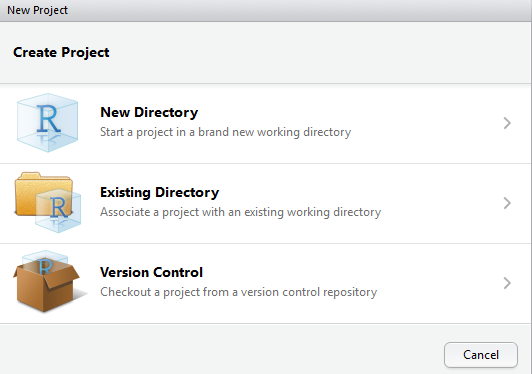
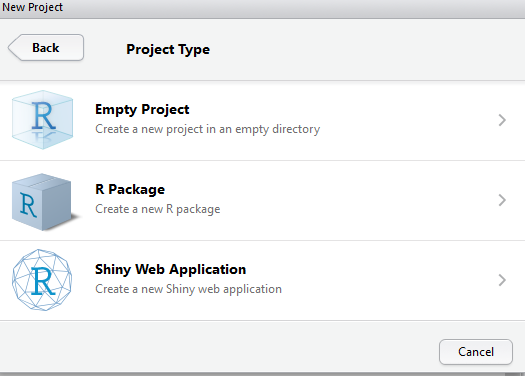
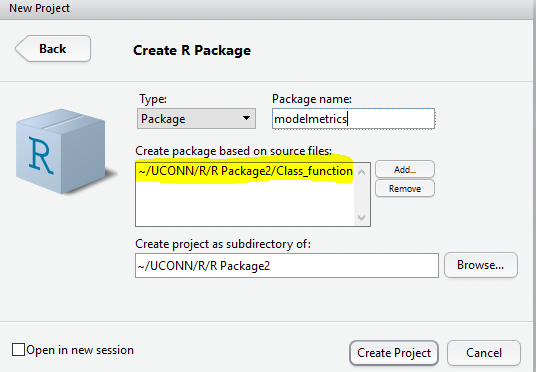
CREATING A PACKAGE IN R

**1. Create the Structure for the package:**

* Go to File-> New Project-> New Directory /Existing directory-> R Package

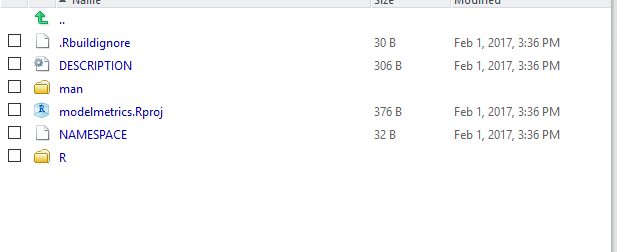






Give the R package name. Under Create package based on source files heading-> browse the R file that has the functions that we created during the class (Write the code for your functions in this .R file. These functions calculates the model metrics to evaluate the model performance)

Under the Create Project as subdirectory of heading: Give a location so that bare minimum folders are present. The structure of the package created looks as follows.



#Install the devtools package

install.packages(“devtools”,dependencies=TRUE)

#After Installing devtools package, load the library.

Library(devtools)

#Install the roxygen2 package

install.packages(“roxygen2”)

library(roxygen2)

**2. Add the description of the package**

* Then add the documentation to the description file (just click on the description) as follows where you mention title of the package, description, author, license etc. MIT is the generic license that we mention while creating the package.

-----------------------------------------------------The Code to put in the description file-------------------------------------------

Package: modelmetrics

Type: Package

Title: Model metrics calculation

Version: 0.1

Author: Pranitha Buddiga <pranitha.bsk3@gmail.com>

Maintainer: Pranitha Buddiga <pranitha.bsk3@gmail.com>

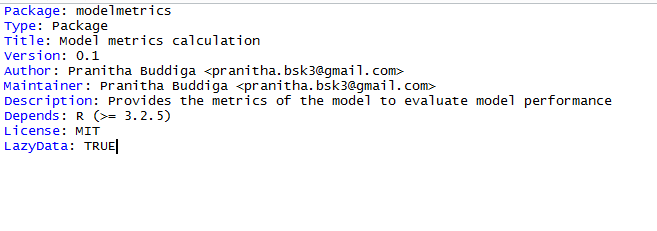
Description: Provides the metrics of the model to evaluate model performance

Depends: R (>= 3.2.5)

