# USGS_13311000

## Step 0: Load packages

## Step 1: Load PRISM and USGS

## Step 2: Model 1

specify year and month for analysis # Step 6: Create the correlation plot # Step 7: create training and test data

## Step 0: Load packages

## Step 1: Load PRISM and USGS

```
prism_df<-readRDS("prism_df_13311000.rds")
summary(prism_df)
```

```
##       Date                  yr           vpdmax       mean_AirTemperature_C
##  Min.   :2011-10-01   Min.   :2011   Min.   : 0.660   Min.   :-19.900
##  1st Qu.:2014-09-23   1st Qu.:2014   1st Qu.: 6.162   1st Qu.:  2.225
##  Median :2018-06-18   Median :2018   Median :14.750   Median :  8.900
##  Mean   :2018-01-24   Mean   :2018   Mean   :15.799   Mean   :  7.848
##  3rd Qu.:2021-03-28   3rd Qu.:2021   3rd Qu.:24.110   3rd Qu.: 14.400
##  Max.   :2022-09-30   Max.   :2022   Max.   :46.700   Max.   : 22.800
##
##  mean_AirTemperature_C_1 max_AirTemperature_C max_AirTemperature_C_1
##  Min.   :-19.900         Min.   :-12.10       Min.   :-12.10
##  1st Qu.:  2.200         1st Qu.:  8.20       1st Qu.:  8.20
##  Median :  8.900         Median : 17.70       Median : 17.70
##  Mean   :  7.845         Mean   : 15.93       Mean   : 15.93
##  3rd Qu.: 14.400         3rd Qu.: 24.30       3rd Qu.: 24.30
##  Max.   : 22.800         Max.   : 33.20       Max.   : 33.20
##
##    log_mean_Q    max_StreamTemp   mean_StreamTemp        mo
##  Min.   :1.875   Min.   : 0.000   Min.   : 0.000   Min.   : 1.000
##  1st Qu.:2.380   1st Qu.: 5.100   1st Qu.: 3.000   1st Qu.: 5.000
##  Median :2.785   Median : 8.800   Median : 6.100   Median : 7.000
##  Mean   :3.190   Mean   : 8.649   Mean   : 6.127   Mean   : 6.833
##  3rd Qu.:4.018   3rd Qu.:13.000   3rd Qu.: 9.800   3rd Qu.: 9.000
##  Max.   :5.900   Max.   :19.200   Max.   :13.200   Max.   :12.000
##                  NA's   :2
##       doy
##  Min.   :  1.0
##  1st Qu.:143.0
##  Median :197.0
```

```
##  Mean   :192.4
##  3rd Qu.:251.0
##  Max.   :365.0
##
```

Check missing data

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

```
##
## 2012 2013 2014 2016 2017 2018 2019 2020 2021 2022
##   92   92   92   91   92   92   92   92   90   92
```

Plot monthly temp

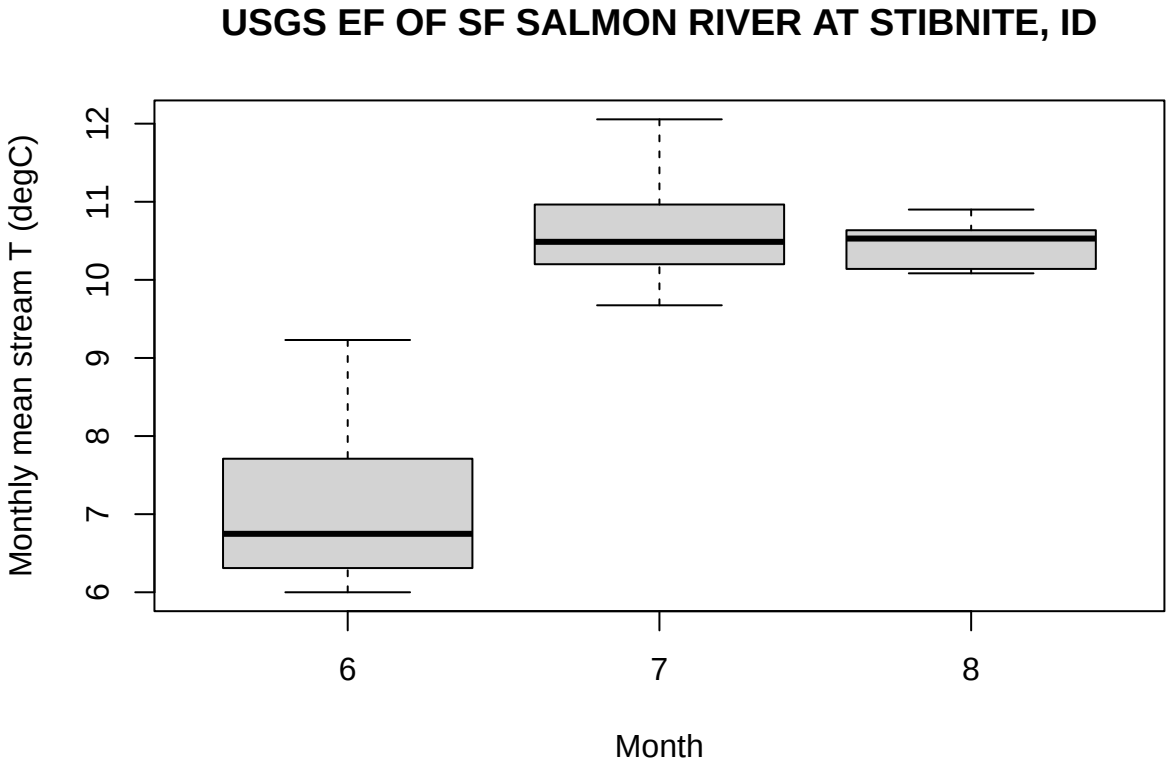
```
prism_df2<-prism_df
prism_df2$yr<-as.character(prism_df$yr)
prism_df2$mo<-as.character(prism_df$mo)
max_ST_yr<-prism_df2 %>% group_by( yr ,  mo  )%>% summarise(max_ST = max(mean_StreamTemp),mean_ST=mean(
```

```
## `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"),]
```

```
## # A tibble: 30 x 4
## # Groups:   yr [10]
##    yr    mo    max_ST mean_ST
##    <chr> <chr>  <dbl>   <dbl>
##  1 2012  6        9.2    6.27
##  2 2012  7       11.9   10.6
##  3 2012  8       12.1   10.6
##  4 2013  6       12.1    7.71
##  5 2013  7       12.9   11.3
##  6 2013  8       11.5   10.8
##  7 2014  6        8.7    6.69
##  8 2014  7       12.6   11.0
##  9 2014  8       11.9   10.3
## 10 2016  6       11.1    8.16
## # ... with 20 more rows
```

