Evaluation_6

Christiane

2019-04-19

Production de bananes polaires L'Association québécoise des producteurs de bananes polaires (AQPBP) a recensé la productivité de chacun des sites (en Mg par hectares) depuis plusieurs années. Des indicateurs météo ont été consignés sur la durée des saisons de croissance. Les médianes des indicateurs et des rendements compilés au cours des années se trouvent dans le tableau banane.csv. 1. L'AQPBP vous mandate de créer un modèle de prédiction du rendement en fonction de la météo et de la position. Créez un modèle d'autoapprentissage avec entraînement et test avec la méthode de votre choix. Par exemple, comme ceci. id_tr <- createDataPartition(banane$yield, p = 0.7, list = FALSE) banane_tr <- banane[id_tr,] banane_te <- banane[-id_tr,] ml_mod <- train(yield ~ lon + lat + degree_days + cumul_precip + sdi, data = banane_tr, method = "celle-que-vous-desirez") # mettre à l'échelle si nécessaire Le modèle est-il fiable? 2. Un organisme responsable des prévisions climatiques s'attend à ce que, d'ici 2050, les degrés-jours augmentent de 15%, les précipitations totales augmentent de 8% et que l'indice sdi diminue de 12% de manière uniforme sur tout le territoire de culture au Québec. L'AQPBP vous demande ce qui adviendra de la production totale de bananes polaires au Québec. Calculez d'abord la production totale pour tous les sites (somme des rendements), sachant qu'il y a 6000 hectares en culture (multiplier la productivité en Mg/ha par 6000 ha). Prenez le tableau initial et multipliez la colonne degrés-jour par 1.15, la colonne des précipitations par 1.08 et la colonne des sdi par 0.88. Puis utilisez la fonction predict pour prédire le rendement à l'aide du modèle que vous avez créé en (1). Enfin, effectuez la somme des rendements prédits (en multipliant par 6000 ha). Y aurait-il augmentation ou diminution des rendements? *Une augmentaion, si mes calculs sont corrects.* 3. Enfin, l'AQPBP aimerait avoir une carte de prédiction spatiale des précipitations cumulées actuelles (cumul_precip). Utilisez les codes fournis dans le chapitre sur les données géospatiales pour effectuer une prédiction spatiale. Utilisez la méthode de calcul que vous désirez (processus gaussien ou autre). Ne tentez pas de créer le meilleur modèle possible en essayant plusieurs algorithmes: créer le vôtre, commentez-le et donnez des pistes pour l'amélioration. Créez une grille de points sur lesquels les précipitations seront prédites (pas nécessairement avec sf, expand.grid suffira), puis créez une carte avec ggmap. Soumettez votre rapport dans un PDF généré depuis un fichier Rmd, accompagné du zip de vos calculs et de vos données (output: pdf_document)

Evaluation6_Christiane

```
library(sf)
```

```
## Warning: package 'sf' was built under R version 3.5.3
```

```
## Linking to GEOS 3.6.1, GDAL 2.2.3, PROJ 4.9.3
```

```
library(ggmap)
```

```
## Warning: package 'ggmap' was built under R version 3.5.3
```

```
## Loading required package: ggplot2
```

```
## Google's Terms of Service: https://cloud.google.com/maps-platform/terms/.
```

```
## Please cite ggmap if you use it! See citation("ggmap") for details.
```

