Configuration Manual

MSc Research Project
Data Analytics

Yogesh Maruti Patil
Student ID: x17169828

School of Computing
National College of Ireland

Supervisor: Dr. Catherine Mulwa

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Yogesh Maruti Patil
Student ID:	x17169828
Programme:	Data Analytics
Year:	2019
Module:	MSc Research Project
Supervisor:	Dr. Catherine Mulwa
Submission Due Date:	16/09/2019
Project Title:	Configuration Manual
Word Count:	671
Page Count:	20

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	15th September 2019

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Yogesh Maruti Patil
x17169828

1 System Configuration

The systems configuration plays an important role in order to start up with a new project. It is always suggested to look for the specifications first and download the required software accordingly. The configuration of the system used in this project is as shown in figure 1.

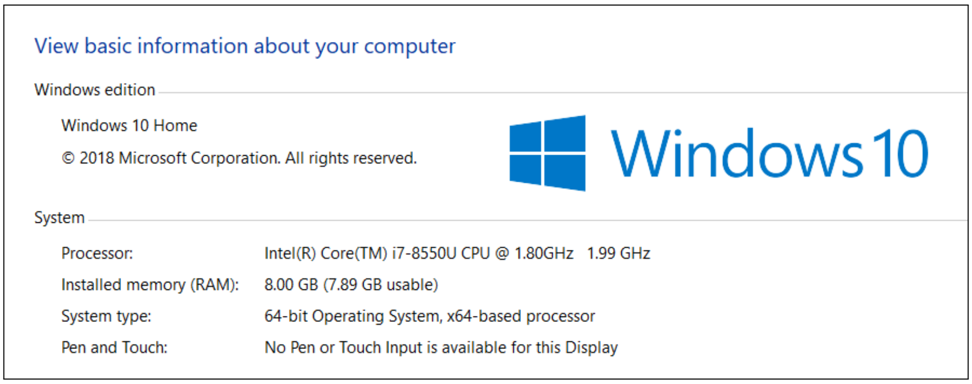


Figure 1: System Configuration

2 Software Installations

Some software installations were required to be done for this project. The softwares required for this project included R, R Studio, Tableau,and Microsoft Excel. The steps involved in their installation are being enlisted in this section.

2.1 Installation of R and R Studio

R is a software environment that comes with a Graphical User interface. R studio is an Integrated Development Environment. The steps involved in this installation are as follows:

1. The R installer for Windows can be downloded from the CRAN website [1] as shown in 2.The setup is quite easy, just press Next when required according to the instruction.

[1] https://cran.r-project.org/

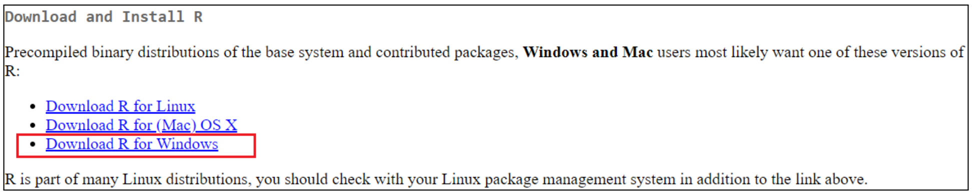


Figure 2: R installation from CRAN website

2. The second step involves installing the R studio. R studio installer can be downloaded from the RStudio website [2] as shown in 3. Download it according to system specification (e.g.32/64 bits). The setup for installation are quite easy, just press Next when required according to the instruction.

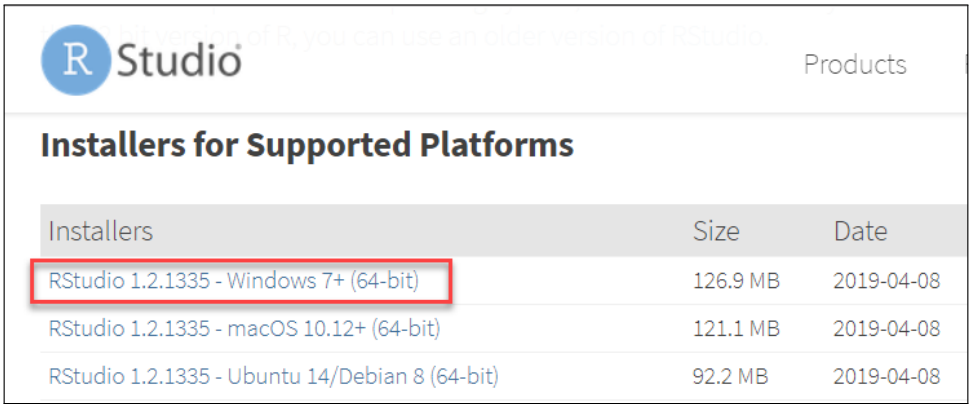


Figure 3: R Studio installation from RStudio website

3. Once the above two steps are done R Studio is ready to use. The initial window of R studio is as shown in 4. There after one can start coding.

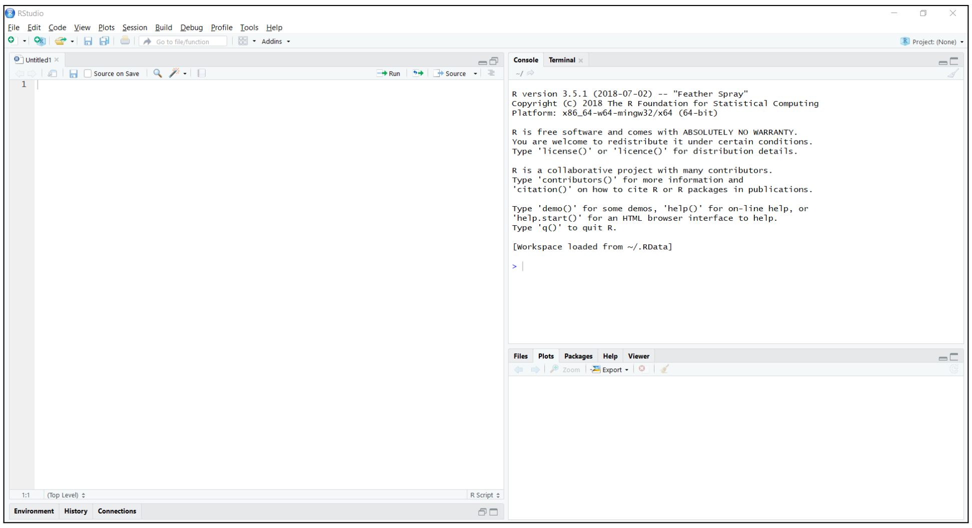


Figure 4: R Studio

[2]https://www.rstudio.com/products/rstudio/download/download

2.2 Installation of Tableau

Tableau is used for the visualization of the data. The steps involved in its installation are as follows:

1. The Tableau installer for Windows can be downloded from the Tableau website [3] as shown in 5. The setup is quite easy, just press Next when required according to the instruction.

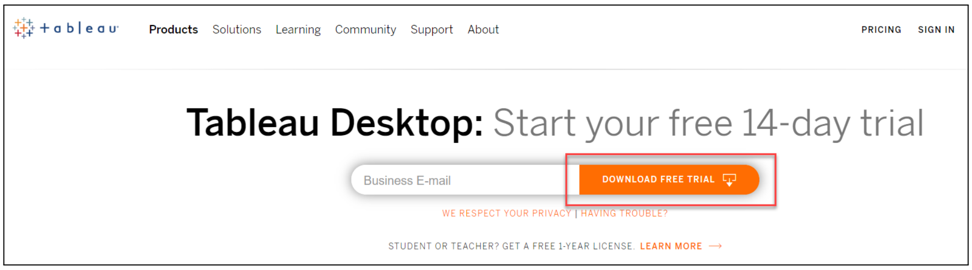


Figure 5: Tableau Website

2. Once the installation is done, one can connect any data file into Tableau and start working on charts.

3 Data Download

Data is downloaded from two sources for this project. The detailed information about them are as follows.

3.1 Solar Irradiance Data

For this project the solar data was downloaded from Solcast website [4]. Here free solar irradiance data is available for academic use. The steps involved in downloading the data are as follows:

1. Go to the Solcast website and register a new account as a student as shown in figure 6 and 7

[3]https://www.tableau.com/products/desktop/download?os=windows

[4]https://solcast.com.au/solar-radiation-data/

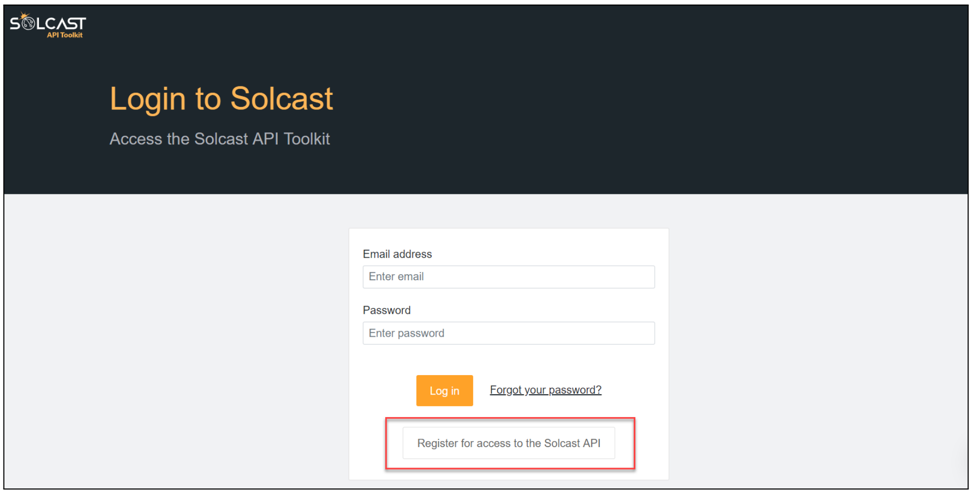


Figure 6: Solcast Login/Registration Page

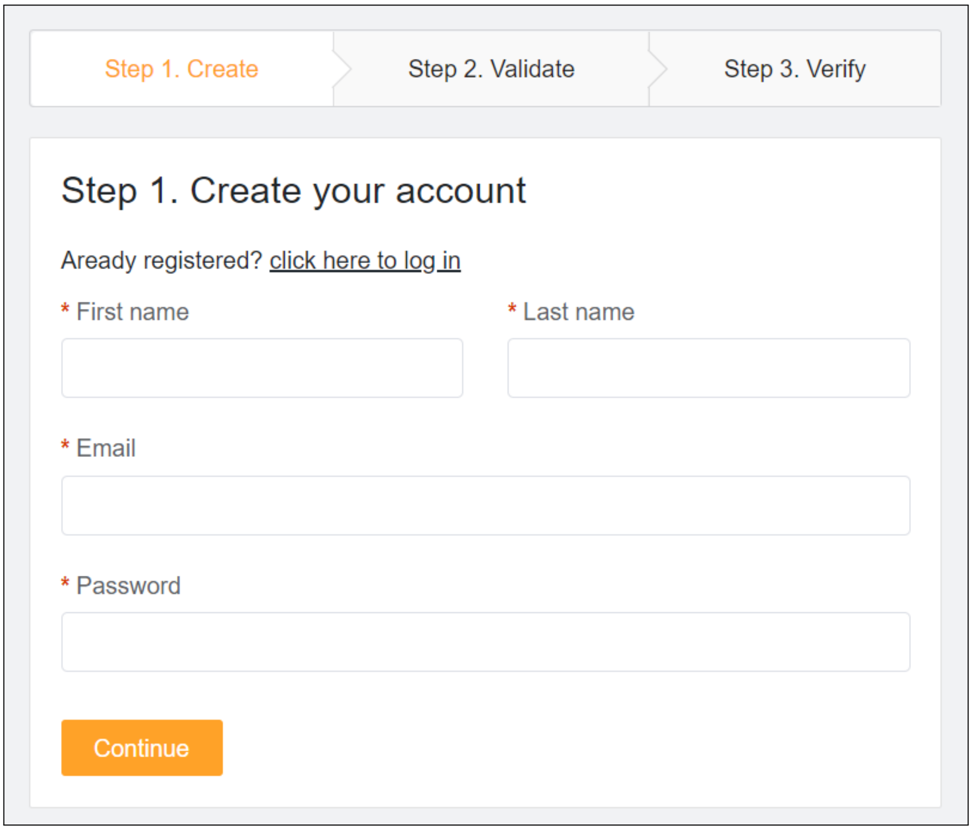


Figure 7: Solcast account creation page

2. Once the account is created, login into Solcast website. You will get this screen as shown in 8 with an initial free credit of $ 250. With each location data download for one year accounting for $ 50. For this project the data was for one location and for three years duration so free $150 were deducted after downloading.

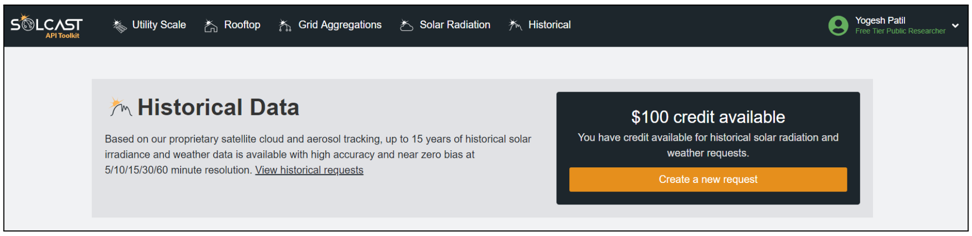


Figure 8: Solcast Free Credits for Students

3. Now to create a new historical data request follow the steps as in figures 9, 10, 11,

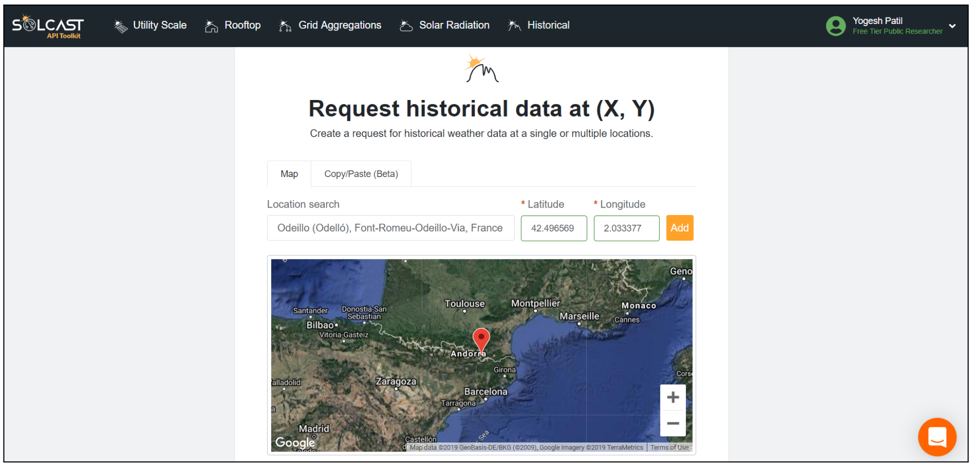


Figure 9: Historical Data Request

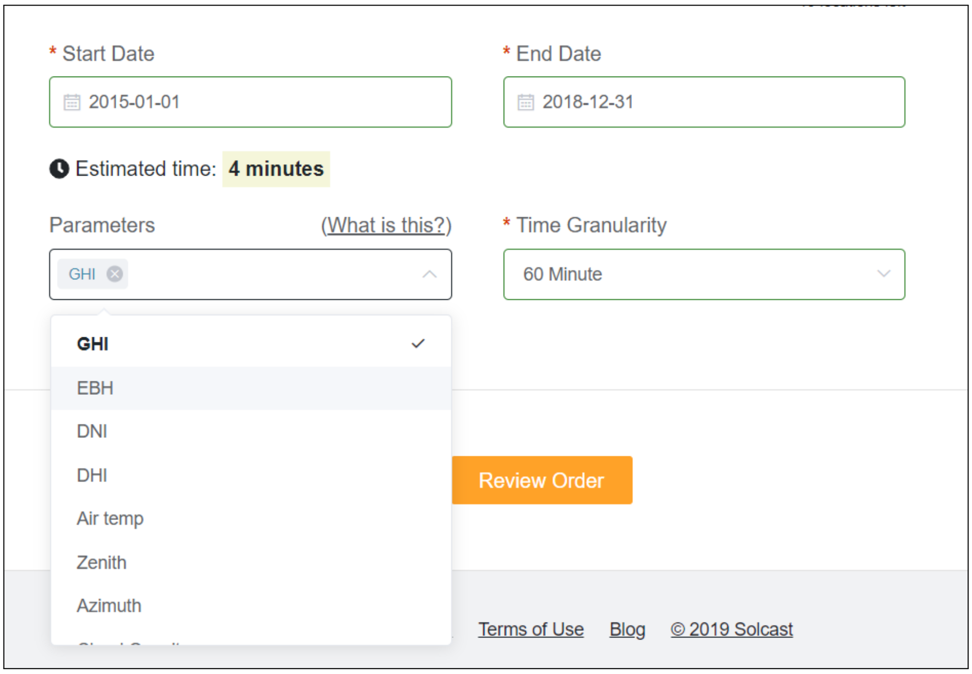


Figure 10: Solar irradiance data component selection

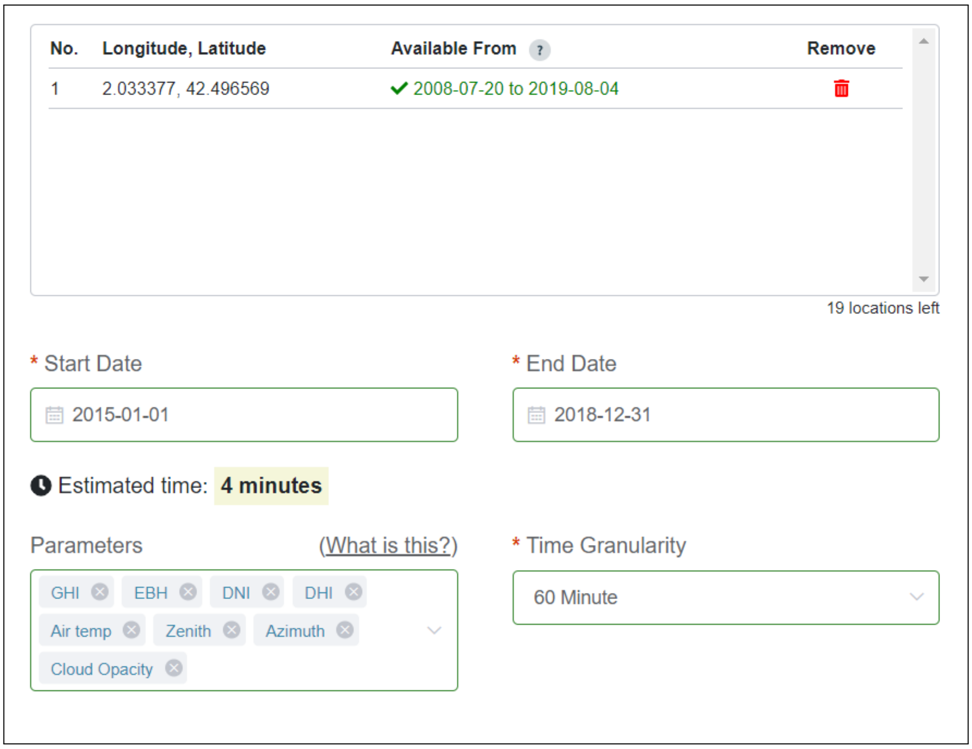


Figure 11: Solar irradiance data selected

4. Once the request is raised it can be checked in historical requests section as shown in figure 12. From there the data can be downloaded in csv format.