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



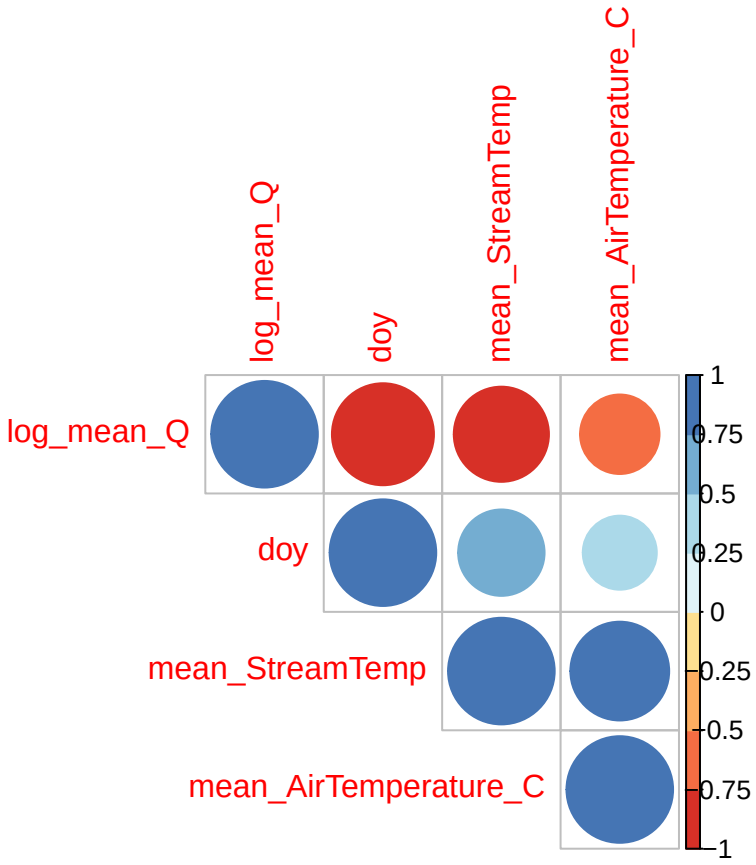

## Step 2: Model 1

**site_id<- 13311000**

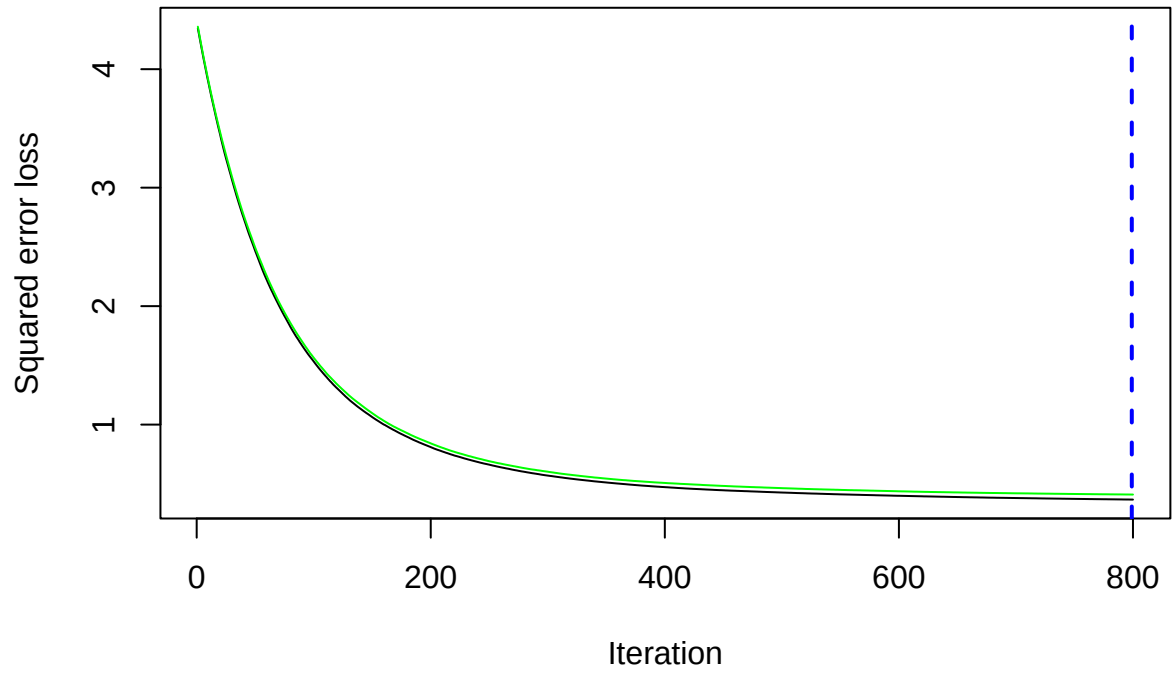
**Specify year and month for analysis: c(6,7,8)**

**Specify variables<-c("mean_StreamTemp" ,"log_mean_Q","doy","mean_AirTemperature_ )**

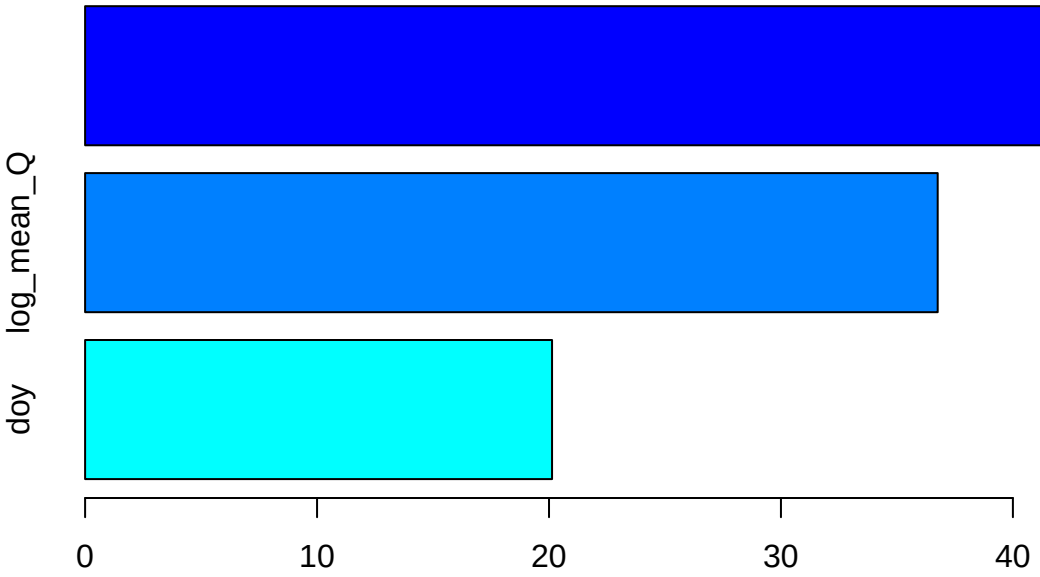
```
site_id<- 13311000
daily_df_summer<-prism_df[prism_df$mo %in% c(6,7,8),]
# Create the correlation plot
M <-cor( daily_df_summer[,c("mean_StreamTemp"
                            ,"max_StreamTemp"
                          ,"log_mean_Q"
                          ,"mean_AirTemperature_C"
          ,"mean_AirTemperature_C_1"
          ,"max_AirTemperature_C_1"
     ,"max_AirTemperature_C"
     ,"vpdmax"
     ,"doy")])

variables<-c("mean_StreamTemp" ,"log_mean_Q","doy","mean_AirTemperature_C" )
v<-"Q_d_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)
```