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.5.3
```

```
## Loading required package: lattice
```

```
library(kknn)
```

```
## Warning: package 'kknn' was built under R version 3.5.3
```

```
##
## Attaching package: 'kknn'
```

```
## The following object is masked from 'package:caret':
##
##     contr.dummy
```

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 3.5.3
```

```
## -- Attaching packages ---------------------------------------------- tidyverse 1.2.1 --
```

```
## v tibble  2.1.1       v purrr   0.3.2
## v tidyr   0.8.3       v dplyr   0.8.0.1
## v readr   1.3.1       v stringr 1.3.1
## v tibble  2.1.1       v forcats 0.3.0
```

```
## Warning: package 'tibble' was built under R version 3.5.3
```

```
## Warning: package 'tidyr' was built under R version 3.5.3
```

```
## Warning: package 'readr' was built under R version 3.5.3
```

```
## Warning: package 'purrr' was built under R version 3.5.3
```

```
## Warning: package 'dplyr' was built under R version 3.5.3
```

```
## -- Conflicts ------------------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x purrr::lift()   masks caret::lift()
```

```
library(tidyr)
library(readr)
library(ggplot2)
library(plyr)
```

```
## ------------------------------------------------------------------------

## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)

## ------------------------------------------------------------------------

##
## Attaching package: 'plyr'

## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize

## The following object is masked from 'package:purrr':
##
##     compact
```

```
library(dplyr)
library(lattice)
library(compositions)
```

```
## Warning: package 'compositions' was built under R version 3.5.3

## Loading required package: tensorA

## Warning: package 'tensorA' was built under R version 3.5.2

##
## Attaching package: 'tensorA'

## The following object is masked from 'package:base':
##
##     norm

## Loading required package: robustbase

## Warning: package 'robustbase' was built under R version 3.5.3

## Loading required package: energy

## Warning: package 'energy' was built under R version 3.5.3

## Loading required package: bayesm

## Warning: package 'bayesm' was built under R version 3.5.3
```

```
## Welcome to compositions, a package for compositional data analysis.
## Find an intro with "? compositions"

##
## Attaching package: 'compositions'

## The following object is masked from 'package:caret':
##
##     R2

## The following objects are masked from 'package:stats':
##
##     cor, cov, dist, var

## The following objects are masked from 'package:base':
##
##     %*%, scale, scale.default
```

```
library(agridat)
```

```
## Warning: package 'agridat' was built under R version 3.5.3
```

```
library(expsmooth)
```

```
## Warning: package 'expsmooth' was built under R version 3.5.3

## Loading required package: forecast

## Warning: package 'forecast' was built under R version 3.5.3
```

```
library(forecast)
library(tinytex)
```

```
## Warning: package 'tinytex' was built under R version 3.5.3
```

```
setwd("C:/Users/Utilisateur/Dropbox/Analyse")
banane <- read_csv("CodeR/_Evaluation6/banane.csv")
```

```
## Parsed with column specification:
## cols(
##   station_name = col_character(),
##   lat = col_double(),
##   lon = col_double(),
##   degree_days = col_double(),
##   cumul_precip = col_double(),
##   sdi = col_double(),
##   yield = col_double()
## )
```

```r
bananee <- banane %>% dplyr::select(-station_name)
id_tr <- createDataPartition(bananee$yield, p = 0.7, list = FALSE)
banane_tr <- bananee[id_tr, ]
banane_te <- bananee[-id_tr, ]
ml_mod <- caret::train(yield ~ lon + lat + degree_days + cumul_precip + sdi,
                data = banane_tr,
                method = "kknn")

kknn_grid <- expand.grid(kmax = 3:6,
          distance = 1:2,
          kernel = c("rectangular", "gaussian", "optimal"))

ctrl <- trainControl(method="repeatedcv", repeats = 5)

#set.seed()
clf  <- train(yield ~ .,
              data = banane_tr,
              method = "kknn",
              tuneGrid = kknn_grid,
              trainControl=ctrl)
clf
```

```
## k-Nearest Neighbors
##
## 51 samples
##  5 predictor
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 51, 51, 51, 51, 51, 51, ...
## Resampling results across tuning parameters:
##
##   kmax  distance  kernel       RMSE      Rsquared   MAE
##   3     1         rectangular  4.729135  0.5244242  3.887429
##   3     1         gaussian     4.688680  0.5284012  3.844924
##   3     1         optimal      4.655525  0.5365056  3.805873
##   3     2         rectangular  4.800787  0.5185499  3.963709
##   3     2         gaussian     4.628870  0.5360777  3.818887
##   3     2         optimal      4.643547  0.5313631  3.831968
##   4     1         rectangular  4.715064  0.5302638  3.869141
##   4     1         gaussian     4.667787  0.5321130  3.824722
##   4     1         optimal      4.629337  0.5397388  3.785186
##   4     2         rectangular  4.725599  0.5236509  3.903604
##   4     2         gaussian     4.613591  0.5368669  3.805451
##   4     2         optimal      4.627238  0.5336676  3.819884
##   5     1         rectangular  4.715064  0.5302638  3.869141
##   5     1         gaussian     4.666176  0.5317244  3.815779
##   5     1         optimal      4.623278  0.5406808  3.780180
##   5     2         rectangular  4.732441  0.5221853  3.912398
##   5     2         gaussian     4.619014  0.5358672  3.809448
##   5     2         optimal      4.621732  0.5348270  3.813748
##   6     1         rectangular  4.715064  0.5302638  3.869141
##   6     1         gaussian     4.666176  0.5317244  3.815779
```

```
##   6     1         optimal      4.621041  0.5411576  3.773952
##   6     2         rectangular  4.732441  0.5221853  3.912398
##   6     2         gaussian     4.628948  0.5340739  3.818617
##   6     2         optimal      4.619772  0.5358900  3.809575
##
## RMSE was used to select the optimal model using the smallest value.
## The final values used for the model were kmax = 4, distance = 2 and
##  kernel = gaussian.
```

```
pred_tr <- predict(clf)
pred_te <- predict(clf, newdata = banane_te)

tablepred <- table(banane_tr$yield, pred_tr)


yield_sum <- bananee %>%
  sum(bananee$yield)
 yield_total <- yield_sum*6000

 banane_mult <- (data = bananee +
  bananee$degree_days*1.15 +
  bananee$cumul_precip*1.08 +
  bananee$sdi*0.88)

 ml_yield <- caret::train(yield ~ lon + lat + degree_days + cumul_precip + sdi,
                       data = banane_mult,
                       method = "kknn")

 yield_predict_sum <- (data = banane_mult +
                       sum(banane_mult$yield))
yield_predict_total <- yield_predict_sum*6000


banane_geo <- bananee %>%
st_as_sf(coords = c("lon", "lat"), crs = 4326) %>%
st_transform("+proj=longlat +datumNAD83")


banane_geo %>% st_coordinates() %>% summary()
```

```
##        X                Y
##  Min.   :-79.10   Min.   :45.03
##  1st Qu.:-73.63   1st Qu.:46.15
##  Median :-71.69   Median :47.41
##  Mean   :-71.46   Mean   :47.37
##  3rd Qu.:-69.53   3rd Qu.:48.49
##  Max.   :-61.77   Max.   :49.84
```

