usgs_13296000

Step 0: Load packages

Step 1: Load PRISM and USGS

Step 0: Load packages

Step 1: Load PRISM and USGS

```r
prism_df<-readRDS("prism_df_13296000.rds")
summary(prism_df)
```

```
##       Date                  yr           vpdmax      mean_AirTemperature_C
##  Min.   :2011-10-05   Min.   :2011   Min.   : 0.46   Min.   :-21.800      
##  1st Qu.:2014-03-22   1st Qu.:2014   1st Qu.: 3.09   1st Qu.: -4.500      
##  Median :2016-09-21   Median :2016   Median : 6.97   Median :  2.100      
##  Mean   :2016-11-24   Mean   :2016   Mean   :10.94   Mean   :  2.805      
##  3rd Qu.:2019-07-20   3rd Qu.:2019   3rd Qu.:17.25   3rd Qu.: 10.600      
##  Max.   :2022-09-29   Max.   :2022   Max.   :45.45   Max.   : 22.000      
##  mean_AirTemperature_C_1 max_AirTemperature_C max_AirTemperature_C_1
##  Min.   :-21.800         Min.   :-15.50       Min.   :-15.50        
##  1st Qu.: -4.500         1st Qu.:  1.30       1st Qu.:  1.30        
##  Median :  2.100         Median :  8.70       Median :  8.70        
##  Mean   :  2.812         Mean   : 10.16       Mean   : 10.17        
##  3rd Qu.: 10.700         3rd Qu.: 19.50       3rd Qu.: 19.60        
##  Max.   : 22.000         Max.   : 32.90       Max.   : 32.90        
##    log_mean_Q     max_StreamTemp   mean_StreamTemp        mo        
##  Min.   :3.332   Min.   : 0.000   Min.   : 0.000   Min.   : 1.000  
##  1st Qu.:4.015   1st Qu.: 0.300   1st Qu.: 0.200   1st Qu.: 3.000  
##  Median :4.328   Median : 6.000   Median : 3.900   Median : 7.000  
##  Mean   :4.722   Mean   : 6.591   Mean   : 4.945   Mean   : 6.559  
##  3rd Qu.:5.170   3rd Qu.:11.600   3rd Qu.: 8.900   3rd Qu.:10.000  
##  Max.   :8.023   Max.   :20.400   Max.   :16.100   Max.   :12.000  
##       doy       
##  Min.   :  1.0  
##  1st Qu.: 90.0  
##  Median :186.0  
##  Mean   :184.3  
##  3rd Qu.:279.0  
##  Max.   :366.0  
```

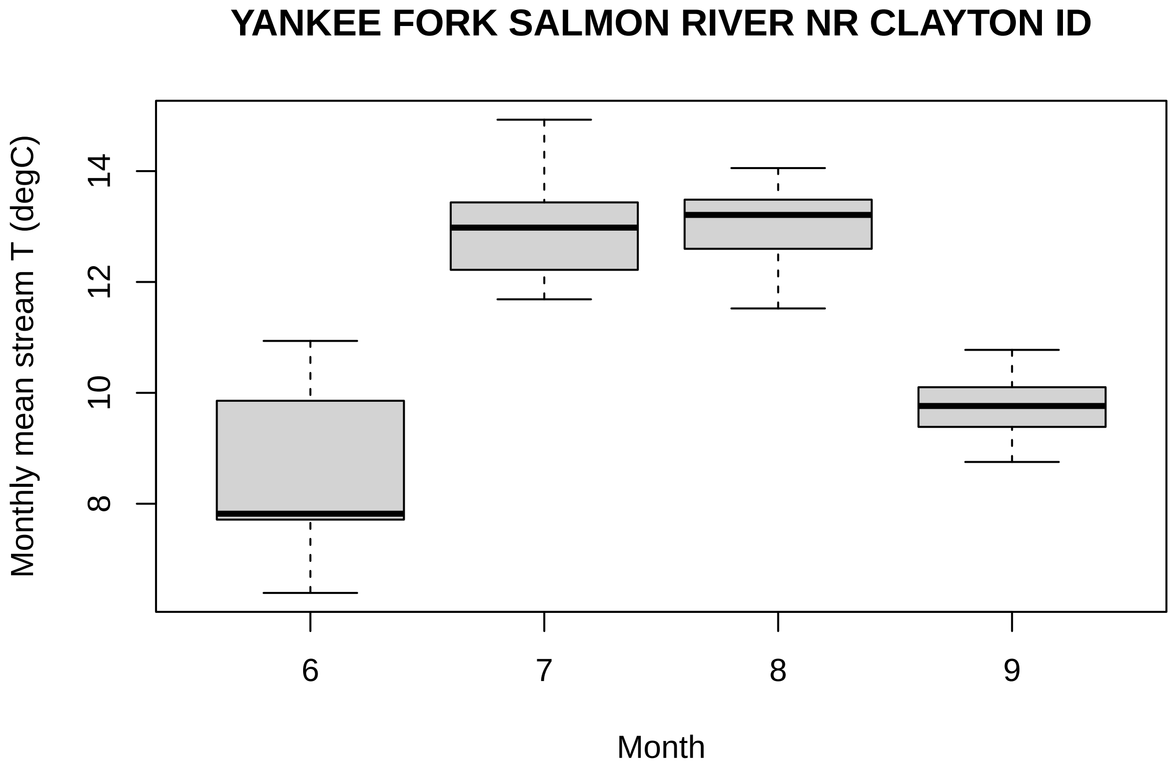
```r
prism_df2<-prism_df
prism_df2$yr<-as.character(prism_df$yr)
prism_df2$mo<-as.character(prism_df$mo)
max_ST_yr<-prism_df2 %>% group_by( yr ,  mo  )%>% summarise(max_ST = max(mean_StreamTemp),mean_ST=mean(
```

```
## `summarise()` has grouped output by 'yr'. You can override using the `.groups`
## argument.
```

```
max_ST_yr[max_ST_yr$mo %in% c("6",  "7" , "8","9"),]
```

```
## # A tibble: 43 x 4
## # Groups:   yr [11]
##    yr    mo    max_ST mean_ST
##    <chr> <chr>  <dbl>   <dbl>
##  1 2012  6       10.7    7.52
##  2 2012  7       14.7   13.0
##  3 2012  8       14     11.5
##  4 2012  9       11.1    8.75
##  5 2013  6       14.6    9.84
##  6 2013  7       15.3   14.1
##  7 2013  8       14.5   13.4
##  8 2013  9       14.1   10.6
##  9 2014  6        9.6    7.65
## 10 2014  7       14.5   13.0
## # ... with 33 more rows
```

```
boxplot(max_ST_yr[max_ST_yr$mo %in% c("6",  "7" , "8","9"),]$mean_ST~max_ST_yr[max_ST_yr$mo %in% c("6",
```



Check missing data

```
table(prism_df[prism_df$mo  %in% c(6,7,8),]$yr)
```

```
##
## 2012 2013 2014 2015 2016 2017 2018 2019 2020 2021 2022
##   92   92   92   92   85   50   92   92   92   35   31
```

```
prism_df<-prism_df[!prism_df$yr %in% c(2017,2021,2022),]
table(prism_df[prism_df$mo  %in% c(6,7,8),]$yr)
```

```
##
## 2012 2013 2014 2015 2016 2018 2019 2020
##   92   92   92   92   85   92   92   92
```

Step 2: Model 0

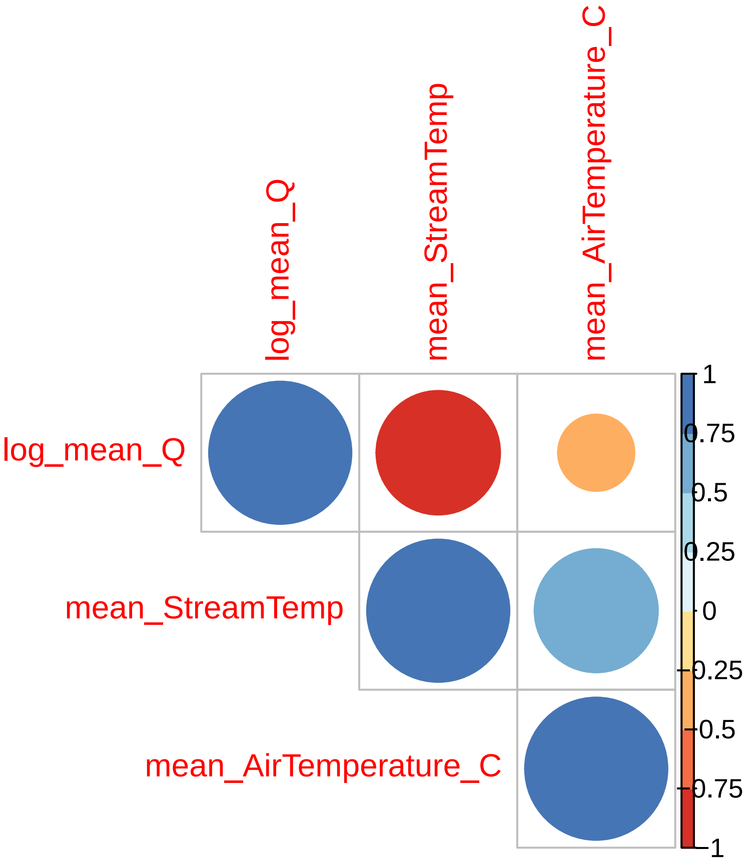
site_id<- 13296000 (USGS 13296000 YANKEE FORK SALMON RIVER NR CLAYTON ID)