Figure 12: Recent historical Solar irradiance data downloaded

3.2 Meteorological Measures Data

For this project the meteorological measures data was downloaded from OpenWeatherMap website [5]. The steps involved in downloading the data are as follows:
1. Go to the OpenWeatherMap website and register a new account as a student as shown in figure 6 and 7

[5]https://openweathermap.org/

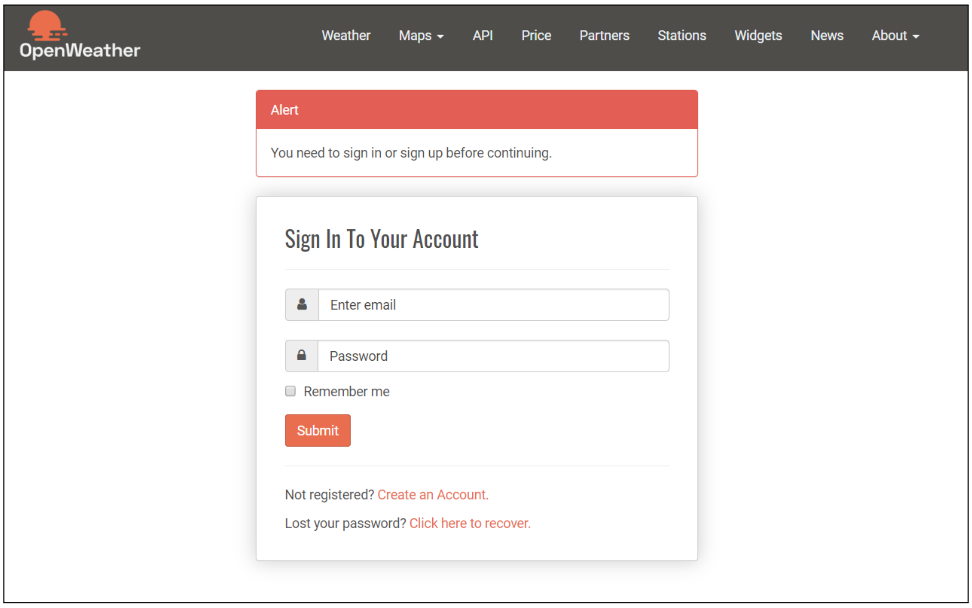


Figure 13: OpenWeatherMap Login/Registration Page

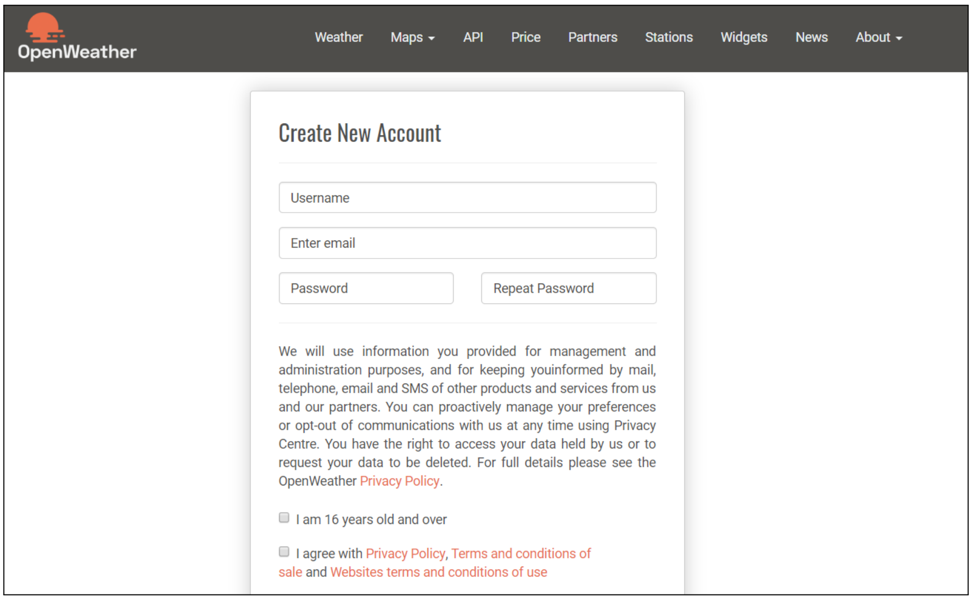


Figure 14: OpenWeatherMap account creation page

2. Once the account is created, login into OpenWeatherMap website. You will get this screen as shown in 15. From here you can create a new historical data request do as in figure 15.

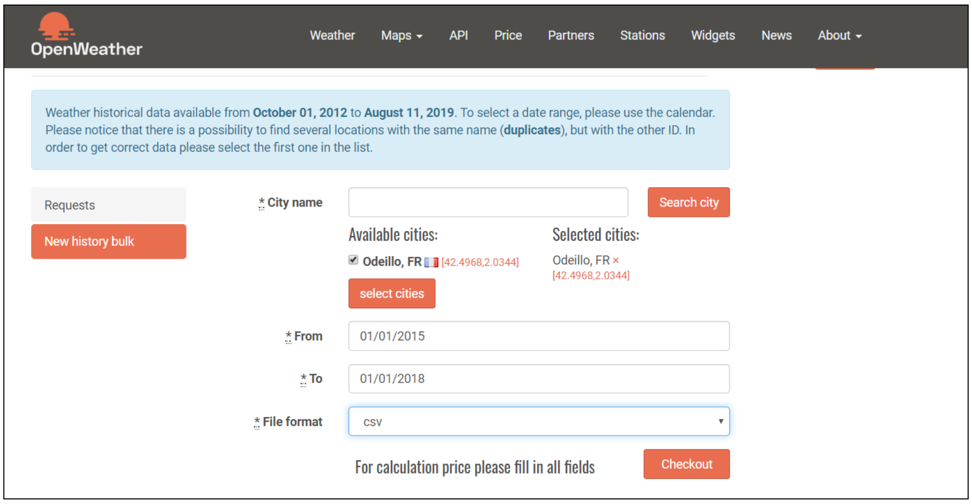
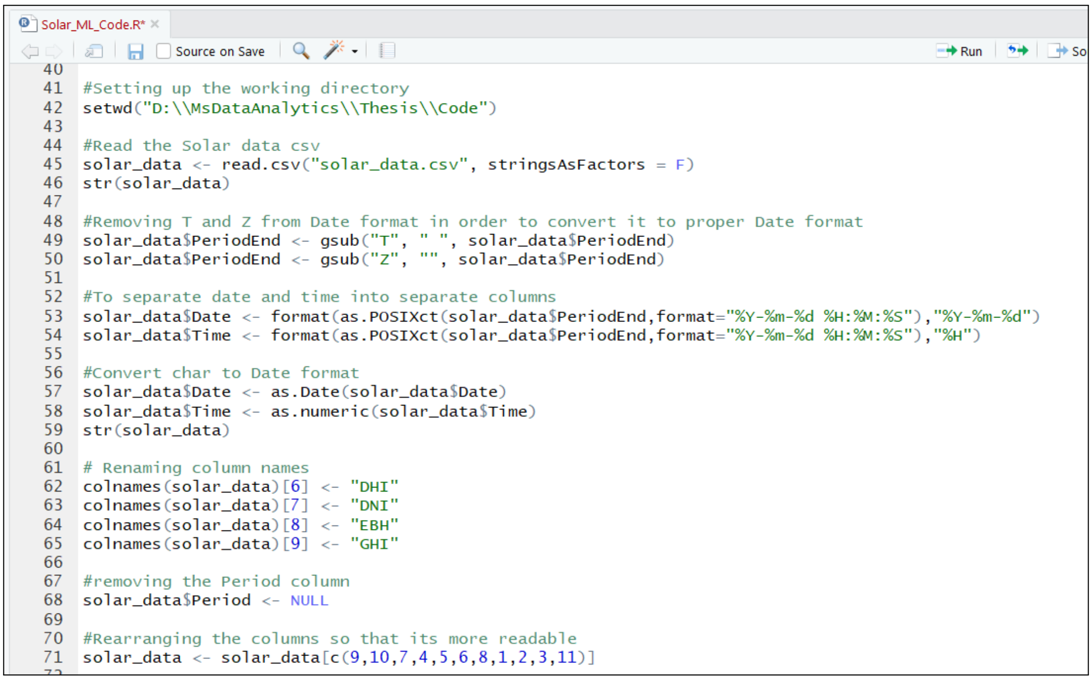


Figure 15: OpenWeatherMap Meteorological Data Download

4 Data Pre-processing

The data is pre-processed before putting into a model.



```
#Setting up the working directory
setwd("D:\\MsDataAnalytics\\Thesis\\Code")

#Read the Solar data csv
solar_data <- read.csv("solar_data.csv", stringsAsFactors = F)
str(solar_data)

#Removing T and Z from Date format in order to convert it to proper Date format
solar_data$PeriodEnd <- gsub("T", " ", solar_data$PeriodEnd)
solar_data$PeriodEnd <- gsub("Z", "", solar_data$PeriodEnd)

#To separate date and time into separate columns
solar_data$Date <- format(as.POSIXct(solar_data$PeriodEnd,format="%Y-%m-%d %H:%M:%S"),"%Y-%m-%d")
solar_data$Time <- format(as.POSIXct(solar_data$PeriodEnd,format="%Y-%m-%d %H:%M:%S"),"%H")

#Convert char to Date format
solar_data$Date <- as.Date(solar_data$Date)
solar_data$Time <- as.numeric(solar_data$Time)
str(solar_data)

# Renaming column names
colnames(solar_data)[6] <- "DHI"
colnames(solar_data)[7] <- "DNI"
colnames(solar_data)[8] <- "EBH"
colnames(solar_data)[9] <- "GHI"

#removing the Period column
solar_data$Period <- NULL

#Rearranging the columns so that its more readable
solar_data <- solar_data[c(9,10,7,4,5,6,8,1,2,3,11)]
```