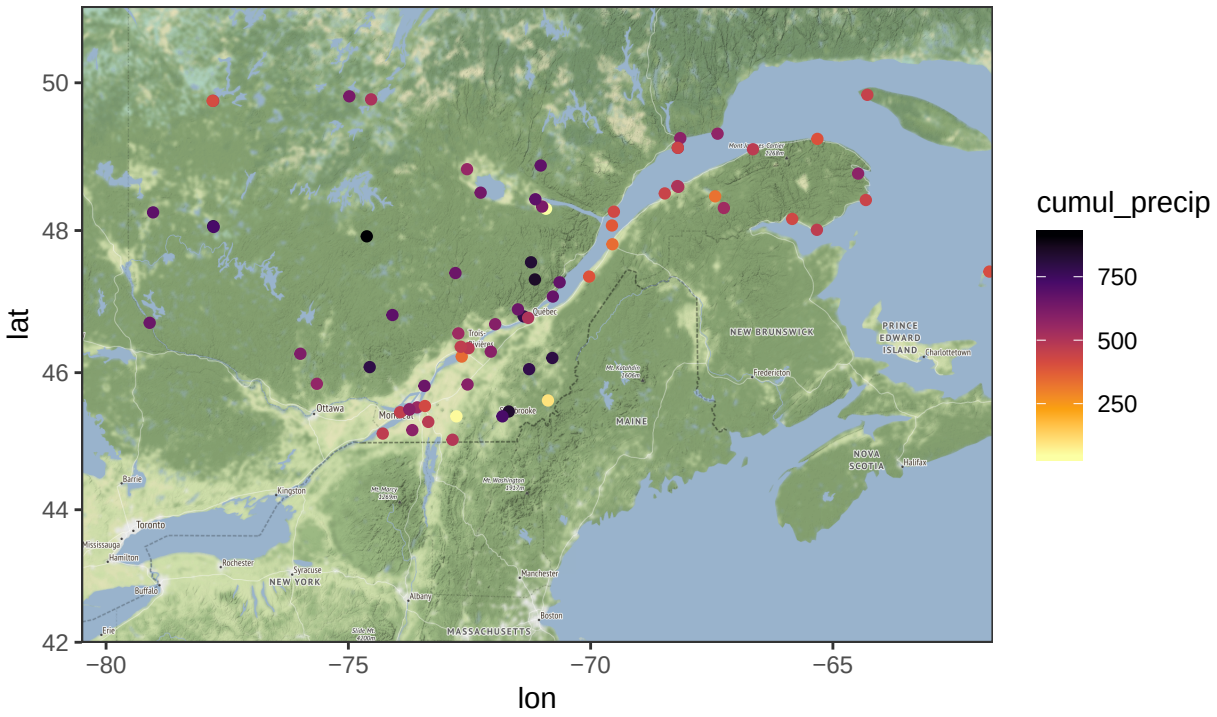
```
banane_map <- get_stamenmap(bbox=c(left = -80.50, right = -61.70, bottom = 42, top = 51),
              zoom=7, maptype="terrain")
```

```
## 48 tiles needed, this may take a while (try a smaller zoom).
```

```
## Source : http://tile.stamen.com/terrain/7/35/42.png

## Source : http://tile.stamen.com/terrain/7/36/42.png

## Source : http://tile.stamen.com/terrain/7/37/42.png

## Source : http://tile.stamen.com/terrain/7/38/42.png

## Source : http://tile.stamen.com/terrain/7/39/42.png

## Source : http://tile.stamen.com/terrain/7/40/42.png

## Source : http://tile.stamen.com/terrain/7/41/42.png

## Source : http://tile.stamen.com/terrain/7/42/42.png

## Source : http://tile.stamen.com/terrain/7/35/43.png

## Source : http://tile.stamen.com/terrain/7/36/43.png

## Source : http://tile.stamen.com/terrain/7/37/43.png

## Source : http://tile.stamen.com/terrain/7/38/43.png

## Source : http://tile.stamen.com/terrain/7/39/43.png

## Source : http://tile.stamen.com/terrain/7/40/43.png

## Source : http://tile.stamen.com/terrain/7/41/43.png

## Source : http://tile.stamen.com/terrain/7/42/43.png

## Source : http://tile.stamen.com/terrain/7/35/44.png

## Source : http://tile.stamen.com/terrain/7/36/44.png

## Source : http://tile.stamen.com/terrain/7/37/44.png

## Source : http://tile.stamen.com/terrain/7/38/44.png

## Source : http://tile.stamen.com/terrain/7/39/44.png

## Source : http://tile.stamen.com/terrain/7/40/44.png

## Source : http://tile.stamen.com/terrain/7/41/44.png

## Source : http://tile.stamen.com/terrain/7/42/44.png
```

```