Specify year and month for analysis: c(6)

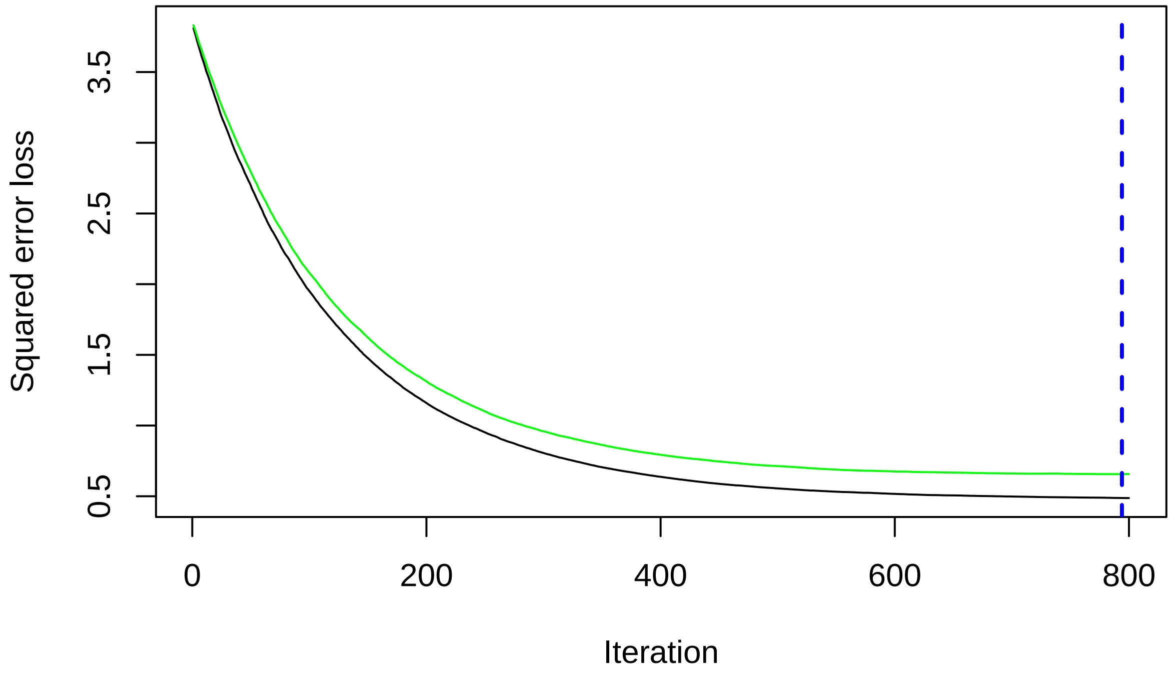
Specify variables<-c(“mean_StreamTemp” ,“log_mean_Q”, “mean_AirTemperature_C”)

v<-“Q_T”

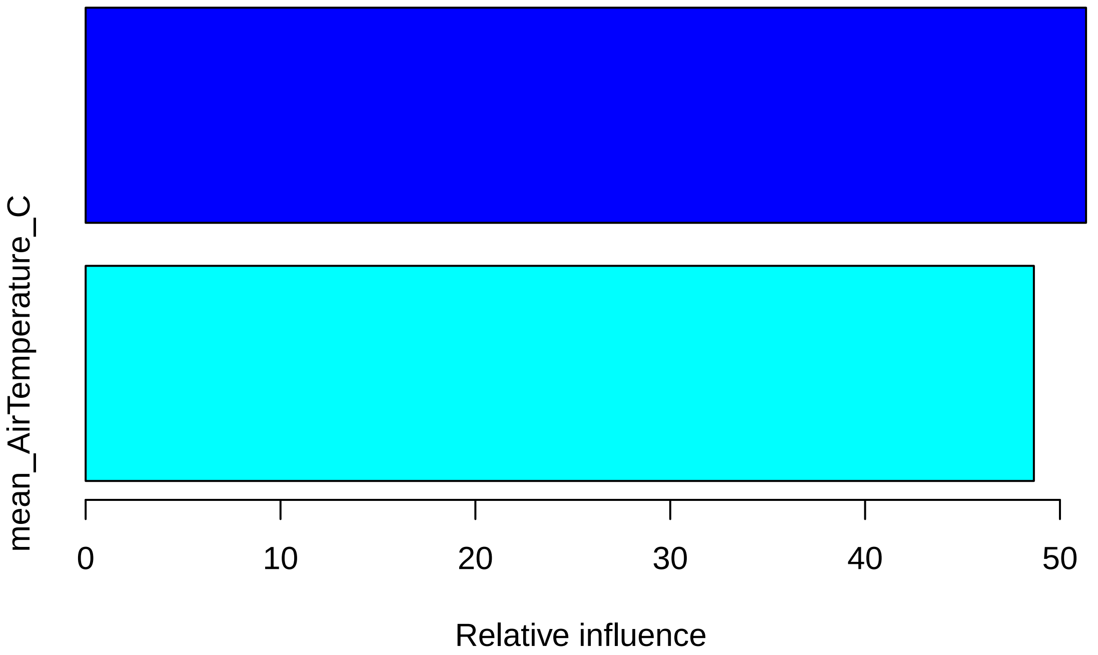
```
site_id<- 13296000
name<-"USGS 13296000 YANKEE FORK SALMON RIVER NR CLAYTON ID"
daily_df_summer<-prism_df[prism_df$mo %in% c(6),]
# Create the correlation plot
M <-cor( daily_df_summer[,c("mean_StreamTemp"
                            ,"max_StreamTemp"
                        ,"log_mean_Q"
                        ,"mean_AirTemperature_C"
          ,"mean_AirTemperature_C_1"
          ,"max_AirTemperature_C_1"
     ,"max_AirTemperature_C"
     ,"vpdmax"
     #,"doy"
     )])
variables<-c("mean_StreamTemp" ,"log_mean_Q", "mean_AirTemperature_C" )
v<-"Q_T"

M_1 <-cor( daily_df_summer[, variables ])
corrplot(M_1, type="upper", order="hclust",
         col=brewer.pal(n=8, name="RdYlBu"))
```



```
# set seed for generating random data.
set.seed(0)
# createDataPartition() function from the caret package to split the original dataset into a training a
parts = createDataPartition(  daily_df_summer$mean_StreamTemp , p = .8, list = F)
train = daily_df_summer[parts, variables ]
test =  daily_df_summer[-parts, variables  ]
# feature and target array
test_x = test[, -1]
test_y = test[, 1]
model_gbm = gbm(train$mean_StreamTemp      ~.,
                data = train,
                distribution = "gaussian",
                cv.folds = 10,
                shrinkage = .01,
                n.minobsinnode = 10,
                n.trees = 800)
# model performance
perf_gbm1 = gbm.perf( model_gbm, method = "cv")
```



```
rinf<-summary(model_gbm)
```

