ML interpolation

Amber Lee

6/22/2021

Load libraries

```
library(tidyverse)
library(stringr)
library(lubridate)

library(rpart) # for regression tree
library(rpart.plot)
library(caret) # for other models
library(rattle)
library(kableExtra)
library(broom)
```

Read data

```
water20 <- read.csv("cleaned_data.csv", header = TRUE)
```

Very brief data exploration

Removed the QF code variable

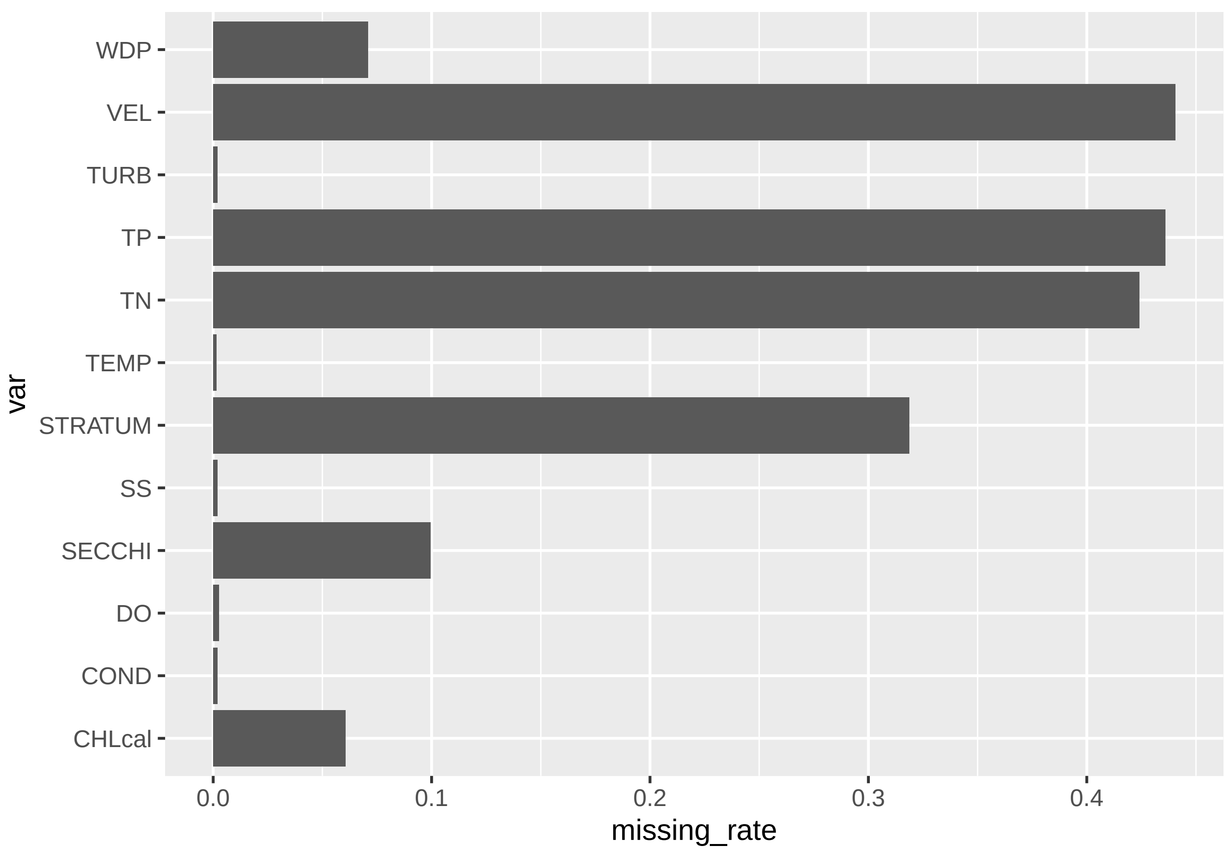
```
# remove the QF code variables
water20 <- water20 %>% select(all_of(names(water20)
                                     [str_detect(names(water20),
                                                 "QF", negate = T)]))
```

Checked the missingness rate per variable

```
var_missing_rate <- sapply(water20, function(x) sum(is.na(x))/length(x))

var_missing_rate <- data.frame(var = names(var_missing_rate),
                               missing_rate = unname(var_missing_rate))

var_missing_rate %>%
  mutate(`missing rate` = round(missing_rate, digits = 3)) %>%
  select(-missing_rate) %>%
  arrange(-`missing rate`) %>%
  kbl(booktabs = T)
```

var	missing rate
VEL	0.440
TP	0.436
TN	0.424
STRATUM	0.319
SECCHI	0.100
WDP	0.071
CHLcal	0.061
DO	0.003
TEMP	0.002
TURB	0.002
COND	0.002
SS	0.002
SHEETBAR	0.000
DATE	0.000
LATITUDE	0.000
LONGITUDE	0.000
FLDNUM	0.000
LOCATCD	0.000

```r
var_missing_rate %>% filter(missing_rate > 0) %>%
  ggplot(aes(x = var, y = missing_rate)) +
  geom_bar(stat = "identity") +
  coord_flip()
```



Added year and season variables, changed the `FLDNUM` variable to be categorical (character).