Figure 16: Loading and processing of Solar Data

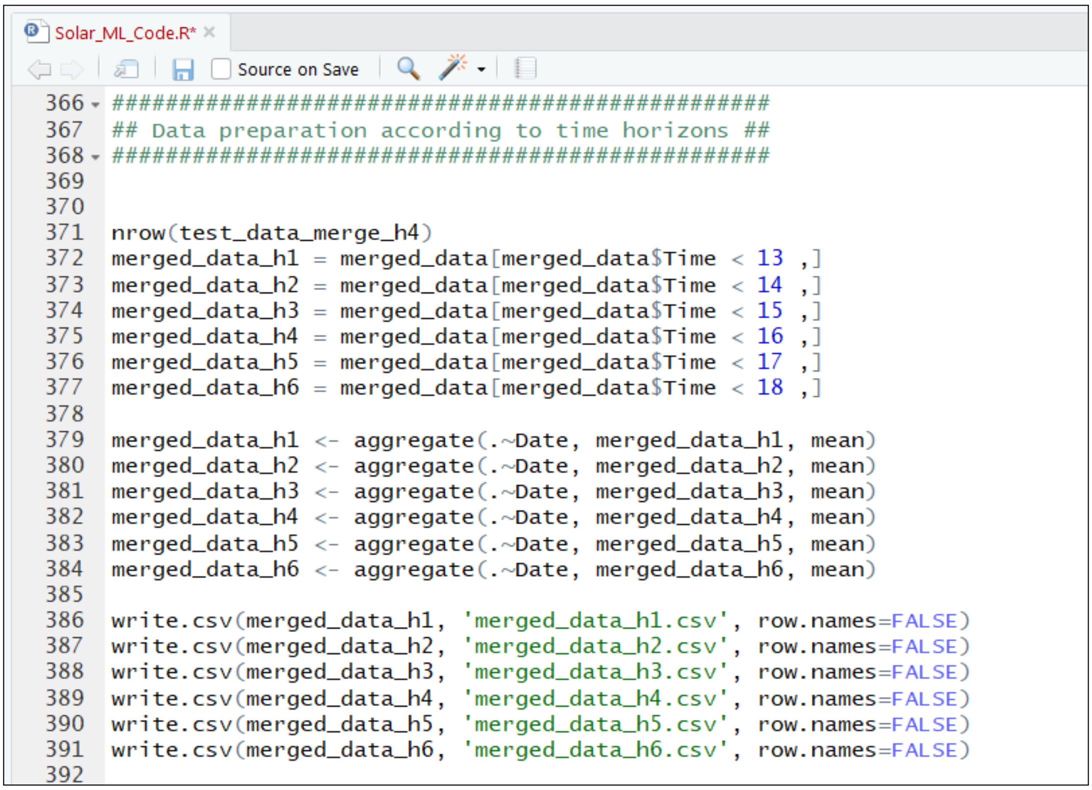
```r
#Loading the Meteorological measures data
meteorological_data <- read.csv("meteorological_data.csv", stringsAsFactors = F)
str(meteorological_data)

#Removing the unwanted columns
meteorological_data$dt <- NULL
meteorological_data$city_id <- NULL
meteorological_data$city_name <- NULL
meteorological_data$lat <- NULL
meteorological_data$lon <- NULL
meteorological_data$temp_max <- NULL
meteorological_data$temp_min <- NULL
meteorological_data$sea_level <- NULL
meteorological_data$grnd_level <- NULL
meteorological_data$wind_deg <- NULL
meteorological_data$rain_1h <- NULL
meteorological_data$rain_3h <- NULL
meteorological_data$rain_24h <- NULL
meteorological_data$rain_today <- NULL
meteorological_data$snow_1h <- NULL
meteorological_data$snow_3h <- NULL
meteorological_data$snow_24h <- NULL
meteorological_data$snow_today <- NULL
meteorological_data$weather_id <- NULL
meteorological_data$weather_icon <- NULL
meteorological_data$weather_description <- NULL

#Removing the unwanted characters from dt_iso
meteorological_data$dt_iso <- gsub(" \\+0000 UTC", "", meteorological_data$dt_iso)
```

Figure 17: Loading and processing of Meteorological Data

```r
##############################################
## Merging of solar and meteorological data ##
##############################################

merged_data <- merge(solar_data,meteorological_data,by.x = "PeriodEnd",by.y = "dt_iso")
str(merged_data)
#To separate date and time into separate columns
merged_data$Date <- format(as.POSIXct(merged_data$PeriodEnd,format="%Y-%m-%d %H:%M:%S"),"%Y-%m-%d")
merged_data$Time <- format(as.POSIXct(merged_data$PeriodEnd,format="%Y-%m-%d %H:%M:%S"),"%H")
#Convert char to Date format
merged_data$Date <- as.Date(merged_data$Date)
merged_data$Time <- as.numeric(merged_data$Time)

merged_data$PeriodEnd <- NULL
merged_data$weather_main <- NULL
merged_data$AirTemp <- NULL
merged_data <- merged_data[c(2,9,14,7,13,6,3,4,5,8,10,11,12,1)]
#Taking data before 2018-01-01
merged_data <- merged_data[merged_data$Date < "2018-01-01",]
#Taking data after 11 am and before 6pm
merged_data = merged_data[merged_data$Time > 11 ,]
merged_data = merged_data[merged_data$Time < 18 ,]
#remove NA if any
merged_data <- na.omit(merged_data)
#Rename the columns properly
colnames(merged_data)[3] <- "Clouds"
colnames(merged_data)[5] <- "Wind.Speed"
colnames(merged_data)[8] <- "Cloud.Opacity"
colnames(merged_data)[11] <- "Temperature"
colnames(merged_data)[12] <- "Pressure"
colnames(merged_data)[13] <- "Humidity"
str(merged_data)
#write data to csv
write.csv(merged_data, 'MergedSolarData.csv', row.names=FALSE)
```

Figure 18: Merging of Solar and Meteorological Data

5 Data Preparation

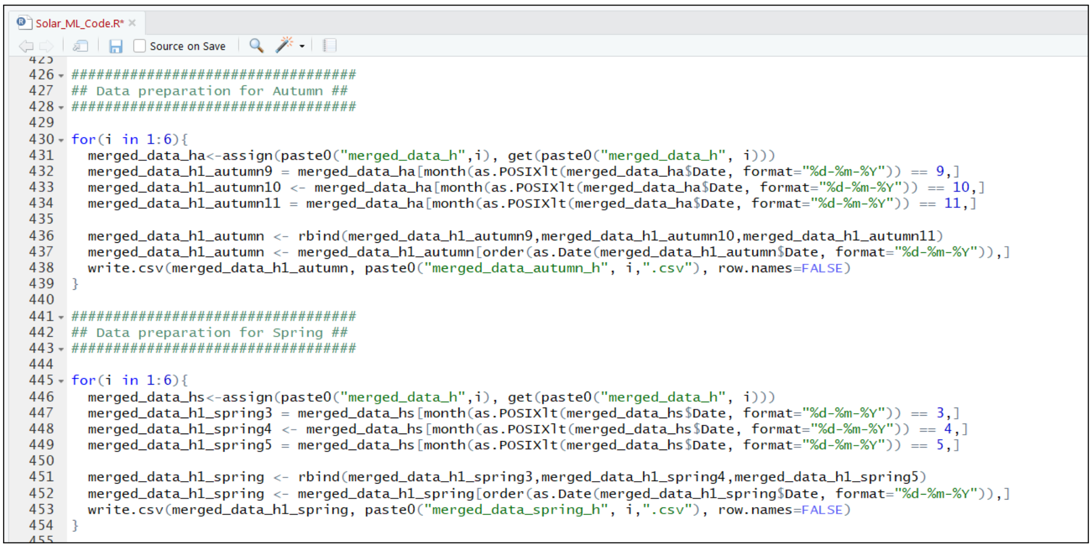


Solar_ML_Code.R*

Source on Save

```r
################################################
## Data preparation according to time horizons ##
################################################


nrow(test_data_merge_h4)
merged_data_h1 = merged_data[merged_data$Time < 13 ,]
merged_data_h2 = merged_data[merged_data$Time < 14 ,]
merged_data_h3 = merged_data[merged_data$Time < 15 ,]
merged_data_h4 = merged_data[merged_data$Time < 16 ,]
merged_data_h5 = merged_data[merged_data$Time < 17 ,]
merged_data_h6 = merged_data[merged_data$Time < 18 ,]

merged_data_h1 <- aggregate(.~Date, merged_data_h1, mean)
merged_data_h2 <- aggregate(.~Date, merged_data_h2, mean)
merged_data_h3 <- aggregate(.~Date, merged_data_h3, mean)
merged_data_h4 <- aggregate(.~Date, merged_data_h4, mean)
merged_data_h5 <- aggregate(.~Date, merged_data_h5, mean)
merged_data_h6 <- aggregate(.~Date, merged_data_h6, mean)

write.csv(merged_data_h1, 'merged_data_h1.csv', row.names=FALSE)
write.csv(merged_data_h2, 'merged_data_h2.csv', row.names=FALSE)
write.csv(merged_data_h3, 'merged_data_h3.csv', row.names=FALSE)
write.csv(merged_data_h4, 'merged_data_h4.csv', row.names=FALSE)
write.csv(merged_data_h5, 'merged_data_h5.csv', row.names=FALSE)
write.csv(merged_data_h6, 'merged_data_h6.csv', row.names=FALSE)

```

Figure 19: Code for data preparation for time horizon(h+1 to h+6)



Solar_ML_Code.R*

Source on Save

```r
##################################
## Data preparation for Autumn ##
##################################

for(i in 1:6){
  merged_data_ha<-assign(paste0("merged_data_h",i), get(paste0("merged_data_h", i)))
  merged_data_h1_autumn9 = merged_data_ha[month(as.POSIXlt(merged_data_ha$Date, format="%d-%m-%Y")) == 9,]
  merged_data_h1_autumn10 <- merged_data_ha[month(as.POSIXlt(merged_data_ha$Date, format="%d-%m-%Y")) == 10,]
  merged_data_h1_autumn11 = merged_data_ha[month(as.POSIXlt(merged_data_ha$Date, format="%d-%m-%Y")) == 11,]

  merged_data_h1_autumn <- rbind(merged_data_h1_autumn9,merged_data_h1_autumn10,merged_data_h1_autumn11)
  merged_data_h1_autumn <- merged_data_h1_autumn[order(as.Date(merged_data_h1_autumn$Date, format="%d-%m-%Y")),]
  write.csv(merged_data_h1_autumn, paste0("merged_data_autumn_h", i,".csv"), row.names=FALSE)
}

##################################
## Data preparation for Spring ##
##################################

for(i in 1:6){
  merged_data_hs<-assign(paste0("merged_data_h",i), get(paste0("merged_data_h", i)))
  merged_data_h1_spring3 = merged_data_hs[month(as.POSIXlt(merged_data_hs$Date, format="%d-%m-%Y")) == 3,]
  merged_data_h1_spring4 <- merged_data_hs[month(as.POSIXlt(merged_data_hs$Date, format="%d-%m-%Y")) == 4,]
  merged_data_h1_spring5 = merged_data_hs[month(as.POSIXlt(merged_data_hs$Date, format="%d-%m-%Y")) == 5,]

  merged_data_h1_spring <- rbind(merged_data_h1_spring3,merged_data_h1_spring4,merged_data_h1_spring5)
  merged_data_h1_spring <- merged_data_h1_spring[order(as.Date(merged_data_h1_spring$Date, format="%d-%m-%Y")),]
  write.csv(merged_data_h1_spring, paste0("merged_data_spring_h", i,".csv"), row.names=FALSE)
}
```