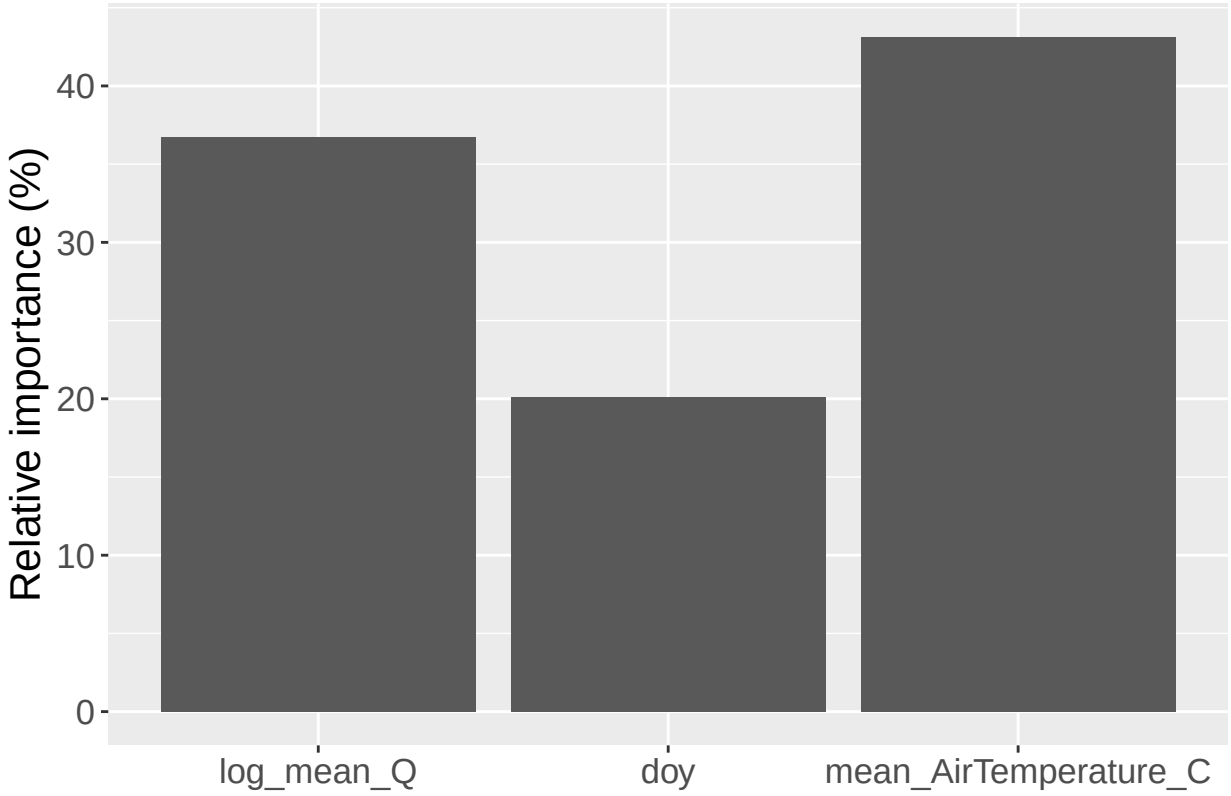

```
rinf$max_yr<-max(as.numeric(daily_df_summer$yr))
rinf$min_yr<-min(as.numeric(daily_df_summer$yr))
rinf$max_mo<-max(as.numeric(daily_df_summer$mo))
rinf$min_mo<-min(as.numeric(daily_df_summer$mo))
rinf$site_id<- site_id
rinf
```

```
##                                     var  rel.inf max_yr min_yr max_mo
## mean_AirTemperature_C mean_AirTemperature_C 43.11100   2022   2012      8
## log_mean_Q                       log_mean_Q 36.75771   2022   2012      8
```

```
## doy                                    doy 20.13129    2022    2012       8
##                         min_mo  site_id
## mean_AirTemperature_C        6 13311000
## log_mean_Q                   6 13311000
## doy                          6 13311000
```

```
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 799 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')
```