** I filtered out observations where `TP` was greater than 2 because that represented .1% of the data **

```
water20 <- water20 %>%
  filter(TP <= 2) %>% # only 70 samples for TP > 2, outlier values screw up model
  mutate(nice_date = mdy(DATE),
         year = year(nice_date),
         quarter = quarter(nice_date, fiscal_start = 3),
         FLDNUM = case_when(FLDNUM == 1 ~ "Lake City, MN",
                            FLDNUM == 2 ~ "Onalaska, WI",
                            FLDNUM == 3 ~ "Bellevue, IA",
                            FLDNUM == 4 ~ "Brighton, IL",
                            FLDNUM == 5 ~ "Jackson, MO",
                            FLDNUM == 6 ~ "Havana, IL"),
         FLDNUM = as.factor(FLDNUM),
         STRATUM = as.factor(STRATUM)) %>% # CART can split on categorical variable if
                                           # encoded as a factor variable
  select(-SHEETBAR, -nice_date, -DATE, -LOCATCD)
```

Notes

- regression (what are the steps for it?)
- classification and regression trees (CART)
- neural networks
- support vector machines

predicting TP with entire dataset

80/20

```
set.seed(4747)

# train and test data created from rows with existing TP values
fullTP <- water20 %>% filter(!is.na(TP))

# this gives me a total of ten 80/20 splits
train_idx <- createDataPartition(fullTP$TP, p = .8,
                                 list = FALSE, times = 10)

watertrain1 <- fullTP[train_idx[,1],]
watertest1 <- fullTP[-train_idx[,1],]
```

There are:

- 105,000 samples in the filtered water dataset `water20`
- 59,000 samples in the training data `fullTP`. This is because the testing and training data both need existing `TP` values
- 47,000 samples in the `watertrain1`
- and 11,800 samples in the `watertest1`