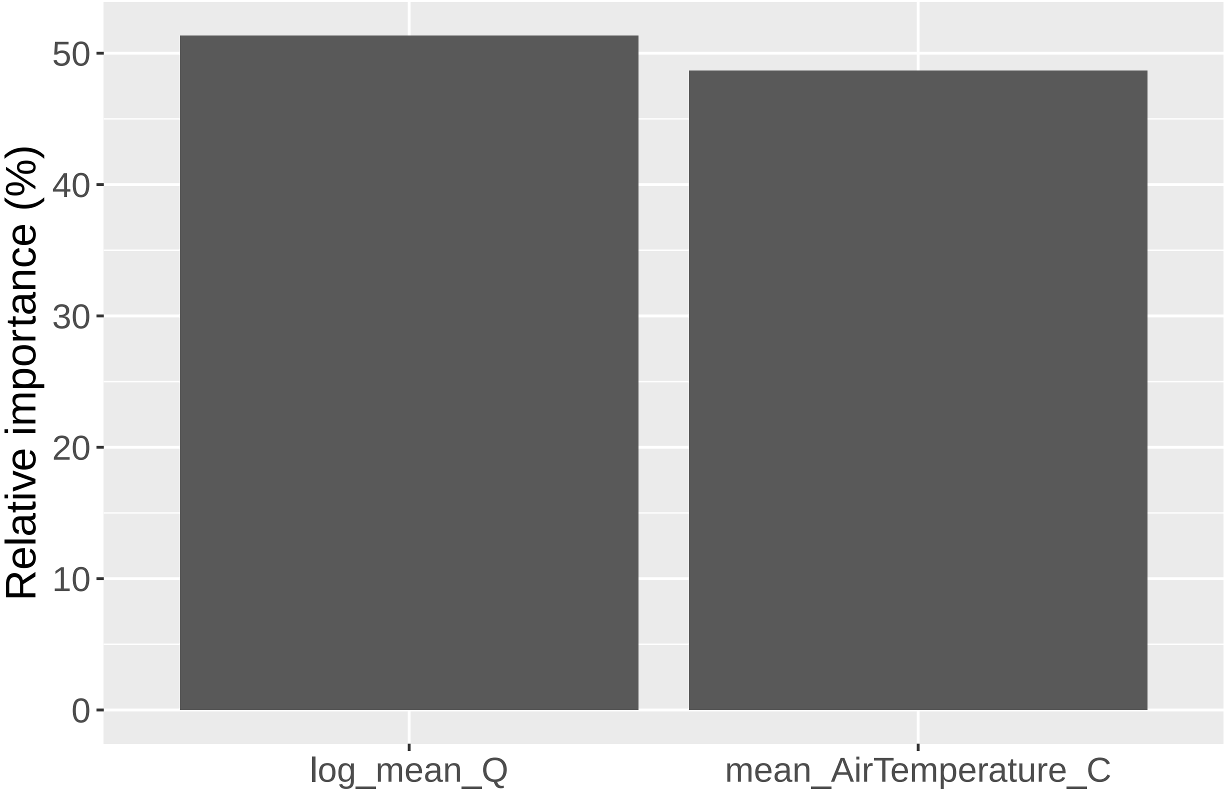


```
rinf$max_yr<-max(as.numeric(daily_df_summer$yr))
rinf$min_yr<-min(as.numeric(daily_df_summer$yr))
rinf$max_mo<-max(as.numeric(daily_df_summer$mo))
rinf$min_mo<-min(as.numeric(daily_df_summer$mo))
rinf$site_id<- site_id
rinf
```

```
##                                          var  rel.inf max_yr min_yr max_mo
## log_mean_Q                       log_mean_Q 51.33864   2020   2012      6
## mean_AirTemperature_C mean_AirTemperature_C 48.66136   2020   2012      6
##                         min_mo  site_id
## log_mean_Q                   6 13296000
## mean_AirTemperature_C        6 13296000
```

```
saveRDS(rinf ,file=   paste("rinf",site_id,rinf$min_mo[1],rinf$max_mo[1],v,".rds",sep="_")  )
rinf$var<- factor(rinf$var, levels=c(  variables[-1]  ))
ggplot( rinf )+  geom_bar(  aes( x=var, y= rel.inf),  stat = "summary")+ scale_x_discrete(labels=  vari
```

```
## No summary function supplied, defaulting to `mean_se()`
```



```
#test_y <-test_y$max_StreamTemp
pred_y = predict.gbm(model_gbm, test_x)
```

```
## Using 794 trees...
```

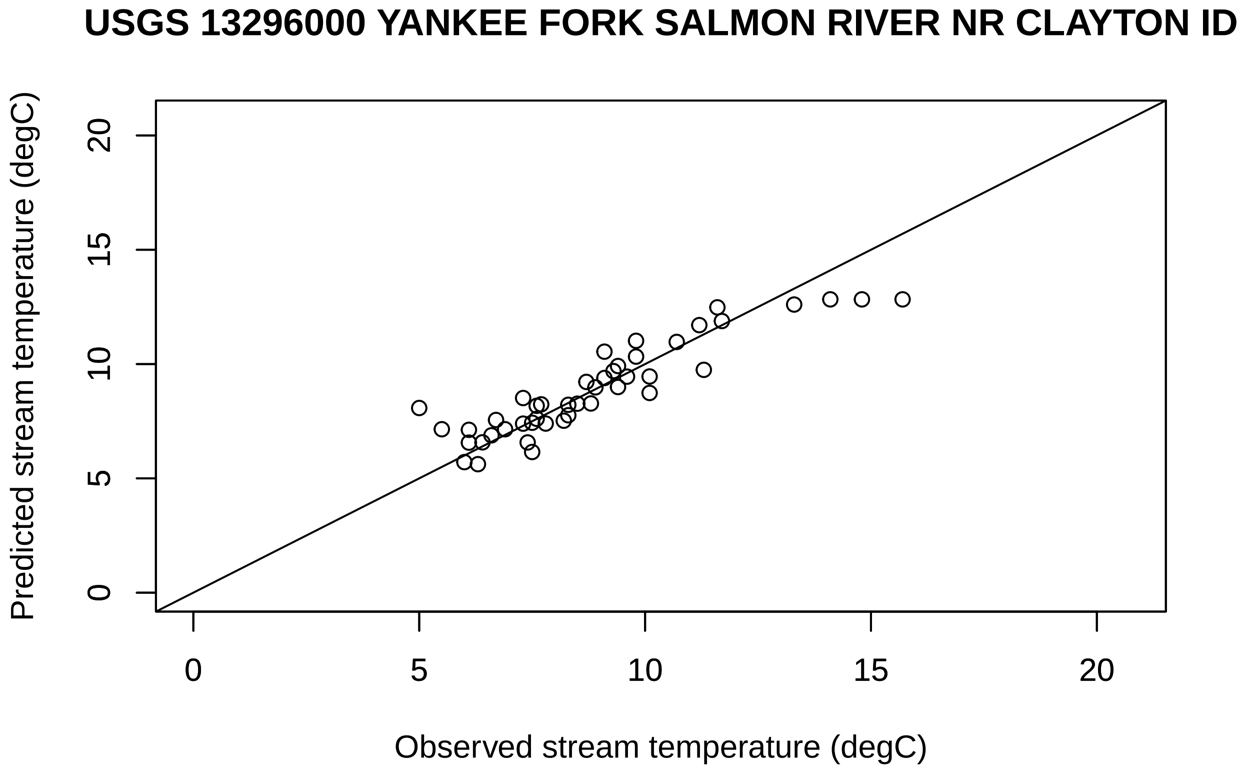
```
residuals =  test_y   - pred_y
xlim=c(min(test_y)-5,max(test_y)+5)
RMSE = sqrt(mean(residuals^2))
cat('The root mean square error of the test data is ', round(RMSE,3),'\n')
```

```
## The root mean square error of the test data is  1.007
```

```
y_test_mean = mean( test_y  )
# Calculate total sum of squares
tss =  sum(( test_y   - y_test_mean)^2 )
# Calculate residual sum of squares
rss =  sum(residuals^2)
# Calculate R-squared
rsq  =  1 - (rss/tss)
cat('The R-square of the test data is ', round(rsq,3), '\n')
```

```
## The R-square of the test data is  0.824
```

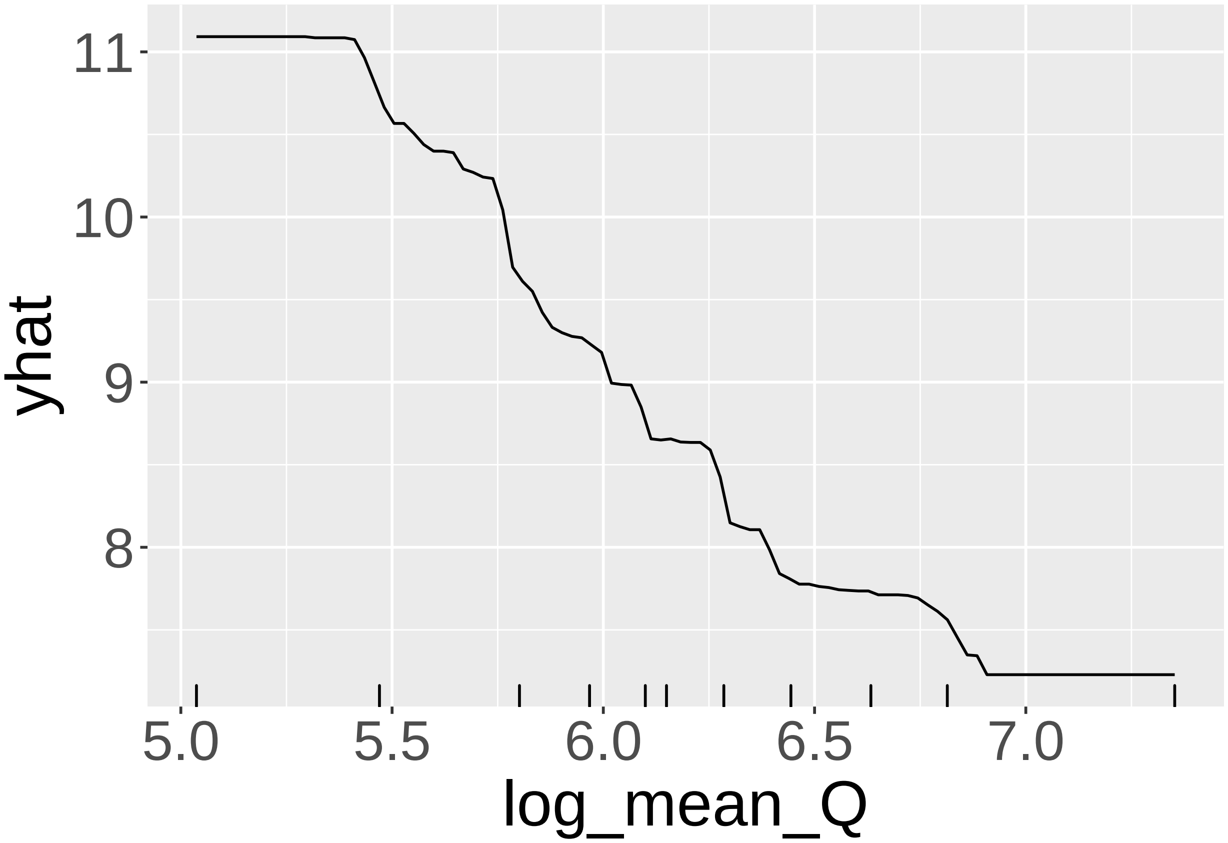
```
plot(  test_y , pred_y,xlim= xlim ,ylim= xlim, xlab="Observed stream temperature (degC)", ylab="Predicte
par(new=T)
x=c(min(test_y)-10,max(test_y)+10)
plot(x,x,type="l",xlim= xlim ,ylim= xlim,xlab="",ylab="")
```