## Source : http://tile.stamen.com/terrain/7/35/45.png
## Source : http://tile.stamen.com/terrain/7/36/45.png
## Source : http://tile.stamen.com/terrain/7/37/45.png
## Source : http://tile.stamen.com/terrain/7/38/45.png
## Source : http://tile.stamen.com/terrain/7/39/45.png
## Source : http://tile.stamen.com/terrain/7/40/45.png
## Source : http://tile.stamen.com/terrain/7/41/45.png
## Source : http://tile.stamen.com/terrain/7/42/45.png
## Source : http://tile.stamen.com/terrain/7/35/46.png
## Source : http://tile.stamen.com/terrain/7/36/46.png
## Source : http://tile.stamen.com/terrain/7/37/46.png
## Source : http://tile.stamen.com/terrain/7/38/46.png
## Source : http://tile.stamen.com/terrain/7/39/46.png
## Source : http://tile.stamen.com/terrain/7/40/46.png
## Source : http://tile.stamen.com/terrain/7/41/46.png
## Source : http://tile.stamen.com/terrain/7/42/46.png
## Source : http://tile.stamen.com/terrain/7/35/47.png
## Source : http://tile.stamen.com/terrain/7/36/47.png
## Source : http://tile.stamen.com/terrain/7/37/47.png
## Source : http://tile.stamen.com/terrain/7/38/47.png
## Source : http://tile.stamen.com/terrain/7/39/47.png
## Source : http://tile.stamen.com/terrain/7/40/47.png
## Source : http://tile.stamen.com/terrain/7/41/47.png
## Source : http://tile.stamen.com/terrain/7/42/47.png
```

```
banane_coord <- banane_geo %>%
  st_coordinates()%>%
  as_tibble() %>%
  mutate(cumul_precip = banane_geo$cumul_precip,
         degree_days = banane_geo$degree_days)

ggmap(banane_map) +
  geom_point(data = banane_coord, aes(x=X, y=Y, colour = cumul_precip)) +
  scale_colour_viridis_c(option = "inferno", direction = -1) +
  theme_bw()
```




```
library("caret")
banane_dd <- bananee %>%
  dplyr::select(lon, lat, cumul_precip) %>%
  drop_na()
banane_dd_sc <- banane_dd %>%
  mutate(cumul_precip = (cumul_precip - mean(cumul_precip))/sd(cumul_precip))
train_id_cumul <- createDataPartition(y = banane_dd_sc$cumul_precip, p = 0.7, list = FALSE)

library("kernlab")
```

```
## Warning: package 'kernlab' was built under R version 3.5.2
```