using the rpart library

https://www.statmethods.net/advstats/cart.html

```r
tr.TP <- rpart(TP ~ .,
               data = watertrain1,
               control = rpart.control(xval = 15,
                                       minsplit = 100))

printcp(tr.TP)
```

```
##
## Regression tree:
## rpart(formula = TP ~ ., data = watertrain1, control = rpart.control(xval = 15,
##     minsplit = 100))
##
## Variables actually used in tree construction:
## [1] COND SS   TEMP TURB
##
## Root node error: 1252.8/47394 = 0.026434
##
## n= 47394
##
##         CP nsplit rel error  xerror     xstd
## 1 0.290222      0   1.00000 1.00003 0.022199
## 2 0.080946      1   0.70978 0.71062 0.019402
## 3 0.040385      2   0.62883 0.62947 0.017458
## 4 0.027189      3   0.58845 0.59008 0.016191
## 5 0.023035      4   0.56126 0.56303 0.015791
## 6 0.021124      5   0.53822 0.53991 0.015655
## 7 0.016767      6   0.51710 0.51915 0.015577
## 8 0.010873      7   0.50033 0.50647 0.015435
## 9 0.010000      8   0.48946 0.50134 0.015413
```

```r
png("tree_TP.png")
fancyRpartPlot(tr.TP)
dev.off()
```

```
## pdf
##   2
```

```r
png("tree_TP_CV.png")
plotcp(tr.TP)
dev.off()
```

```
## pdf
##   2
```

```r
summary(tr.TP)
```

```
## Call:
## rpart(formula = TP ~ ., data = watertrain1, control = rpart.control(xval = 15,
##     minsplit = 100))
##   n= 47394
##
##           CP nsplit rel error    xerror       xstd
## 1 0.29022215      0 1.0000000 1.0000321 0.02219880
```

```
## 2 0.08094634      1 0.7097778 0.7106159 0.01940201
## 3 0.04038529      2 0.6288315 0.6294724 0.01745763
## 4 0.02718854      3 0.5884462 0.5900796 0.01619053
## 5 0.02303451      4 0.5612577 0.5630315 0.01579095
## 6 0.02112450      5 0.5382232 0.5399098 0.01565544
## 7 0.01676703      6 0.5170987 0.5191532 0.01557656
## 8 0.01087301      7 0.5003316 0.5064659 0.01543478
## 9 0.01000000      8 0.4894586 0.5013362 0.01541311
##
## Variable importance
##      TURB        SS    FLDNUM  LATITUDE LONGITUDE      COND    SECCHI      TEMP
##        39        27         7         7         6         6         6         1
##
## Node number 1: 47394 observations,    complexity param=0.2902222
##   mean=0.1964352, MSE=0.02643391
##   left son=2 (33800 obs) right son=3 (13594 obs)
##   Primary splits:
##       TURB     < 38.5      to the left,  improve=0.2895250, (105 missing)
##       SECCHI   < 28.5      to the right, improve=0.2309235, (7909 missing)
##       SS       < 37.39     to the left,  improve=0.2290521, (159 missing)
##       LATITUDE < 41.11549  to the right, improve=0.1822682, (0 missing)
##       FLDNUM   splits as  LRRRLL, improve=0.1808393, (557 missing)
##   Surrogate splits:
##       SS        < 46.805    to the left,  agree=0.920, adj=0.721, (101 split)
##       LATITUDE  < 40.32079  to the right, agree=0.786, adj=0.251, (4 split)
##       FLDNUM    splits as  LLRRLL, agree=0.782, adj=0.241, (0 split)
##       LONGITUDE < -89.72545 to the left,  agree=0.778, adj=0.224, (0 split)
##       SECCHI    < 28.5      to the right, agree=0.773, adj=0.208, (0 split)
##
## Node number 2: 33800 observations,    complexity param=0.04038529
##   mean=0.1408882, MSE=0.01081529
##   left son=4 (32363 obs) right son=5 (1437 obs)
##   Primary splits:
##       COND     < 738.5     to the left,  improve=0.13838440, (114 missing)
##       TURB     < 18.5      to the left,  improve=0.10829110, (60 missing)
##       FLDNUM   splits as  LRRLLL, improve=0.09726929, (532 missing)
##       LATITUDE < 41.77794  to the right, improve=0.08999491, (0 missing)
##       SECCHI   < 47.5      to the right, improve=0.08849416, (6260 missing)
##
## Node number 3: 13594 observations,    complexity param=0.08094634
##   mean=0.334547, MSE=0.03852133
##   left son=6 (12888 obs) right son=7 (706 obs)
##   Primary splits:
##       TURB     < 307       to the left,  improve=0.19146020, (45 missing)
##       SS       < 584.65    to the left,  improve=0.17680420, (54 missing)
##       SECCHI   < 11.5      to the right, improve=0.09771220, (1649 missing)
##       FLDNUM   splits as  LLRLLR, improve=0.05577148, (25 missing)
##       LATITUDE < 38.94276  to the left,  improve=0.04493640, (0 missing)
##   Surrogate splits:
##       SS < 410.75    to the left,  agree=0.975, adj=0.519, (41 split)
##
## Node number 4: 32363 observations,    complexity param=0.0211245
##   mean=0.1327355, MSE=0.00667315
##   left son=8 (22056 obs) right son=9 (10307 obs)
```

```
##   Primary splits:
##       TURB   < 17.5      to the left,  improve=0.12253150, (56 missing)
##       SECCHI < 47.5      to the right, improve=0.10193660, (5957 missing)
##       TEMP   < 21.35     to the left,  improve=0.09280021, (80 missing)
##       SS     < 14.465    to the left,  improve=0.09034200, (101 missing)
##       DO     < 8.45      to the right, improve=0.08267267, (151 missing)
##   Surrogate splits:
##       SS        < 21.81     to the left,  agree=0.892, adj=0.662, (56 split)
##       LATITUDE  < 41.74147  to the right, agree=0.756, adj=0.233, (0 split)
##       FLDNUM    splits as  LRRRLL, agree=0.743, adj=0.193, (0 split)
##       LONGITUDE < -90.11358 to the left,  agree=0.714, adj=0.103, (0 split)
##       SECCHI    < 47.5      to the right, agree=0.709, adj=0.085, (0 split)
##
## Node number 5: 1437 observations
##   mean=0.3244962, MSE=0.0688925
##
## Node number 6: 12888 observations,    complexity param=0.02718854
##   mean=0.3143319, MSE=0.02699046
##   left son=12 (10334 obs) right son=13 (2554 obs)
##   Primary splits:
##       COND     < 667.5     to the left,  improve=0.09719369, (23 missing)
##       FLDNUM   splits as  LLRLLL, improve=0.08934998, (23 missing)
##       SECCHI   < 16.5      to the right, improve=0.06942041, (1493 missing)
##       TURB     < 101.5     to the left,  improve=0.06591051, (44 missing)
##       LATITUDE < 38.94067  to the left,  improve=0.06457725, (0 missing)
##   Surrogate splits:
##       FLDNUM splits as  LLRLLL, agree=0.848, adj=0.231, (20 split)
##
## Node number 7: 706 observations,    complexity param=0.01676703
##   mean=0.7035737, MSE=0.1053764
##   left son=14 (563 obs) right son=15 (143 obs)
##   Primary splits:
##       SS        < 797.4     to the left,  improve=0.2868661, (6 missing)
##       TURB      < 571       to the left,  improve=0.2838516, (1 missing)
##       LATITUDE  < 39.98717  to the left,  improve=0.2068708, (0 missing)
##       LONGITUDE < -89.52492 to the right, improve=0.2059435, (0 missing)
##       FLDNUM    splits as  RRRLRR, improve=0.1947993, (2 missing)
##   Surrogate splits:
##       TURB      < 710       to the left,  agree=0.903, adj=0.521, (6 split)
##       LATITUDE  < 44.55795  to the left,  agree=0.800, adj=0.014, (0 split)
##       LONGITUDE < -92.45883 to the right, agree=0.800, adj=0.014, (0 split)
##
## Node number 8: 22056 observations,    complexity param=0.01087301
##   mean=0.113187, MSE=0.004892865
##   left son=16 (15762 obs) right son=17 (6294 obs)
##   Primary splits:
##       TEMP    < 20.35     to the left,  improve=0.12668110, (54 missing)
##       DO      < 8.45      to the right, improve=0.11464010, (109 missing)
##       quarter < 3.5       to the right, improve=0.06091910, (0 missing)
##       TURB    < 6.5       to the left,  improve=0.05681586, (37 missing)
##       SS      < 5.09      to the left,  improve=0.05059193, (70 missing)
##   Surrogate splits:
##       DO       < 8.45      to the right, agree=0.809, adj=0.332, (1 split)
##       LATITUDE < 37.32416  to the right, agree=0.716, adj=0.006, (53 split)
```

```
##       TN       < 0.4125    to the right, agree=0.714, adj=0.000, (0 split)
##
## Node number 9: 10307 observations
##   mean=0.1745675, MSE=0.007915123
##
## Node number 12: 10334 observations,    complexity param=0.02303451
##   mean=0.2887743, MSE=0.01979563
##   left son=24 (7155 obs) right son=25 (3179 obs)
##   Primary splits:
##       TURB    < 101.5     to the left,  improve=0.14081750, (41 missing)
##       SECCHI  < 16.5      to the right, improve=0.12720390, (1335 missing)
##       SS      < 161.15    to the left,  improve=0.09944958, (34 missing)
##       CHLcal  < 85.60527  to the left,  improve=0.06410087, (901 missing)
##       STRATUM splits as  LLRRLRL, improve=0.04379174, (5219 missing)
##   Surrogate splits:
##       SS        < 141.45    to the left,  agree=0.891, adj=0.645, (38 split)
##       SECCHI    < 15.5      to the right, agree=0.754, adj=0.203, (3 split)
##       LATITUDE  < 37.03289  to the right, agree=0.692, adj=0.003, (0 split)
##       LONGITUDE < -89.36861 to the left,  agree=0.692, adj=0.003, (0 split)
##
## Node number 13: 2554 observations
##   mean=0.4177428, MSE=0.04276542
##
## Node number 14: 563 observations
##   mean=0.6166412, MSE=0.06636302
##
## Node number 15: 143 observations
##   mean=1.045832, MSE=0.1120805
##
## Node number 16: 15762 observations
##   mean=0.09748293, MSE=0.00339507
##
## Node number 17: 6294 observations
##   mean=0.1525145, MSE=0.006479527
##
## Node number 24: 7155 observations
##   mean=0.2535504, MSE=0.01510136
##
## Node number 25: 3179 observations
##   mean=0.3680532, MSE=0.02128343
```

```
# tidy(tr.TP) TODO
```