Figure 20: Code for data preparation for Autumn and Springs

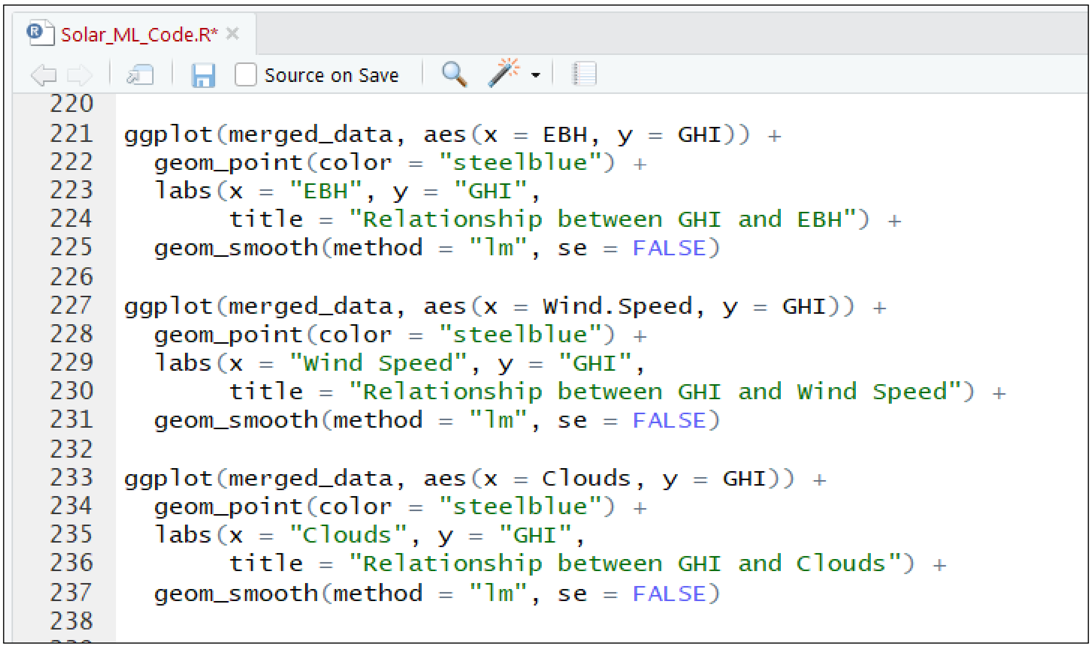
```r
##################################
## Data preparation for Winters ##
##################################


for(i in 1:6){
  merged_data_hw<-assign(paste0("merged_data_h",i), get(paste0("merged_data_h", i)))
  merged_data_h1_winters12 = merged_data_hw[month(as.POSIXlt(merged_data_hw$Date, format="%d-%m-%Y")) == 12,]
  merged_data_h1_winters1 <- merged_data_hw[month(as.POSIXlt(merged_data_hw$Date, format="%d-%m-%Y")) == 1,]
  merged_data_h1_winters2 = merged_data_hw[month(as.POSIXlt(merged_data_hw$Date, format="%d-%m-%Y")) == 2,]

  merged_data_h1_winters <- rbind(merged_data_h1_winters1,merged_data_h1_winters2,merged_data_h1_winters12)
  merged_data_h1_winters <- merged_data_h1_winters[order(as.Date(merged_data_h1_winters$Date, format="%d-%m-%Y")),]
  write.csv(merged_data_h1_winters, paste0("merged_data_winters_h", i,".csv"), row.names=FALSE)
}

##################################
## Data preparation for Summers ##
##################################

for(i in 1:6){
  merged_data_hs<-assign(paste0("merged_data_h",i), get(paste0("merged_data_h", i)))
  merged_data_h1_summers6 = merged_data_hs[month(as.POSIXlt(merged_data_hs$Date, format="%d-%m-%Y")) == 6,]
  merged_data_h1_summers7 <- merged_data_hs[month(as.POSIXlt(merged_data_hs$Date, format="%d-%m-%Y")) == 7,]
  merged_data_h1_summers8 = merged_data_hs[month(as.POSIXlt(merged_data_hs$Date, format="%d-%m-%Y")) == 8,]

  merged_data_h1_summers <- rbind(merged_data_h1_summers6,merged_data_h1_summers7,merged_data_h1_summers8)
  merged_data_h1_summers <- merged_data_h1_summers[order(as.Date(merged_data_h1_summers$Date, format="%d-%m-%Y")),]
  write.csv(merged_data_h1_summers, paste0("merged_data_summers_h", i,".csv"), row.names=FALSE)
}
```

Figure 21: Code for data preparation for Summers and Winters

6 Exploratory Data Analysis

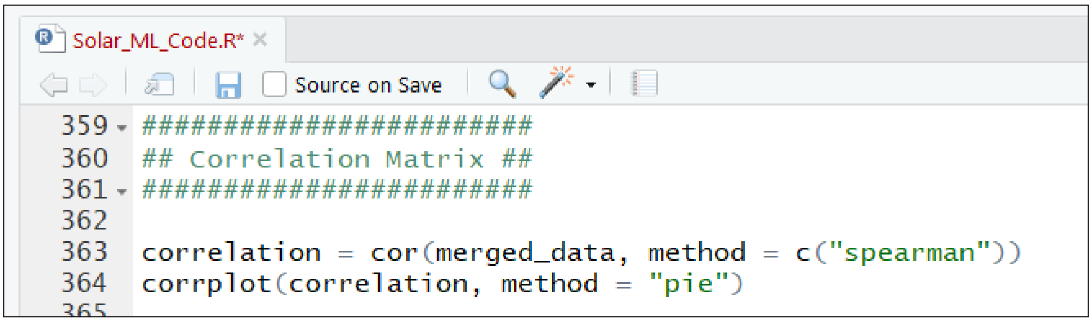
```r
##############################################################
########## Relationship between GHI and Other Predictors #####
##############################################################
ggplot(merged_data, aes(x = Humidity, y = GHI)) +
  geom_point(color = "steelblue") +
  labs(x = "Humidity", y = "GHI",
       title = "Relationship between GHI and Humidity") +
  geom_smooth(method = "lm", se = FALSE)

ggplot(merged_data, aes(x = Zenith, y = GHI)) +
  geom_point(color = "steelblue") +
  labs(x = "Zenith", y = "GHI",
       title = "Relationship between GHI and Zenith") +
  geom_smooth(method = "lm", se = FALSE)

ggplot(merged_data, aes(x = Pressure, y = GHI)) +
  geom_point(color = "steelblue") +
  labs(x = "Pressure", y = "GHI",
       title = "Relationship between GHI and Pressure") +
  geom_smooth(method = "lm", se = FALSE)

ggplot(merged_data, aes(x = Temperature, y = GHI)) +
  geom_point(color = "steelblue") +
  labs(x = "Temperature", y = "GHI",
       title = "Relationship between GHI and Temperature") +
  geom_smooth(method = "lm", se = FALSE)

ggplot(merged_data, aes(x = Azimuth, y = GHI)) +
  geom_point(color = "steelblue") +
  labs(x = "Azimuth", y = "GHI",
       title = "Relationship between GHI and Azimuth") +
  geom_smooth(method = "lm", se = FALSE)

ggplot(merged_data, aes(x = Cloud.Opacity, y = GHI)) +
  geom_point(color = "steelblue") +
  labs(x = "Cloud Opacity", y = "GHI",
       title = "Relationship between GHI and Cloud Opacity") +
  geom_smooth(method = "lm", se = FALSE)
```

Figure 22: Code for Relationship between Dependent and Other Predictors (Part I)



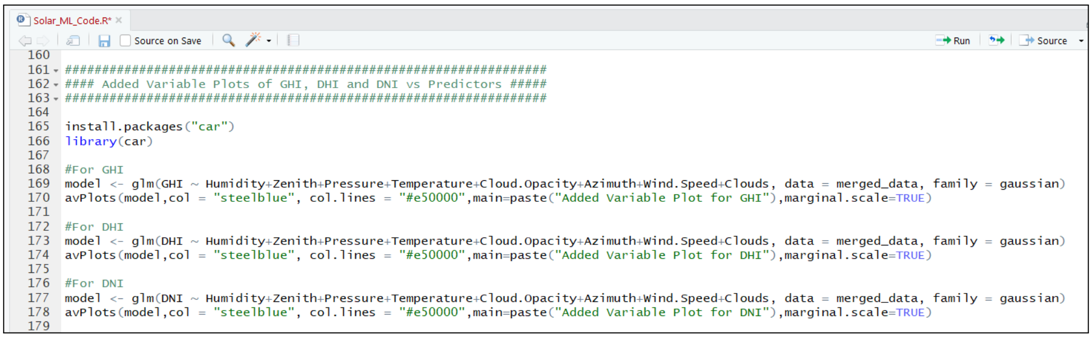
```r
ggplot(merged_data, aes(x = EBH, y = GHI)) +
  geom_point(color = "steelblue") +
  labs(x = "EBH", y = "GHI",
       title = "Relationship between GHI and EBH") +
  geom_smooth(method = "lm", se = FALSE)

ggplot(merged_data, aes(x = Wind.Speed, y = GHI)) +
  geom_point(color = "steelblue") +
  labs(x = "Wind Speed", y = "GHI",
       title = "Relationship between GHI and Wind Speed") +
  geom_smooth(method = "lm", se = FALSE)

ggplot(merged_data, aes(x = Clouds, y = GHI)) +
  geom_point(color = "steelblue") +
  labs(x = "Clouds", y = "GHI",
       title = "Relationship between GHI and Clouds") +
  geom_smooth(method = "lm", se = FALSE)
```

Figure 23: Code for Relationship between Dependent and Other Predictors (Part II)



```r
########################
## Correlation Matrix ##
########################

correlation = cor(merged_data, method = c("spearman"))
corrplot(correlation, method = "pie")
```

Figure 24: Code for Correlation Matrix



```r
#################################################################
#### Added Variable Plots of GHI, DHI and DNI vs Predictors #####
#################################################################

install.packages("car")
library(car)

#For GHI
model <- glm(GHI ~ Humidity+Zenith+Pressure+Temperature+Cloud.Opacity+Azimuth+Wind.Speed+Clouds, data = merged_data, family = gaussian)
avPlots(model,col = "steelblue", col.lines = "#e50000",main=paste("Added Variable Plot for GHI"),marginal.scale=TRUE)

#For DHI
model <- glm(DHI ~ Humidity+Zenith+Pressure+Temperature+Cloud.Opacity+Azimuth+Wind.Speed+Clouds, data = merged_data, family = gaussian)
avPlots(model,col = "steelblue", col.lines = "#e50000",main=paste("Added Variable Plot for DHI"),marginal.scale=TRUE)

#For DNI
model <- glm(DNI ~ Humidity+Zenith+Pressure+Temperature+Cloud.Opacity+Azimuth+Wind.Speed+Clouds, data = merged_data, family = gaussian)
avPlots(model,col = "steelblue", col.lines = "#e50000",main=paste("Added Variable Plot for DNI"),marginal.scale=TRUE)
```

