csv("electronic engineer.csv ", header = TRUE, sep = ",")

head(m1)

wage=m1$PREVAILING\_WAGE

#define the variables

wages=m2$PREVAILING\_WAGE

# Calculating Standard deviation

sd(wages)

# removing null values using omit command

m2=na.omit(m1)

# to check if there is any missing values in our newdata set m2

na.fail(m2)

#To print max values to 999999.

options(max.print=999999)

**# For Agriculture Engineers**

# Import electronic engineer.csv

a1=read.csv("agriculture\_engineer.csv",header=TRUE, sep = ",")

head(a1)

#define the variables

wage=a1$PREVAILING\_WAGE

wages=a2$PREVAILING\_WAGE

# Calculating Standard deviation

sd(wages)

# removing null values using omit command

a2=na.omit(a1)

# to check if there is any missing values in our newdata set m2

na.fail(a2)

#To print max values to 999999.

options(max.print=999999)