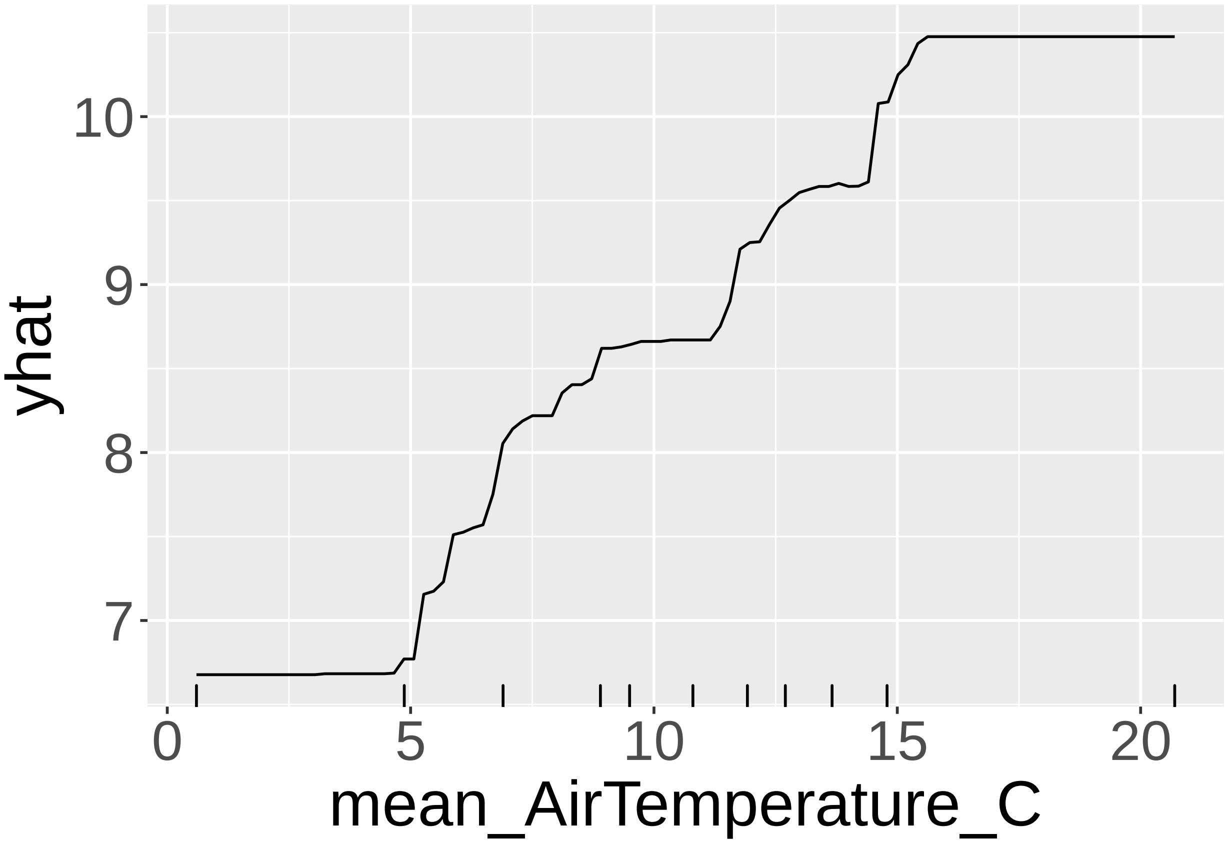
```
length(variables)
```

```
## [1] 3
```

```
model_gbm %>%
  pdp::partial(pred.var =  variables[2], n.trees = model_gbm$n.trees, grid.resolution = 100)%>%
  autoplot(rug = TRUE, train = train)+theme(axis.text=element_text(size=21),
        axis.title=element_text(size=24))
```



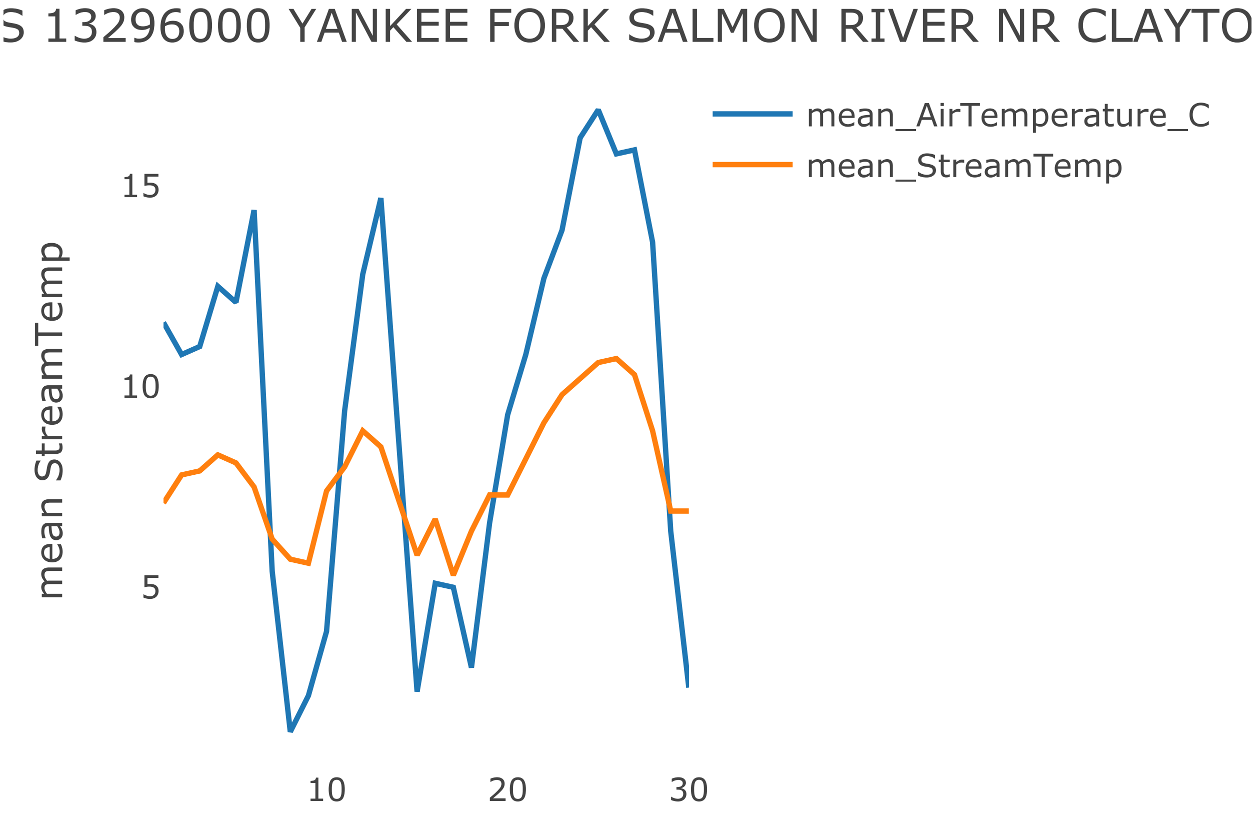
```
model_gbm %>%
  pdp::partial(pred.var =  variables[3], n.trees = model_gbm$n.trees, grid.resolution = 100)%>%
  autoplot(rug = TRUE, train = train)+theme(axis.text=element_text(size=21),
        axis.title=element_text(size=24))
```