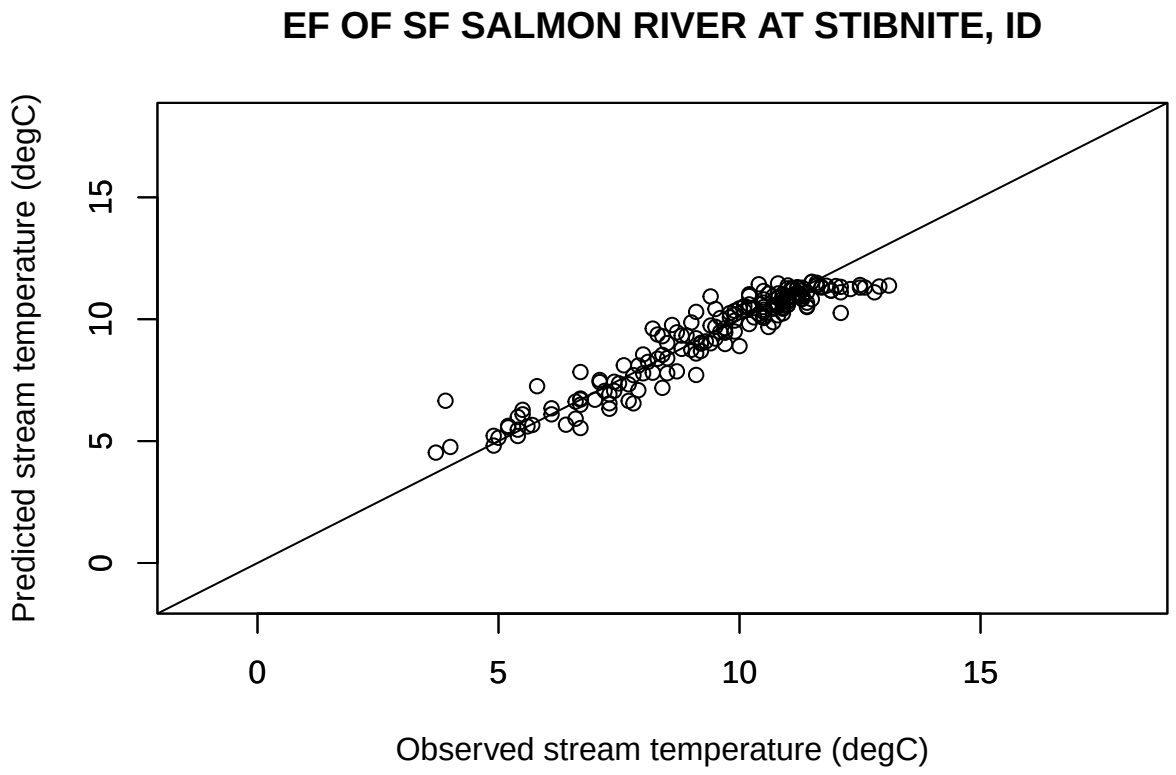
```
## The root mean square error of the test data is  0.65
```

```
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)
```

```r
# 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  0.902
```

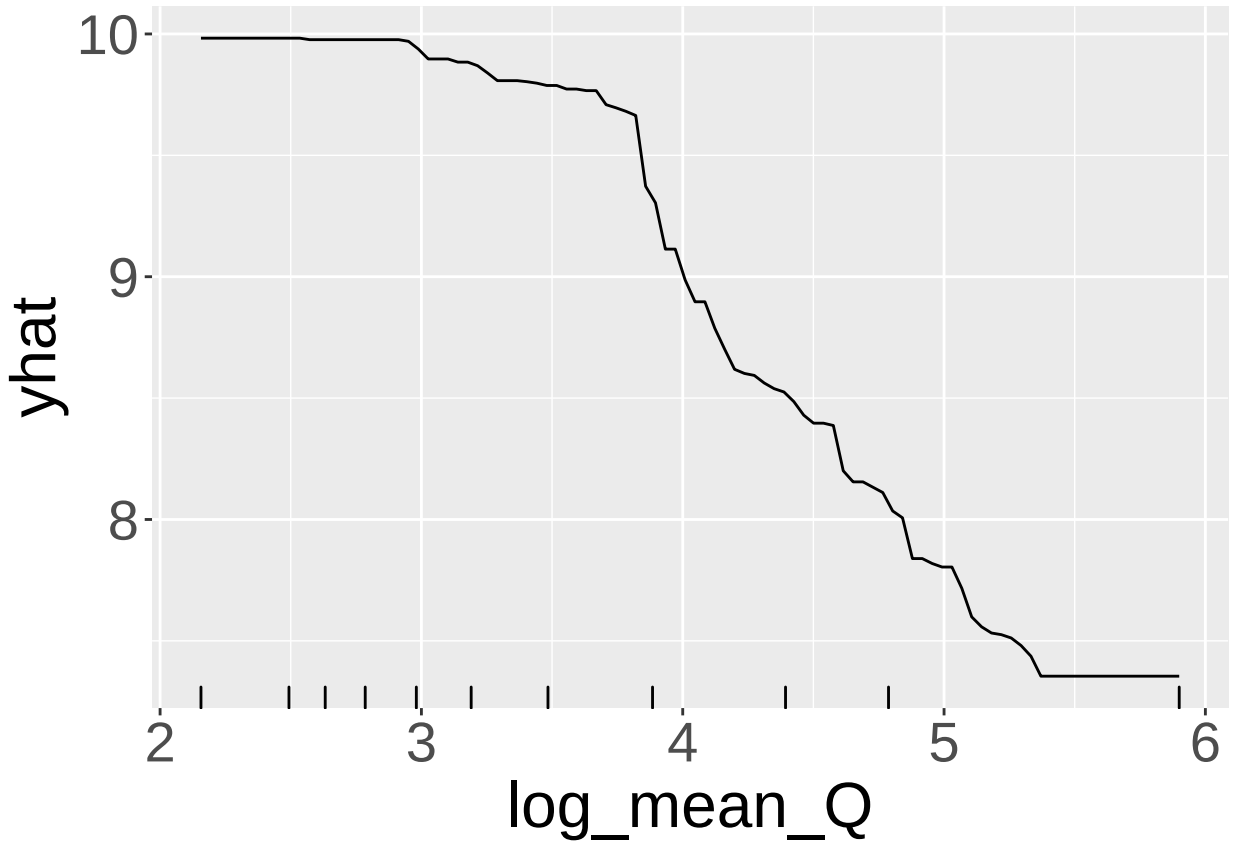
```r
plot(  test_y , pred_y,xlim= xlim ,ylim= xlim, xlab="Observed stream temperature (degC)", ylab="Predicte
par(new=T)
x=c(min(test_y)-10,max(test_y)+10)
plot(x,x,type="l",xlim= xlim ,ylim= xlim,xlab="",ylab="")
```