Figure 25: Code for Added Variable plot

7 Feature Selection

```r
###############################################
# Feature Importance based on Random Forest ##
###############################################

#############
###For GHI###
#############

merged_data_randomF = randomForest(GHI~ Humidity+Zenith+Pressure+Temperature+Cloud.Opacity+Azimuth+Wind.Speed+Clouds, data=merged_data)
# Create an importance based on mean decreasing gini
imp <- varImpPlot(merged_data_randomF)
imp <- as.data.frame(imp)
imp$varnames <- rownames(imp)
rownames(imp) <- NULL

ggplot(imp, aes(x=reorder(varnames, IncNodePurity), y=IncNodePurity, color=as.factor(varnames))) +
  geom_point(size=3) +
  geom_segment(aes(x=varnames,xend=varnames,y=0,yend=IncNodePurity)) +
  ylab("IncNodePurity") + theme(axis.text = black.bold.8.text) +
  xlab("Variable Name") + theme(axis.text = black.bold.8.text) +
  coord_flip()

#############
###For DHI###
#############
merged_data_randomF = randomForest(DHI~ Humidity+Zenith+Pressure+Temperature+Cloud.Opacity+Azimuth+Wind.Speed+Clouds, data=merged_data)
# Create an importance based on mean decreasing gini
imp <- varImpPlot(merged_data_randomF)

imp <- as.data.frame(imp)
imp$varnames <- rownames(imp)
rownames(imp) <- NULL

ggplot(imp, aes(x=reorder(varnames, IncNodePurity), y=IncNodePurity, color=as.factor(varnames))) +
  geom_point(size=3) +
  geom_segment(aes(x=varnames,xend=varnames,y=0,yend=IncNodePurity)) +
  ylab("IncNodePurity") + theme(axis.text = black.bold.8.text) +
  xlab("Variable Name") + theme(axis.text = black.bold.8.text) +
  coord_flip()
```

Figure 26: Code for feature selection using random forest (Part I)

```r
#############
###For DNI###
#############

merged_data_randomF = randomForest(DNI~ Humidity+Zenith+Pressure+Temperature+Cloud.Opacity+Azimuth+Wind.Speed+Clouds, data=merged_data)
# Create an importance based on mean decreasing gini
imp <- varImpPlot(merged_data_randomF)

imp <- as.data.frame(imp)
imp$varnames <- rownames(imp)
rownames(imp) <- NULL

ggplot(imp, aes(x=reorder(varnames, IncNodePurity), y=IncNodePurity, color=as.factor(varnames))) +
  geom_point(size=3) +
  geom_segment(aes(x=varnames,xend=varnames,y=0,yend=IncNodePurity)) +
  ylab("IncNodePurity") + theme(axis.text = black.bold.8.text) +
  xlab("Variable Name") + theme(axis.text = black.bold.8.text) +
  coord_flip()
```

Figure 27: Code for feature selection using random forest (Part II)

```r
##############################################################
########## Feature selection using Boruta Algorithm ##########
##############################################################
summary(merged_data)
set.seed(111)

#For GHI
merged_data_train_GHI <- Boruta(GHI~ Humidity+Zenith+Pressure+Temperature+Cloud.Opacity+Azimuth+Wind.Speed+Clouds, data = merged_data, doTrace = 2)
print(merged_data_train_GHI)
merged_boruta_GHI <- names(merged_data_train_GHI$finalDecision[merged_data_train_GHI$finalDecision %in% c("Confirmed", "Tentative")])
print(merged_boruta_GHI)
plot(merged_data_train_GHI, cex.axis=.7, las=2, xlab="", main="Variable Importance for GHI")
merged_data_train_GHI_df <- attStats(merged_data_train_GHI)
print(merged_data_train_GHI_df)
merged_data_train_GHI_df <- merged_data_train_GHI_df[c(-3,-4,-5)]
merged_data_train_GHI_df$minImp <- NULL
merged_data_train_GHI_df$maxImp <- NULL
merged_data_train_GHI_df$normHits <- NULL
summary(merged_data_train_GHI_df)
write.csv(merged_data_train_GHI_df, 'merged_data_train_GHI_df.csv')

#For DHI
merged_data_train_DHI <- Boruta(DHI~ Humidity+Zenith+Pressure+Temperature+Cloud.Opacity+Azimuth+Wind.Speed+Clouds, data = merged_data, doTrace = 2)
print(merged_data_train_DHI)
merged_boruta_DHI <- names(merged_data_train_DHI$finalDecision[merged_data_train_DHI$finalDecision %in% c("Confirmed", "Tentative")])
print(merged_boruta_DHI)
plot(merged_data_train_DHI, cex.axis=.7, las=2, xlab="", main="Variable Importance for DHI")
merged_data_train_DHI_df <- attStats(merged_data_train_DHI)
merged_data_train_DHI_df <- merged_data_train_DHI_df[c(-3,-4,-5)]
write.csv(merged_data_train_DHI_df, 'merged_data_train_DHI_df.csv')

#For DNI
merged_data_train_DNI <- Boruta(DNI~ Humidity+Zenith+Pressure+Temperature+Cloud.Opacity+Azimuth+Wind.Speed+Clouds, data = merged_data, doTrace = 2)
print(merged_data_train_DNI)
merged_boruta_DNI <- names(merged_data_train_DNI$finalDecision[merged_data_train_DNI$finalDecision %in% c("Confirmed", "Tentative")])
print(merged_boruta_DNI)
ggplot(merged_data_train_DNI, cex.axis=.7, las=2, xlab="", main="Variable Importance for DNI")
merged_data_train_DNI_df <- attStats(merged_data_train_DNI)
merged_data_train_DNI_df <- merged_data_train_DNI_df[c(-3,-4,-5)]
write.csv(merged_data_train_DNI_df, 'merged_data_train_DNI_df.csv')
```

Figure 28: Code for feature selection using Boruta Algorithm

```r
##########################################
### VIF- For multicolinearity check######
##########################################

xx_reg <- lm(GHI~ Humidity+Zenith+Pressure+Temperature+Cloud.Opacity+Azimuth+Wind.Speed+Clouds, data=merged_data)
summary(xx_reg)
vif(xx_reg)
#if greater than 5 then it should be discarded

xx_reg <- lm(DHI~ Humidity+Zenith+Pressure+Temperature+Cloud.Opacity+Azimuth+Wind.Speed+Clouds, data=merged_data)
summary(xx_reg)
vif(xx_reg)

xx_reg <- lm(DNI~ Humidity+Zenith+Pressure+Temperature+Cloud.Opacity+Azimuth+Wind.Speed+Clouds, data=merged_data)
summary(xx_reg)
vif(xx_reg)
```

Figure 29: Code for feature selection using VIF

8 Implementation, Evaluation and Results

Solar_ML_Code.R*

Source on Save

```r
#################################################
######## Test and Train data Creation ###########
#################################################

merged_data_trial <- merged_data
summary(merged_data_trial)
merged_data_trial$GHI

merged_data_trial$GHI[merged_data_trial$GHI == 0] <- NA
merged_data_trial$GHI <- na_mean(merged_data_trial$GHI)
set.seed(1)

#Data split
div <- sample.split(merged_data_trial, SplitRatio = 0.8)
TrainData <- subset(merged_data_trial, div == TRUE)
TestData <- subset(merged_data_trial, div == FALSE)

dim(TestData)

```

Figure 30: Code for test and train data creation

Solar_ML_Code.R*

Source on Save

```r
## Performance Metrics Calculation Loop ####
#Creation of Data Frame with Table for Evaluation Metrics for all models and all horizons
perf_metric_dataframe <- data.frame(matrix(ncol = 8, nrow = 24),stringsAsFactors=FALSE)
x <- c("Metric", "Model", "h+1", "h+2", "h+3","h+4", "h+5", "h+6")
colnames(perf_metric_dataframe) <- x
#Creation of Function to put the results in the Data frame by time horizons
setresults <- function(model, rmse, mae, nrmse, nmae){
  if(sum(is.na(perf_metric_dataframe$`h+1`)) !=0 ){
    i <-  as.numeric(nrow(perf_metric_dataframe) - (sum(is.na(perf_metric_dataframe$`h+1`))-1))
    perf_metric_dataframe$Metric[i] <- "RMSE"
    perf_metric_dataframe$Model[i] <- model
    perf_metric_dataframe$Metric[i+1] <- "MAE"
    perf_metric_dataframe$Model[i+1] <- model
    perf_metric_dataframe$Metric[i+2] <- "nRMSE"
    perf_metric_dataframe$Model[i+2] <- model
    perf_metric_dataframe$Metric[i+3] <- "nMAE"
    perf_metric_dataframe$Model[i+3] <- model
    perf_metric_dataframe$`h+1`[i] <- rmse
    perf_metric_dataframe$`h+1`[i+1] <- mae
    perf_metric_dataframe$`h+1`[i+2] <- nrmse
    perf_metric_dataframe$`h+1`[i+3] <- nmae
  } else if(sum(is.na(perf_metric_dataframe$`h+2`)) !=0 ){
    i <-  as.numeric(nrow(perf_metric_dataframe) - (sum(is.na(perf_metric_dataframe$`h+2`))-1))
    perf_metric_dataframe$`h+2`[i] <- rmse
    perf_metric_dataframe$`h+2`[i+1] <- mae
    perf_metric_dataframe$`h+2`[i+2] <- nrmse
    perf_metric_dataframe$`h+2`[i+3] <- nmae
  } else if(sum(is.na(perf_metric_dataframe$`h+3`)) !=0){
    i <-  as.numeric(nrow(perf_metric_dataframe) - (sum(is.na(perf_metric_dataframe$`h+3`))-1))
    perf_metric_dataframe$`h+3`[i] <- rmse
    perf_metric_dataframe$`h+3`[i+1] <- mae
    perf_metric_dataframe$`h+3`[i+2] <- nrmse
    perf_metric_dataframe$`h+3`[i+3] <- nmae
  } else if(sum(is.na(perf_metric_dataframe$`h+4`)) !=0){
    i <-  as.numeric(nrow(perf_metric_dataframe) - (sum(is.na(perf_metric_dataframe$`h+4`))-1))
    perf_metric_dataframe$`h+4`[i] <- rmse
    perf_metric_dataframe$`h+4`[i+1] <- mae
    perf_metric_dataframe$`h+4`[i+2] <- nrmse
    perf_metric_dataframe$`h+4`[i+3] <- nmae
  } else if(sum(is.na(perf_metric_dataframe$`h+5`)) !=0){
    i <-  as.numeric(nrow(perf_metric_dataframe) - (sum(is.na(perf_metric_dataframe$`h+5`))-1))
    perf_metric_dataframe$`h+5`[i] <- rmse
```