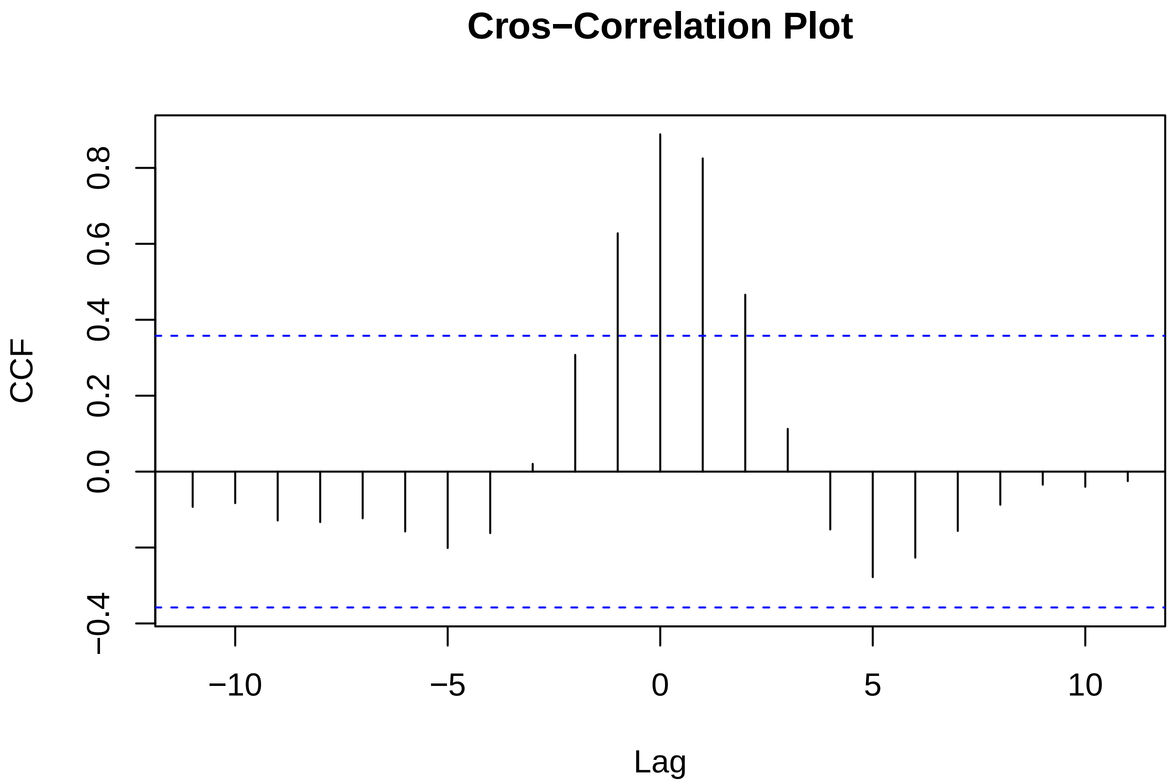
```
unique(daily_df_summer$yr)
```

```
## [1] 2012 2013 2014 2015 2016 2018 2019 2020
```

```
yr=2020
ts <- ts(data = daily_df_summer[daily_df_summer$yr ==yr, c( "mean_AirTemperature_C",  "mean_StreamTemp"
   start = 1,
   end =  dim(daily_df_summer[daily_df_summer$yr ==yr, ])[1],
   frequency = 1)
ts_plot( ts,
        title =  name,
        Ytitle =    "mean StreamTemp",
        Xtitle = " ", )
```



```r
# ccf time series
par(mfrow=c(1,1))
ccf( ts[, c( "mean_AirTemperature_C")],  ts[, c( "mean_StreamTemp")],
    lag.max = 11,
    main = "Cros-Correlation Plot",
    ylab = "CCF")
```



Step 2: Model 1

site_id<- 13296000 (USGS 13296000 YANKEE FORK SALMON RIVER NR CLAYTON ID)

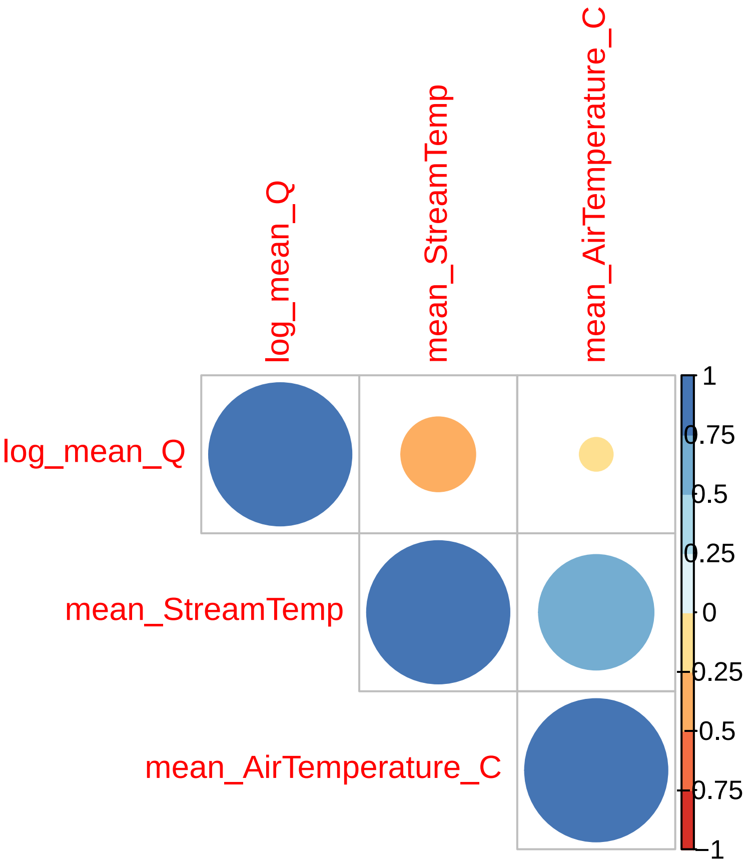
Specify year and month for analysis: c(7,8)

Specify variables<-c("mean_StreamTemp" ,"log_mean_Q", "mean_AirTemperature_C")

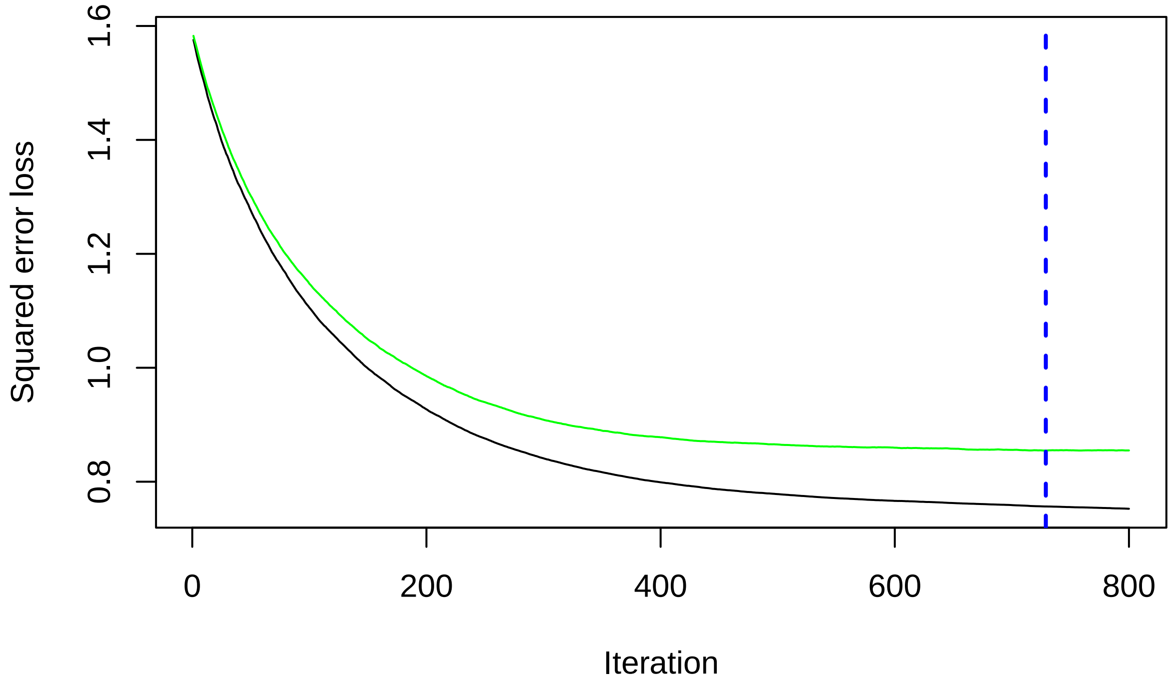
v<-"Q_T"

```
site_id<- 13296000
name<-"USGS 13296000 YANKEE FORK SALMON RIVER NR CLAYTON ID"
daily_df_summer<-prism_df[prism_df$mo %in% c(7,8),]
# Create the correlation plot
M <-cor( daily_df_summer[,c("mean_StreamTemp"
                              ,"max_StreamTemp"
                          ,"log_mean_Q"
                          ,"mean_AirTemperature_C"
          ,"mean_AirTemperature_C_1"
          ,"max_AirTemperature_C_1"
     ,"max_AirTemperature_C"
     ,"vpdmax"
     #,"doy"
     )])
variables<-c("mean_StreamTemp" ,"log_mean_Q", "mean_AirTemperature_C" )
v<-"Q_T"
```