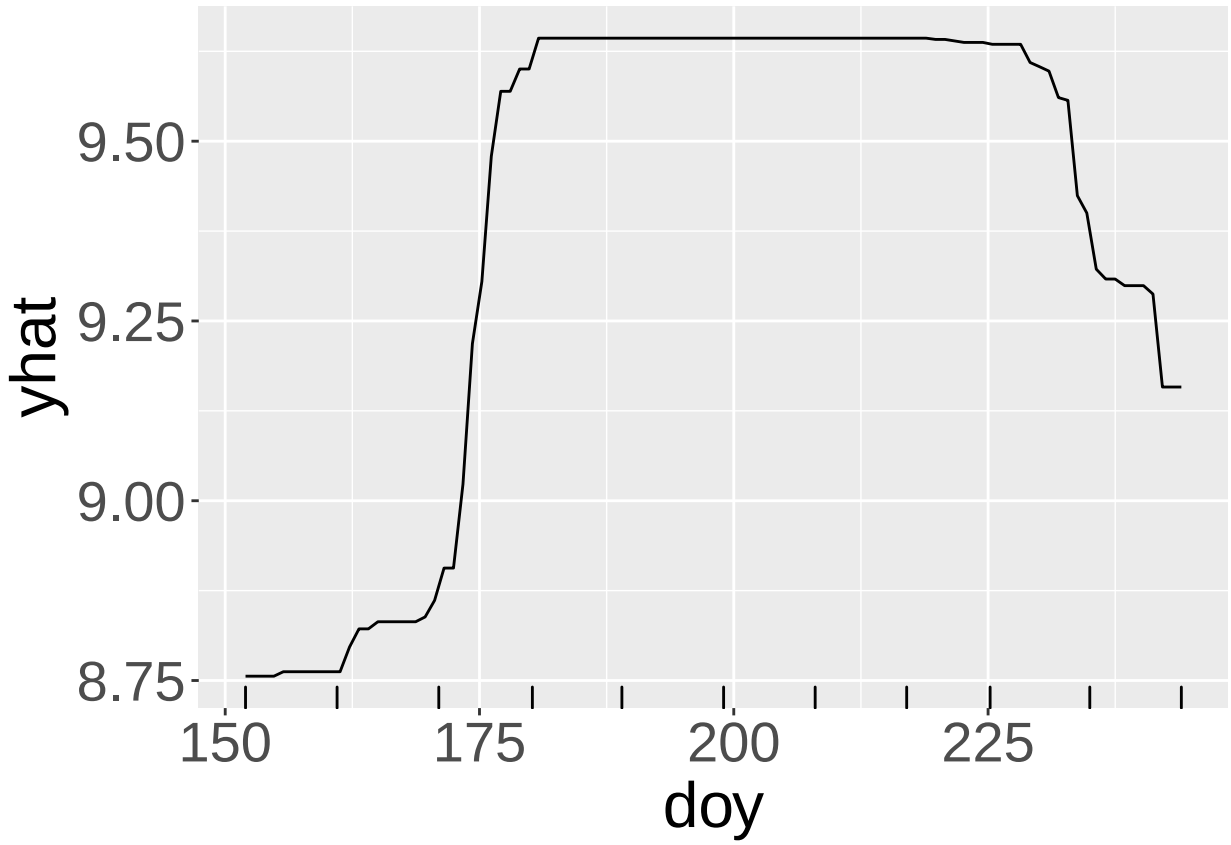

```r
length(variables)
```

```
## [1] 4
```