Figure 31: Code for loading performance metrics in tabular form (Part I)

```r
        perf_metric_dataframe$`h+4`[i+3] <- nmae
    } else if(sum(is.na(perf_metric_dataframe$`h+5`)) !=0){
        i <-  as.numeric(nrow(perf_metric_dataframe) - (sum(is.na(perf_metric_dataframe$`h+5`))-1))
        perf_metric_dataframe$`h+5`[i] <- rmse
        perf_metric_dataframe$`h+5`[i+1] <- mae
        perf_metric_dataframe$`h+5`[i+2] <- nrmse
        perf_metric_dataframe$`h+5`[i+3] <- nmae
    } else if(sum(is.na(perf_metric_dataframe$`h+6`)) !=0){
        i <-  as.numeric(nrow(perf_metric_dataframe) - (sum(is.na(perf_metric_dataframe$`h+6`))-1))
        perf_metric_dataframe$`h+6`[i] <- rmse
        perf_metric_dataframe$`h+6`[i+1] <- mae
        perf_metric_dataframe$`h+6`[i+2] <- nrmse
        perf_metric_dataframe$`h+6`[i+3] <- nmae
    }
    print(perf_metric_dataframe)

  }

```

Figure 32: Code for loading performance metrics in tabular form (Part II)

```r
for (j in 1:6){

  ## For ANNUAL ##
  # merged_data_main<-assign(paste0("merged_data_h",j), get(paste0("merged_data_h", j)))
  # div <- sample.split(merged_data_main, SplitRatio = 0.8)
  # TrainData <- subset(merged_data_main, div == TRUE)
  # TestData <- subset(merged_data_main, div == FALSE)
  # dim(TestData)

  ## For Seasons ##
  seasons_main <- read.csv(paste0("merged_data_h",j,".csv"), stringsAsFactors = F)
  div <- sample.split(seasons_main, SplitRatio = 0.8)
  TrainData <- subset(seasons_main, div == TRUE)
  TestData <- subset(seasons_main, div == FALSE)

  # Run algorithms using 10-fold cross validation
  control <-  trainControl(method="cv", number=10)

  #########################################
  ############## CART ####################
  #########################################

  set.seed(7)
  CART_Train <- train(DNI~ Humidity+Zenith+Pressure+Temperature+Cloud.Opacity+Azimuth+Wind.Speed+Clouds, data=TrainData, method="rpart",trControl=control)
  predicted_val_cart <- predict(CART_Train ,TestData)

  #Evaluation Metrics
  cart_rmse<- hydroGOF::rmse(predicted_val_cart,TestData$DNI)
  cart_nrmse<- hydroGOF::nrmse(predicted_val_cart,TestData$DNI,norm = "sd")
  cart_mae<- hydroGOF::mae(predicted_val_cart,TestData$DNI)
  cart_nmae <- compute.nmae(predicted_val_cart,TestData$DNI)*100

  #Function call to append the results
  perf_metric_dataframe <- setresults("CART", cart_rmse, cart_mae, cart_nrmse, cart_nmae)

```

Figure 33: Code for Loading of Data and CART model implementation

```r
  ####################################################
  ############## LINEAR REGRESSION ####################
  ####################################################
  summary(TrainData)

  LR_Train <- train(DNI~ Humidity+Zenith+Pressure+Temperature+Cloud.Opacity+Azimuth+Wind.Speed+Clouds, data=TrainData, method="lm",trControl=control)
  predicted_val_LR <- predict(LR_Train ,TestData)

  ## Evaluation Metrics

  LR_rmse<- hydroGOF::rmse(predicted_val_LR,TestData$DNI)
  LR_nrmse<- hydroGOF::nrmse(predicted_val_LR,TestData$DNI,norm = "sd")
  LR_mae<- hydroGOF::mae(predicted_val_LR,TestData$DNI)
  LR_nmae <- compute.nmae(predicted_val_LR,TestData$DNI)*100

  #Function call to append the results
  perf_metric_dataframe <- setresults("LR", LR_rmse, LR_mae, LR_nrmse, LR_nmae)


  ##############################################################
  ############## Stochastic Gradient Boosting ###################
  ##############################################################

  SGB_Train <- train(DNI~ Humidity+Zenith+Pressure+Temperature+Cloud.Opacity+Azimuth+Wind.Speed+Clouds, data=TrainData, method="gbm",trControl=control)
  predicted_val_SGB <- predict(SGB_Train ,TestData)

  ## Evaluation Metrics
  SGB_rmse<- hydroGOF::rmse(predicted_val_SGB,TestData$DNI)
  SGB_nrmse<- hydroGOF::nrmse(predicted_val_SGB,TestData$DNI,norm = "sd")
  SGB_mae<- hydroGOF::mae(predicted_val_SGB,TestData$DNI)
  SGB_nmae <- compute.nmae(predicted_val_SGB,TestData$DNI)*100

  #Function call to append the results
  perf_metric_dataframe <- setresults("SGB", SGB_rmse, SGB_mae, SGB_nrmse, SGB_nmae)

```

Figure 34: Code for implementation of LR and SGB model

```r
##########################################
############## KNN ######################
##########################################


tunegrid<- expand.grid(k=9)
set.seed(7)
KNN_Train <- train(DNI~ Humidity+Zenith+Pressure+Temperature+Cloud.Opacity+Azimuth+Wind.Speed+Clouds, data=TrainData, method="knn",trControl=control)
predicted_val_knn <- predict(KNN_Train ,TestData)

## Evaluation Metrics
KNN_rmse<- hydroGOF::rmse(predicted_val_knn,TestData$DNI)
KNN_nrmse<- hydroGOF::nrmse(predicted_val_knn,TestData$DNI,norm = "sd")
KNN_mae<- hydroGOF::mae(predicted_val_knn,TestData$DNI)
KNN_nmae <- compute.nmae(predicted_val_knn,TestData$DNI)*100

#Function call to append the results
perf_metric_dataframe <- setresults("KNN", KNN_rmse, KNN_mae, KNN_nrmse, KNN_nmae)


##########################################
############## SVM #####################
##########################################


set.seed(7)
SVM_Train <- train(DNI~ Humidity+Zenith+Pressure+Temperature+Cloud.Opacity+Azimuth+Wind.Speed+Clouds, data=TrainData, method="svmRadial",trControl=control)

predicted_val_svm <- predict(SVM_Train ,TestData)

#Evaluation Metrics
SVM_rmse<- hydroGOF::rmse(predicted_val_svm,TestData$DNI)
SVM_nrmse<- hydroGOF::nrmse(predicted_val_svm,TestData$DNI,norm = "sd")
SVM_mae<- hydroGOF::mae(predicted_val_svm,TestData$DNI)
SVM_nmae <- compute.nmae(predicted_val_svm,TestData$DNI)*100

#Function call to append the results
perf_metric_dataframe <- setresults("SVM", SVM_rmse, SVM_mae, SVM_nrmse, SVM_nmae)
```

Figure 35: Code for implementation of KNN and SVM model

```r
##########################################
############## RANDOM FOREST ###################
##########################################

set.seed(7)
tunegrid <- expand.grid(.mtry=7)

RF_Train <- train(GHI~GHI, data=TrainData, method="rf", tuneGrid= tunegrid,trControl=control,prox=TRUE,allowParallel=TRUE)
predicted_val_rf <- predict(RF_Train ,TestData)

#Evaluation Metrics
RF_rmse<- hydroGOF::rmse(predicted_val_rf,TestData$DNI)
RF_nrmse<- hydroGOF::nrmse(predicted_val_rf,TestData$DNI,norm = "sd")
RF_mae<- hydroGOF::mae(predicted_val_rf,TestData$DNI)
RF_nmae <- compute.nmae(predicted_val_rf,TestData$DNI)*100

#Function call to append the results
perf_metric_dataframe <- setresults("RF", RF_rmse, RF_mae, RF_nrmse, RF_nmae)



}
```

Figure 36: Code for implementation of RF model

```r
#For ANNUAL
Eval_metric_GHI_Annual <- perf_metric_dataframe[order(perf_metric_dataframe$Metric),]
write.csv(Eval_metric_GHI_Annual, 'Eval_metric_GHI_Annual.csv', row.names=FALSE)

Eval_metric_DHI_Annual <- perf_metric_dataframe[order(perf_metric_dataframe$Metric),]
write.csv(Eval_metric_DHI_Annual, 'Eval_metric_DHI_Annual.csv', row.names=FALSE)

Eval_metric_DNI_Annual <- perf_metric_dataframe[order(perf_metric_dataframe$Metric),]
write.csv(Eval_metric_DNI_Annual, 'Eval_metric_DNI_Annual.csv', row.names=FALSE)
```