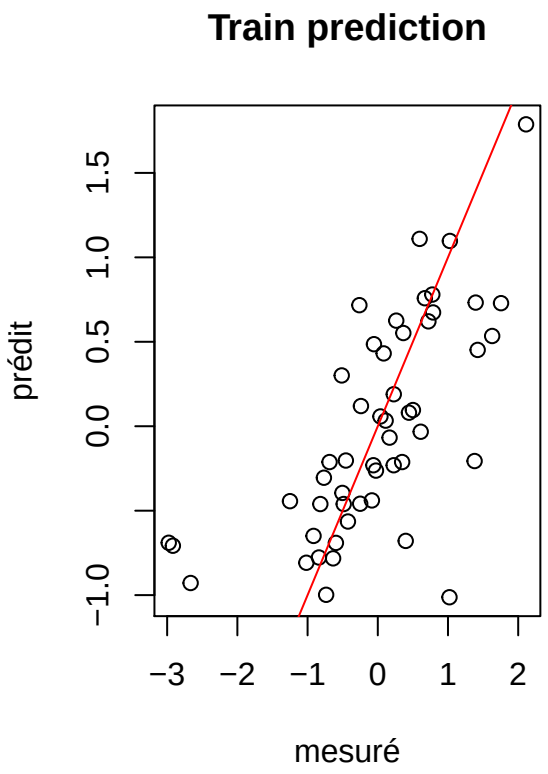
```
##
## Attaching package: 'kernlab'
```