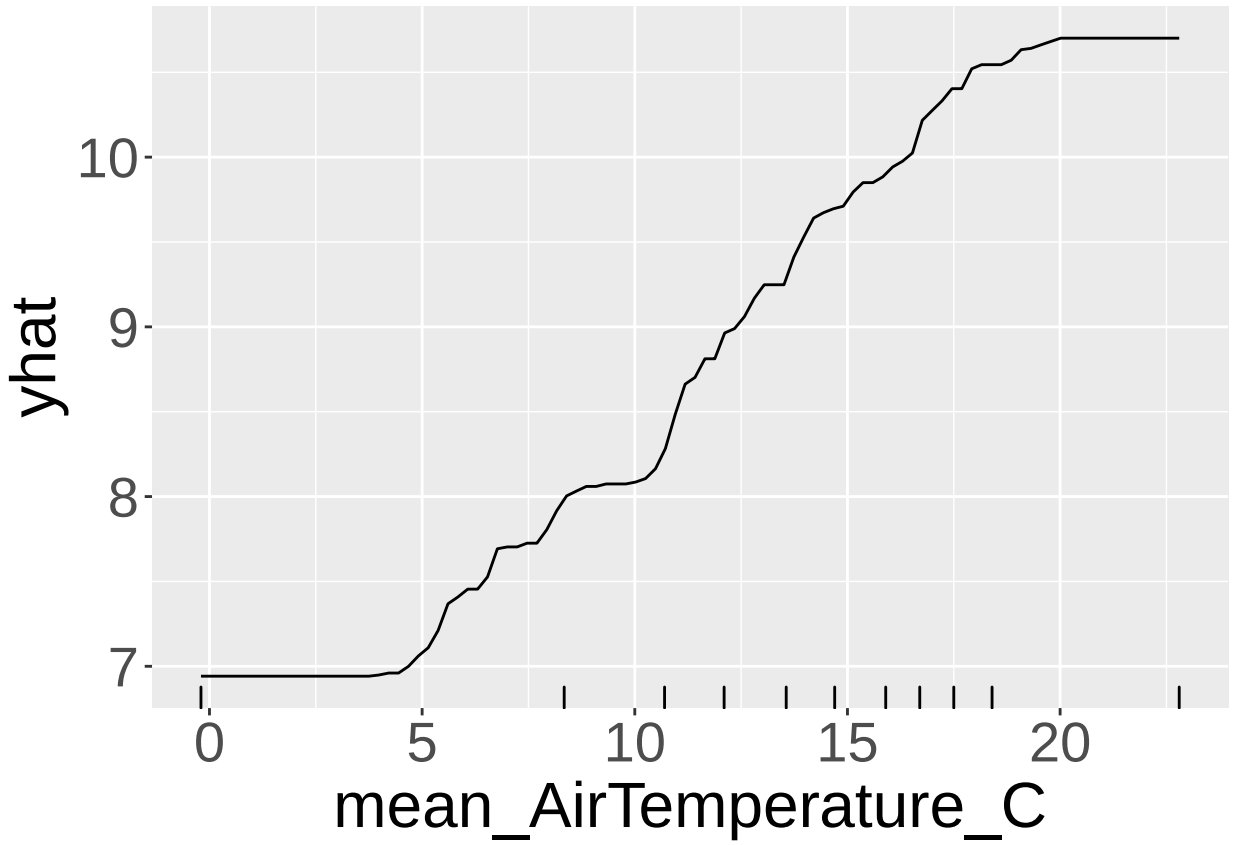
```r
model_gbm %>%
  pdp::partial(pred.var =  variables[2], n.trees = model_gbm$n.trees, grid.resolution = 100)%>%
  autoplot(rug = TRUE, train = train)+theme(axis.text=element_text(size=21),
        axis.title=element_text(size=24))
```




```
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))
```



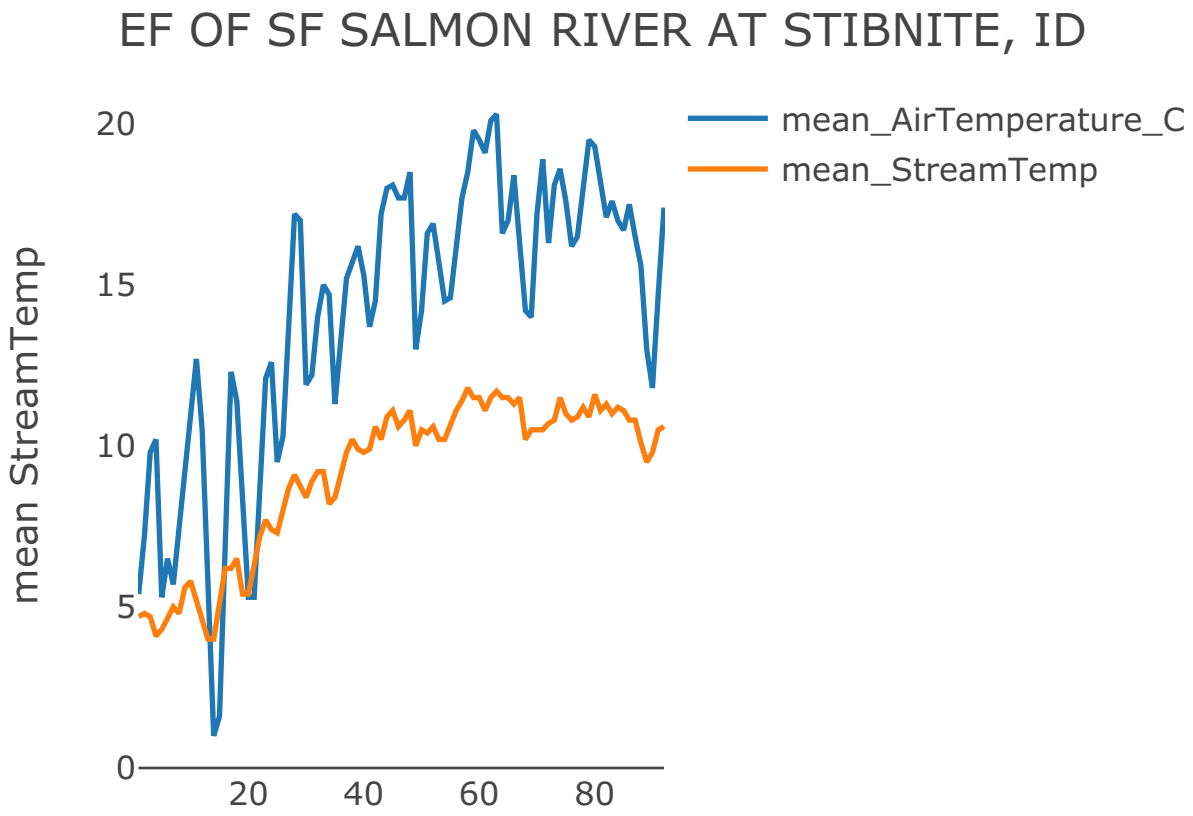

```
model_gbm %>%
  pdp::partial(pred.var =  variables[4], n.trees = model_gbm$n.trees, grid.resolution = 100)%>%
  autoplot(rug = TRUE, train = train)+theme(axis.text=element_text(size=21),
        axis.title=element_text(size=24))
```