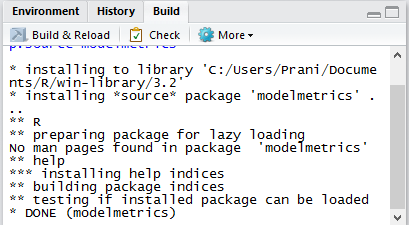
License: MIT

LazyData: TRUE

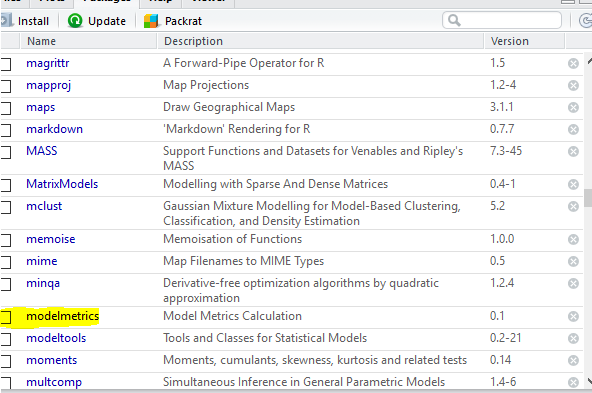
----------------------------------------------------------------------------------------------------------

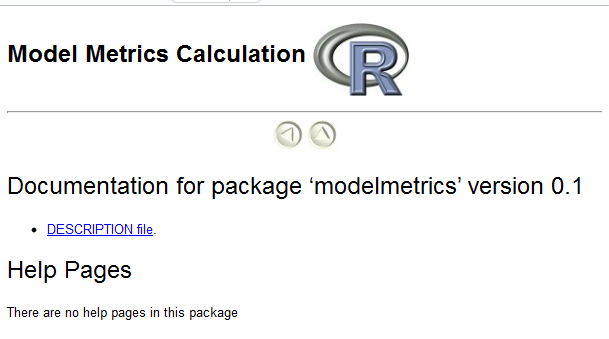


Then the documentation is built by clicking on build and reload option on the right. Once the documentation is successfully built, then we need to work on creating the help pages for the functions present in the library.



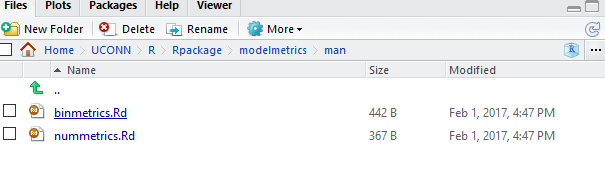
* The library is already loaded and when we click on it and see there are no help pages in the package



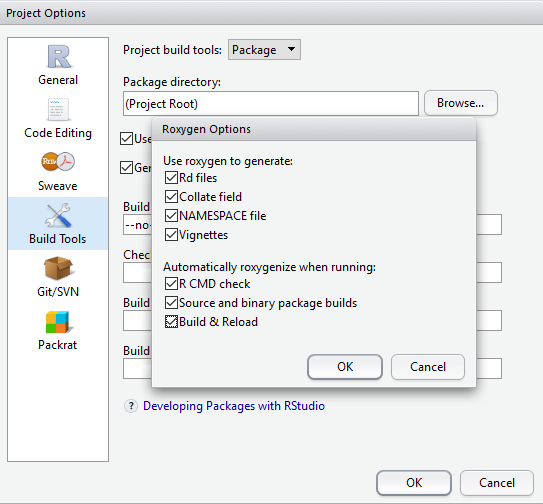


**3. Creating the help pages**

* Then go to the .R file and document it ( discussed later in the next steps) and then build and load it. On successful building it creates .rd files in the man folder



* We need to add the documentation that includes listing out the title of the function, arguments and then if there is a single function export it using @export in the R file. Then Build and Reload the R file.
* Before building and reloading the R file make sure to do this. Go to more options click on configure Build Tools and check the following options as shown in below pic



-----------------------------**The Code to create help pages for the functions**-------

#' finding model metrics

#'

#' This takes in the predicted and actual values.

#'

#' @name nummetrics

#' @param a,m are the actual and predicted values

#' @return vector of model metrics

#' @export

nummetrics = function(a,m)

{

metrics= c(MAD=0,MSE=0,MAPE=0,MPSE=0,tMAD=0,P90=0,R2=0)

metrics["MAD"] <- mean(abs(a-m))

metrics["MSE"] <- mean((a-m)^2)

metrics["MAPE"]<- mean(abs((a-m)/a))

metrics["MPSE"] <- mean(((a-m)/a)^2)

metrics["tMAD"] <- mean(abs(a-m),trim=0.05)

metrics["P90"]<- quantile(abs(a-m),probs =0.9)

SSE<-sum((a-m)^2)

SST<-sum((a-mean(a))^2)

metrics["R2"]<- 1-(SSE/SST)

return(metrics)

}

#' finding model metrics

#'

#' This takes in actual,predicted and no of predictors value

#'