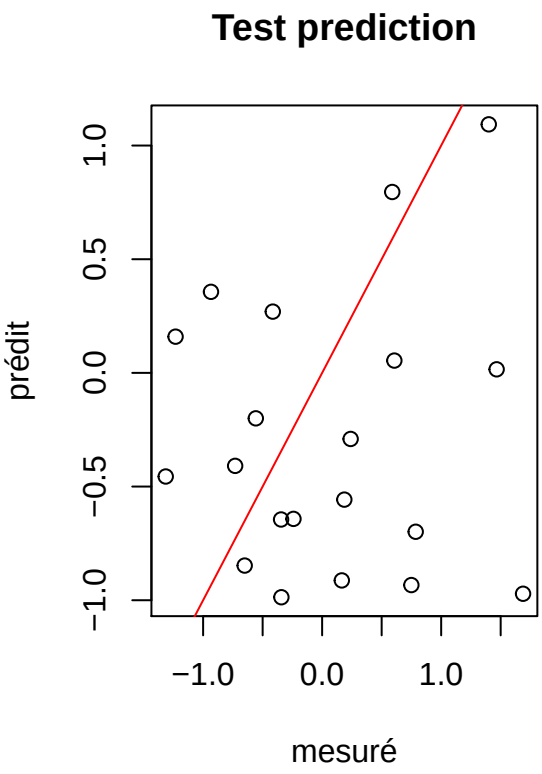
```
## The following object is masked from 'package:purrr':
##
##     cross

## The following object is masked from 'package:ggplot2':
##
##     alpha
```

```
dd_gp <- gausspr(x = banane_dd_sc[train_id_cumul, c("lon", "lat")],
                 y = banane_dd_sc[train_id_cumul, "cumul_precip"],
                 kernel = "rbfdot",
                 #kpar = list(sigma = 01), # laisser optimiser
                 variance.model = TRUE,
                 scale = TRUE,
                 var = 0.1,
                 cross = 5)
```

```
## Using automatic sigma estimation (sigest) for RBF or laplace kernel
```

```
pred_dd_tr <- predict(dd_gp)
pred_dd_te <- predict(dd_gp, newdata =  banane_dd_sc[-train_id_cumul, c("lon", "lat")])

par(mfrow = c(1, 2))
plot(banane_dd_sc$cumul_precip[train_id_cumul], pred_dd_tr, main = "Train prediction", xlab = "mesuré",
abline(0, 1, col="red")
plot(banane_dd_sc$cumul_precip[-train_id_cumul], pred_dd_te, main = "Test prediction", xlab = "mesuré",
abline(0, 1, col="red")
```






```
grid <- expand.grid(lon = seq(from = -80.50, to = -61.70, by = 0.25),
                    lat = seq(from = 42, to = 51, by = 0.25))

grid <- grid %>%
  mutate(pred_dd_mean = predict(dd_gp, newdata = ., type = "response") * sd(banane_dd$cumul_precip) + m
         pred_dd_sd = predict(dd_gp, newdata = ., type = "sdeviation") * sd(banane_dd$cumul_precip))
head(grid)
```

```
##      lon lat pred_dd_mean pred_dd_sd
## 1 -80.50  42     547.0786   174.2072
## 2 -80.25  42     546.9848   174.2069
## 3 -80.00  42     546.8749   174.2065
## 4 -79.75  42     546.7472   174.2060
## 5 -79.50  42     546.5997   174.2052
## 6 -79.25  42     546.4306   174.2042
```

```
grid_geo <- banane_dd_sc %>%
  st_as_sf(coords = c("lon", "lat"), crs = 4326) %>%
  st_transform("+proj=longlat +datumNAD83")

grid_geo %>% st_coordinates() %>% summary()
```