Figure 37: Code for saving annual evaluation metrics file

```
#For SEASONS
#WINTERS
Eval_metric_GHI_Winters <- perf_metric_dataframe[order(perf_metric_dataframe$Metric),]
write.csv(Eval_metric_GHI_Winters, 'Eval_metric_GHI_Winters.csv', row.names=FALSE)

Eval_metric_DHI_Winters <- perf_metric_dataframe[order(perf_metric_dataframe$Metric),]
write.csv(Eval_metric_DHI_Winters, 'Eval_metric_DHI_Winters.csv', row.names=FALSE)

Eval_metric_DNI_Winters <- perf_metric_dataframe[order(perf_metric_dataframe$Metric),]
write.csv(Eval_metric_DNI_Winters, 'Eval_metric_DNI_Winters.csv', row.names=FALSE)

#For SUMMERS
Eval_metric_GHI_Summers <- perf_metric_dataframe[order(perf_metric_dataframe$Metric),]
write.csv(Eval_metric_GHI_Summers, 'Eval_metric_GHI_Summers.csv', row.names=FALSE)

Eval_metric_DHI_Summers <- perf_metric_dataframe[order(perf_metric_dataframe$Metric),]
write.csv(Eval_metric_DHI_Summers, 'Eval_metric_DHI_Summers.csv', row.names=FALSE)

Eval_metric_DNI_Summers <- perf_metric_dataframe[order(perf_metric_dataframe$Metric),]
write.csv(Eval_metric_DNI_Summers, 'Eval_metric_DNI_Summers.csv', row.names=FALSE)

#For Autums
Eval_metric_GHI_Autumn <- perf_metric_dataframe[order(perf_metric_dataframe$Metric),]
write.csv(Eval_metric_GHI_Autumn, 'Eval_metric_GHI_Autumn.csv', row.names=FALSE)

Eval_metric_DHI_Autumn <- perf_metric_dataframe[order(perf_metric_dataframe$Metric),]
write.csv(Eval_metric_DHI_Autumn, 'Eval_metric_DHI_Autumn.csv', row.names=FALSE)

Eval_metric_DNI_Autumn <- perf_metric_dataframe[order(perf_metric_dataframe$Metric),]
write.csv(Eval_metric_DNI_Autumn, 'Eval_metric_DNI_Autumn.csv', row.names=FALSE)

#For Springs
Eval_metric_GHI_Spring <- perf_metric_dataframe[order(perf_metric_dataframe$Metric),]
write.csv(Eval_metric_GHI_Spring, 'Eval_metric_GHI_Spring.csv', row.names=FALSE)

Eval_metric_DHI_Spring <- perf_metric_dataframe[order(perf_metric_dataframe$Metric),]
write.csv(Eval_metric_DHI_Spring, 'Eval_metric_DHI_Spring.csv', row.names=FALSE)

Eval_metric_DNI_Spring <- perf_metric_dataframe[order(perf_metric_dataframe$Metric),]
write.csv(Eval_metric_DNI_Spring, 'Eval_metric_DNI_Spring.csv', row.names=FALSE)
```

Figure 38: Code for saving seasonal evaluation metrics file

9 Visualization of Results in Tableau

The files with results generated in 38 are visualized in Tableau.

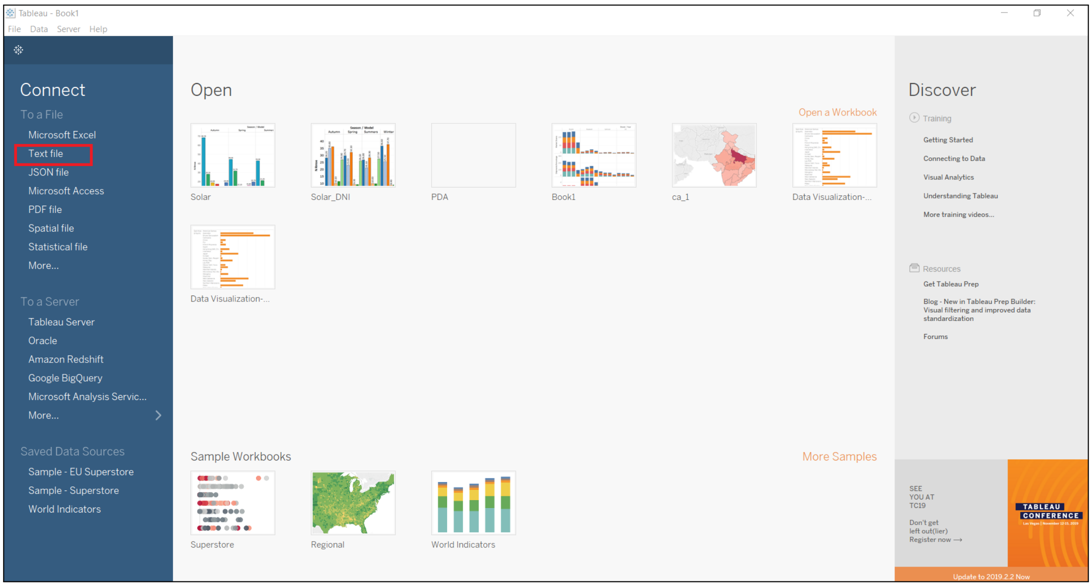


Figure 39: Uploading results csv into Tableau

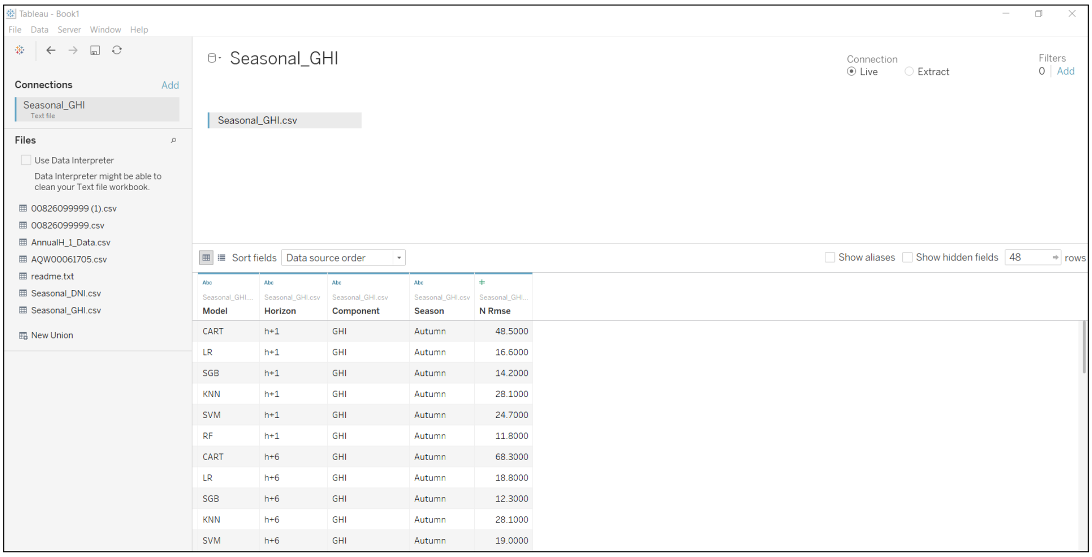


Figure 40: Screenshot after loading the result file in Tableau

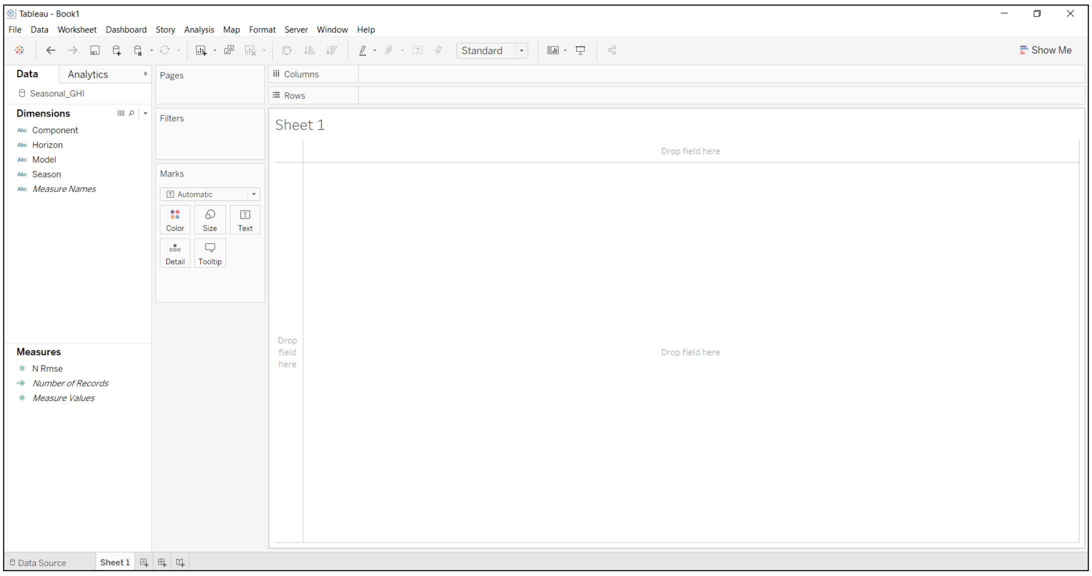


Figure 41: Screenshot of Sheet1 after loading data

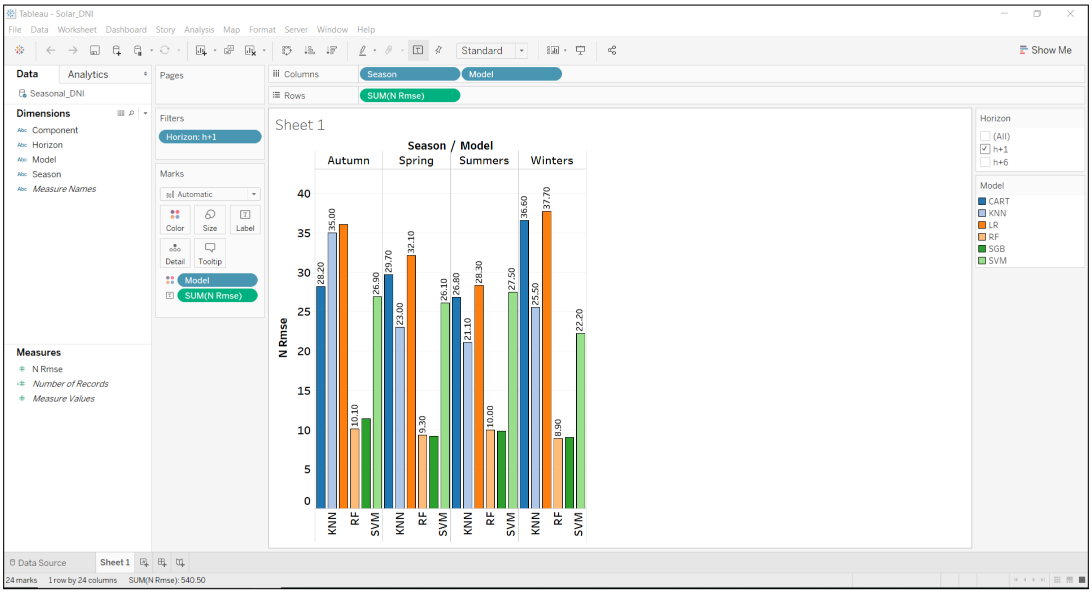


Figure 42: Screenshot after arranging the data into rows and columns