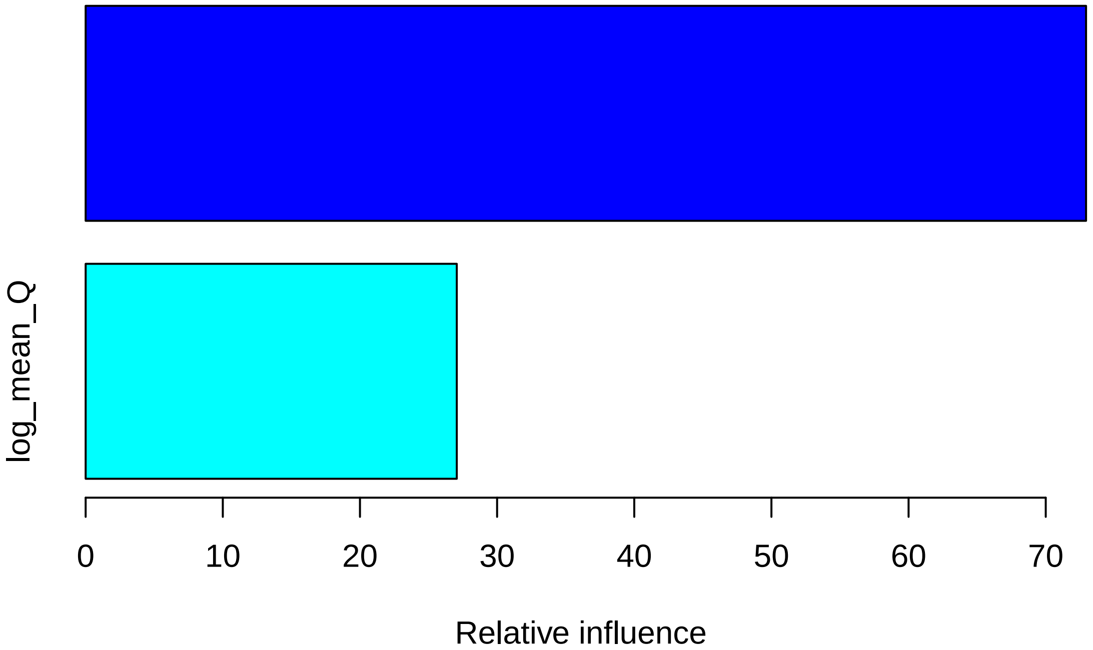
```
M_1 <-cor( daily_df_summer[, variables ])
corrplot(M_1, type="upper", order="hclust",
         col=brewer.pal(n=8, name="RdYlBu"))
```



```
# set seed for generating random data.
set.seed(0)
# createDataPartition() function from the caret package to split the original dataset into a training a
parts = createDataPartition(  daily_df_summer$mean_StreamTemp , p = .8, list = F)
train = daily_df_summer[parts, variables ]
test =  daily_df_summer[-parts, variables  ]
# feature and target array
test_x = test[, -1]
test_y = test[, 1]
model_gbm = gbm(train$mean_StreamTemp      ~.,
                data = train,
                distribution = "gaussian",
                cv.folds = 10,
                shrinkage = .01,
                n.minobsinnode = 10,
                n.trees = 800)
# model performance
perf_gbm1 = gbm.perf( model_gbm, method = "cv")
```



```
rinf<-summary(model_gbm)
```

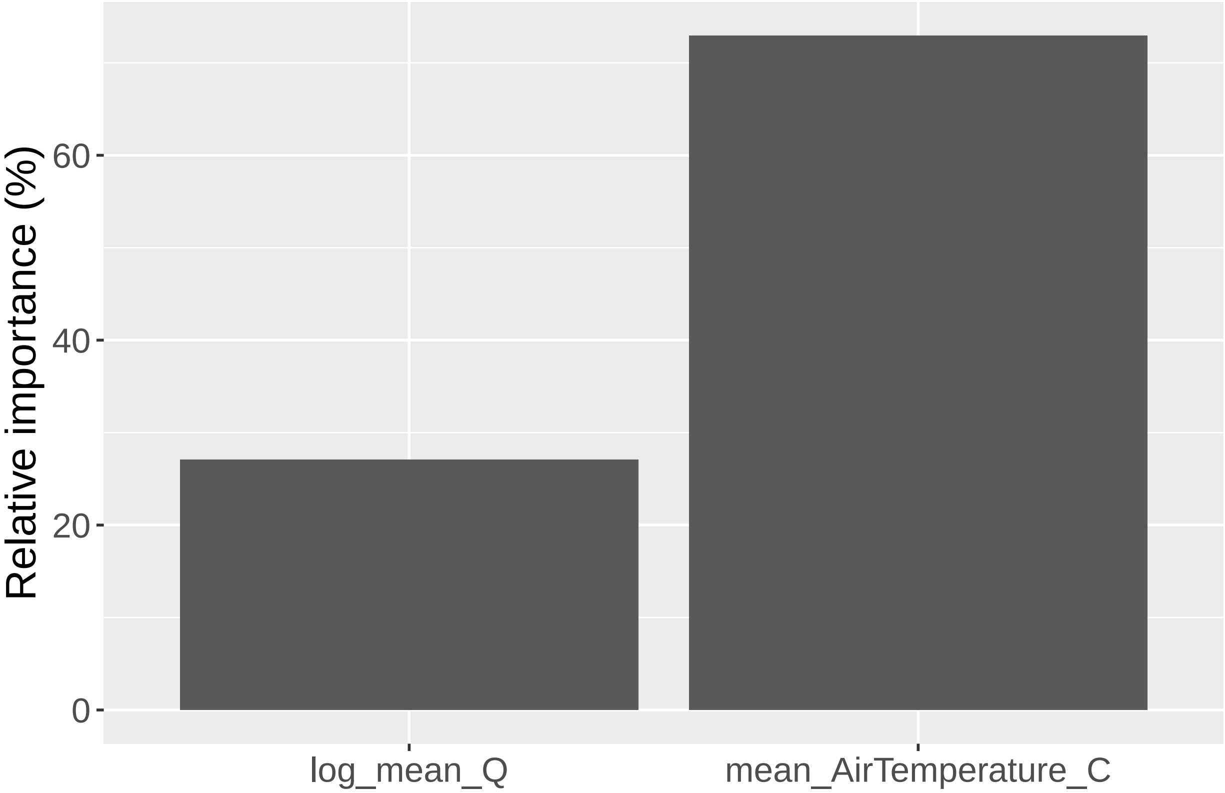


```
rinf$max_yr<-max(as.numeric(daily_df_summer$yr))
rinf$min_yr<-min(as.numeric(daily_df_summer$yr))
rinf$max_mo<-max(as.numeric(daily_df_summer$mo))
rinf$min_mo<-min(as.numeric(daily_df_summer$mo))
rinf$site_id<- site_id
rinf
```

```
##                                               var  rel.inf max_yr min_yr max_mo
## mean_AirTemperature_C mean_AirTemperature_C 72.93972   2020   2012      8
## log_mean_Q                           log_mean_Q 27.06028   2020   2012      8
##                          min_mo  site_id
## mean_AirTemperature_C      7 13296000
## log_mean_Q                 7 13296000
```

```
saveRDS(rinf ,file=   paste("rinf",site_id,rinf$min_mo[1],rinf$max_mo[1],v,".rds",sep="_")  )
rinf$var<- factor(rinf$var, levels=c(  variables[-1]  ))
ggplot( rinf )+  geom_bar(  aes( x=var, y= rel.inf),  stat = "summary")+ scale_x_discrete(labels=  varia
```

```
## No summary function supplied, defaulting to `mean_se()`
```



```
#test_y <-test_y$max_StreamTemp
pred_y = predict.gbm(model_gbm, test_x)
```

```
## Using 729 trees...
```