#' @name binmetrics

#' @param a=actual m=model values k=10 sets a default value and can be changed

#' @return returns a vector of binary prediction model metrics

#' @export

binmetrics =function(a,m,k=10)

{

metrics= c(LL=0,AIC=0,BIC=0,R2=0)

metrics["LL"]= sum(ifelse(a==1,log(m),log(1-m)))

metrics["AIC"]= -2\*metrics["LL"]+2\*k

n=length(a)

metrics["BIC"]= -2\*metrics["LL"]+2\*k\*log(n)

SSE<-sum((a-m)^2)

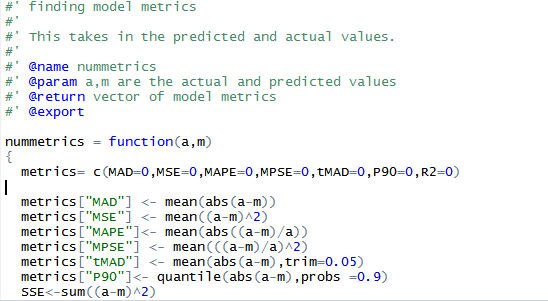
SST<-sum((a-mean(a))^2)

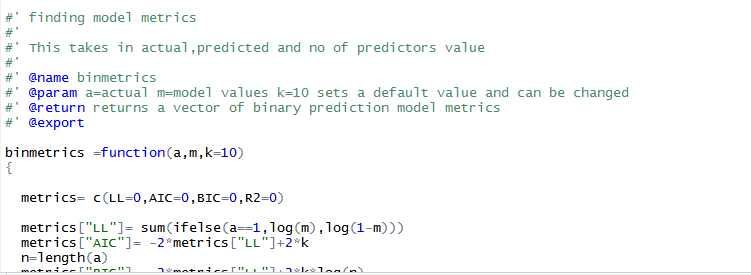
metrics["R2"]<- 1-(SSE/SST)

return(metrics)

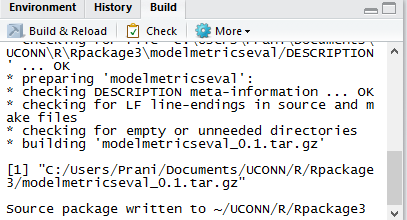
}

---------------------------------------------------------------------------------------------------------------------





* Once its built. Create the package clicking on “**More options**” to the right and then “**Build source package**”. On successful package creation, it will create a tar file. Modlemetric\_0.1.tar.gz file will be created and which can be shared across for others to install and use this package.

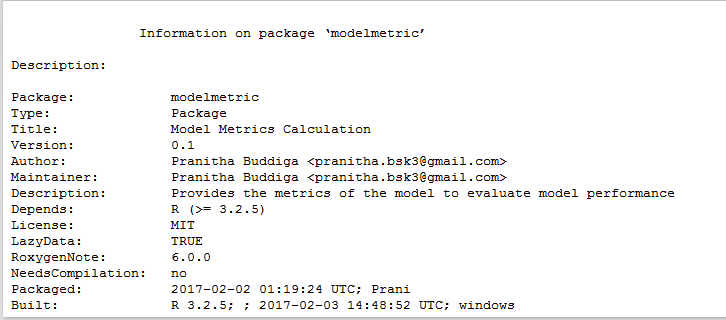


* To check the functionality of the package. Place the tar.gz file in the working directory and Run the following commands in order

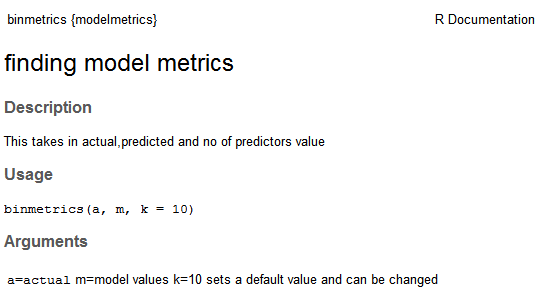
install.packages("modelmetric\_0.1.tar.gz",repos=NULL)

library(help=modelmetric)

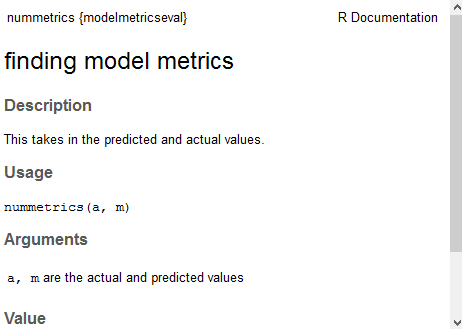
library(modelmetric)



?binmetrics – Gives the help page for the binmetrics function



?nummetrics



Follow the steps as is and create your own packages!