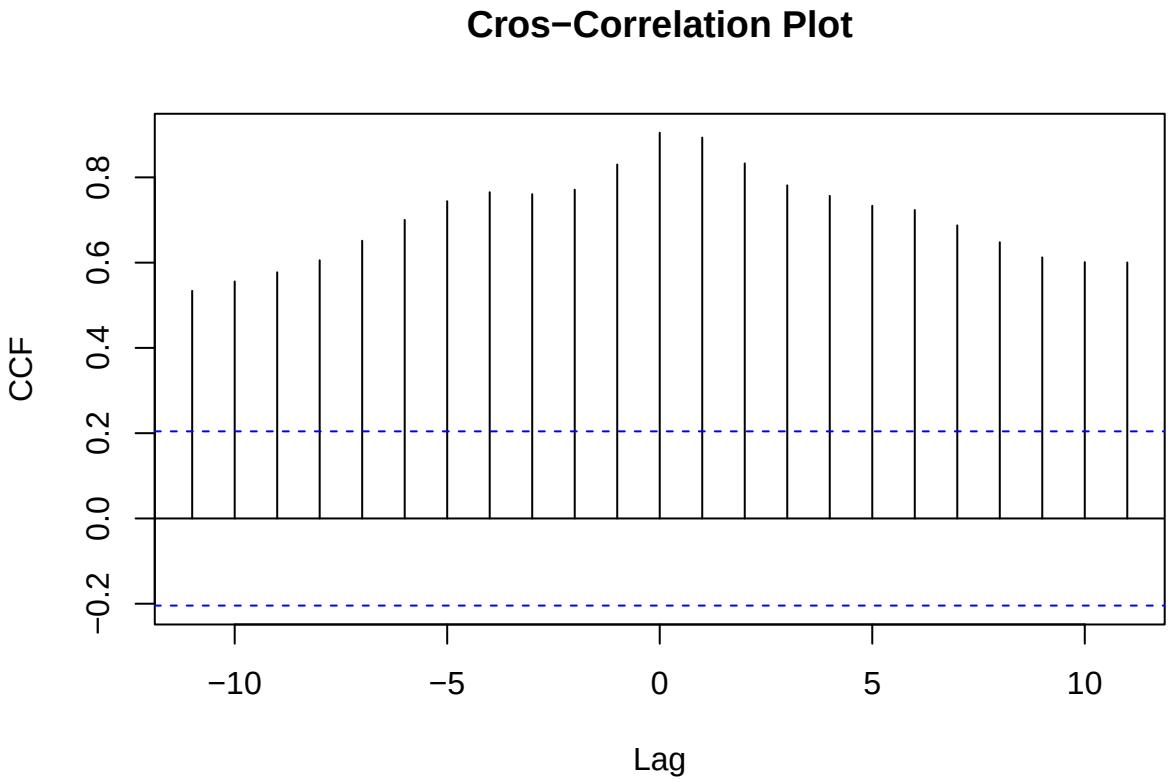

```
unique(daily_df_summer$yr)
```

```
##  [1] 2012 2013 2014 2016 2017 2018 2019 2020 2021 2022
```

```
yr=2022
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 = "EF OF SF SALMON RIVER AT STIBNITE, ID",
        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