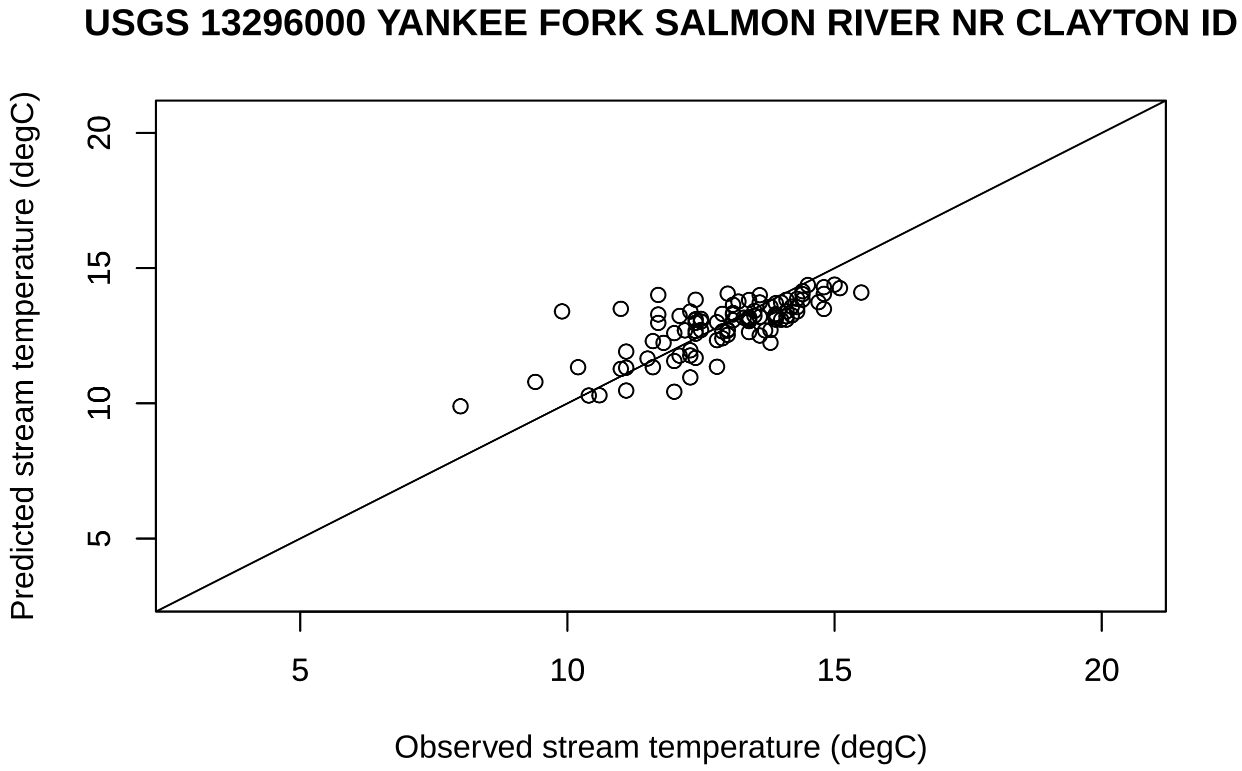
```
residuals =  test_y   - pred_y
xlim=c(min(test_y)-5,max(test_y)+5)
RMSE = sqrt(mean(residuals^2))
cat('The root mean square error of the test data is ', round(RMSE,3),'\n')
```

```
## The root mean square error of the test data is  0.891
```

```
y_test_mean = mean( test_y  )
# Calculate total sum of squares
tss =  sum(( test_y   - y_test_mean)^2 )
# Calculate residual sum of squares
rss =  sum(residuals^2)
# Calculate R-squared
rsq  =  1 - (rss/tss)
cat('The R-square of the test data is ', round(rsq,3), '\n')
```

```
## The R-square of the test data is  0.545
```

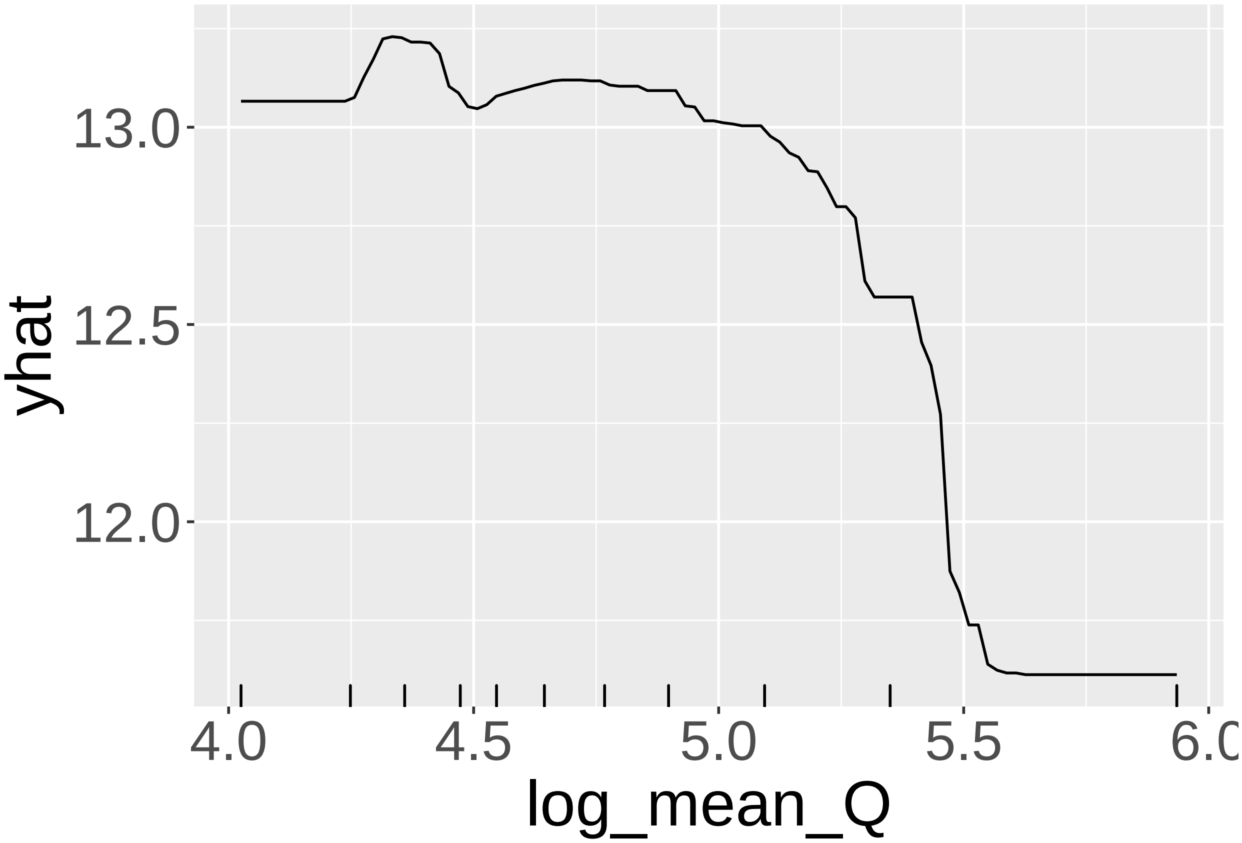
```
plot(  test_y , pred_y,xlim= xlim ,ylim= xlim, xlab="Observed stream temperature (degC)", ylab="Predict
par(new=T)
x=c(min(test_y)-10,max(test_y)+10)
plot(x,x,type="l",xlim= xlim ,ylim= xlim,xlab="",ylab="")
```



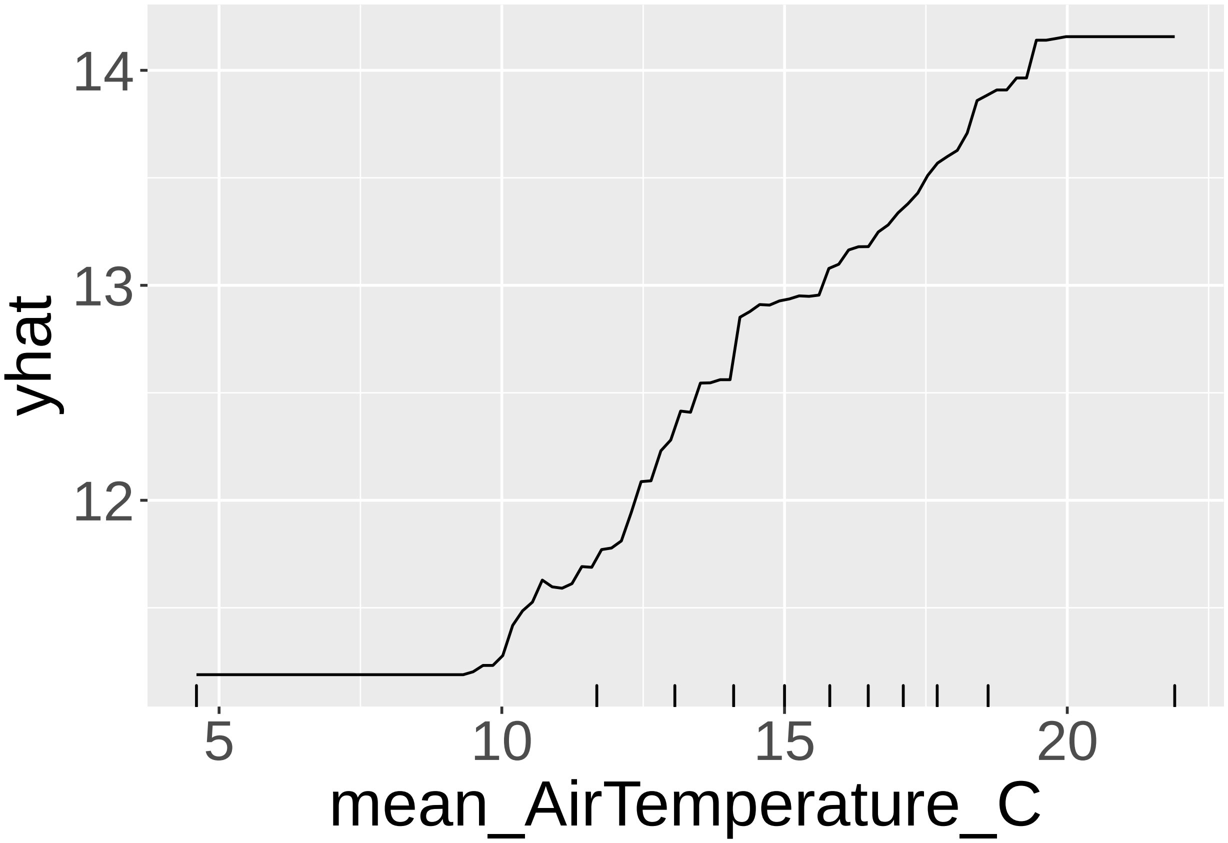
```
length(variables)
```

```
## [1] 3
```

```
model_gbm %>%
  pdp::partial(pred.var =  variables[2], n.trees = model_gbm$n.trees, grid.resolution = 100)%>%
  autoplot(rug = TRUE, train = train)+theme(axis.text=element_text(size=21),
        axis.title=element_text(size=24))
```