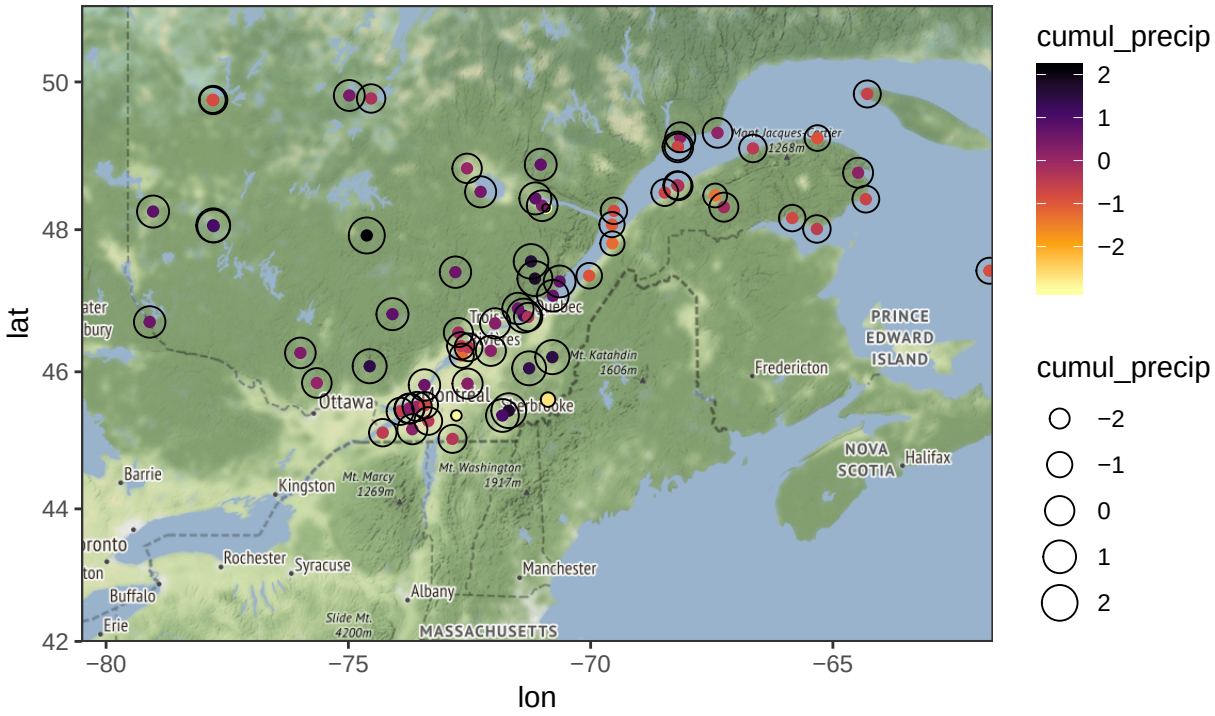
```
##        X                Y
##  Min.   :-79.10   Min.   :45.03
##  1st Qu.:-73.63   1st Qu.:46.15
```

```
##  Median :-71.69   Median :47.41
##  Mean   :-71.46   Mean   :47.37
##  3rd Qu.:-69.53   3rd Qu.:48.49
##  Max.   :-61.77   Max.   :49.84
```

```
grid_map <- get_stamenmap(bbox=c(left = -80.50, right = -61.70, bottom = 42, top = 51),
                          zoom=6, maptype="terrain")
```

```
## Source : http://tile.stamen.com/terrain/6/17/21.png

## Source : http://tile.stamen.com/terrain/6/18/21.png

## Source : http://tile.stamen.com/terrain/6/19/21.png

## Source : http://tile.stamen.com/terrain/6/20/21.png

## Source : http://tile.stamen.com/terrain/6/21/21.png

## Source : http://tile.stamen.com/terrain/6/17/22.png

## Source : http://tile.stamen.com/terrain/6/18/22.png

## Source : http://tile.stamen.com/terrain/6/19/22.png

## Source : http://tile.stamen.com/terrain/6/20/22.png

## Source : http://tile.stamen.com/terrain/6/21/22.png

## Source : http://tile.stamen.com/terrain/6/17/23.png

## Source : http://tile.stamen.com/terrain/6/18/23.png

## Source : http://tile.stamen.com/terrain/6/19/23.png

## Source : http://tile.stamen.com/terrain/6/20/23.png

## Source : http://tile.stamen.com/terrain/6/21/23.png
```

```
grid_coord <- grid_geo %>%
  st_coordinates()%>%
  as_tibble() %>%
  mutate(cumul_precip = banane_dd_sc$cumul_precip)

grid_nogeo <- grid_geo %>%
st_set_geometry(NULL)
grid_pred <- bind_cols(grid_nogeo, grid_coord)

ggmap(grid_map) +
  geom_point(data= grid_pred , aes(x=X, y=Y, colour = cumul_precip)) +
  geom_point(data = grid_coord, aes(x=X, y=Y, size = cumul_precip), shape = 21) +
  scale_colour_viridis_c(option = "inferno", direction = -1) +
  theme_bw()
```


lat
lon
50
48
46
44
42
−80
−75
−70
−65
Ottawa
Montreal
Sherbrooke
Fredericton
PRINCE
EDWARD
ISLAND
NOVA
SCOTIA
Halifax
Barrie
Kingston
Rochester
Syracuse
Buffalo
Erie
Albany
Manchester
MASSACHUSETTS
Mt. Katahdin
1606m
Mt. Marcy
1269m
Mt. Washington
1917m
Slide Mt.
4200m
cumul_precip
2
1
0
−1
−2
cumul_precip
−2
−1
0
1
2