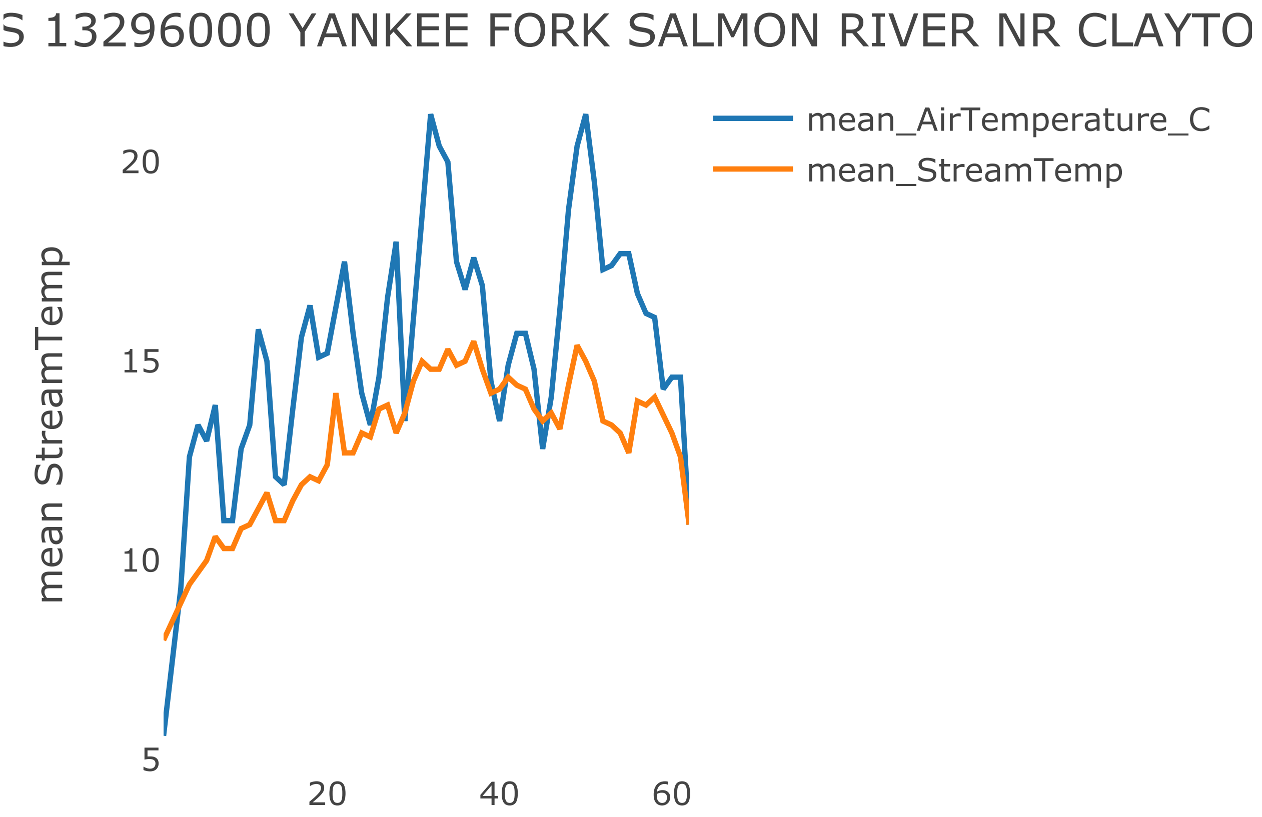
```
model_gbm %>%
  pdp::partial(pred.var =  variables[3], n.trees = model_gbm$n.trees, grid.resolution = 100)%>%
  autoplot(rug = TRUE, train = train)+theme(axis.text=element_text(size=21),
        axis.title=element_text(size=24))
```



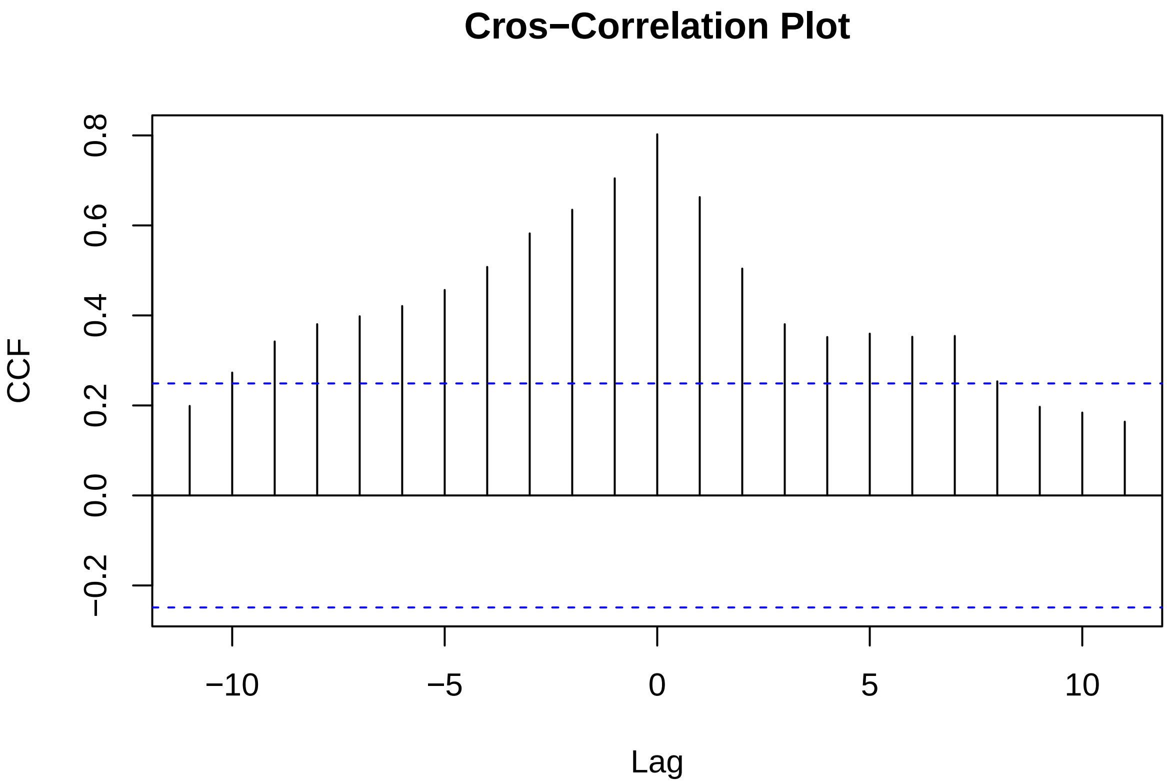
```
unique(daily_df_summer$yr)
```

```
## [1] 2012 2013 2014 2015 2016 2018 2019 2020
```

```
yr=2020
ts <- ts(data = daily_df_summer[daily_df_summer$yr ==yr, c( "mean_AirTemperature_C",  "mean_StreamTemp"
   start = 1,
   end =  dim(daily_df_summer[daily_df_summer$yr ==yr, ])[1],
   frequency = 1)
ts_plot( ts,
        title =  name,
        Ytitle =    "mean StreamTemp",
        Xtitle = " ", )
```



```
# ccf time series
par(mfrow=c(1,1))
ccf( ts[, c( "mean_AirTemperature_C")],  ts[, c( "mean_StreamTemp")],
    lag.max = 11,
    main = "Cros-Correlation Plot",
    ylab = "CCF")
```


Cros−Correlation Plot
CCF
−0.2
0.0
0.2
0.4
0.6
0.8
−10
−5
0
5
10
Lag