https://rstudio-pubs-static.s3.amazonaws.com/27179_e64f0de316fc4f169d6ca300f18ee2aa.html

```
# Prediction error rate in training data = Root node error * rel error * 100%
# Prediction error rate in cross-validation = Root node error * xerror * 100%
# Hence we want the cp value (with a simpler tree) that minimizes the xerror.

#

bestcp <- tr.TP$cptable[which.min(tr.TP$cptable[,"xerror"]),"CP"]
bestcp
```

```
## [1] 0.01
```

```r
tr.TP.prune <- prune(tr.TP, cp = bestcp)

data.frame(round(tr.TP.prune$cptable, digits = 3)) %>%
  kbl(booktabs = T)
```

CP	nsplit	rel.error	xerror	xstd
0.290	0	1.000	1.000	0.022
0.081	1	0.710	0.711	0.019
0.040	2	0.629	0.629	0.017
0.027	3	0.588	0.590	0.016
0.023	4	0.561	0.563	0.016
0.021	5	0.538	0.540	0.016
0.017	6	0.517	0.519	0.016
0.011	7	0.500	0.506	0.015
0.010	8	0.489	0.501	0.015

https://datascience.stackexchange.com/questions/31346/caret-and-rpart-does-caret-automatically-prune-rpart-trees

http://www.rdatamining.com/docs/regression-and-classification-with-r

Model evaluation

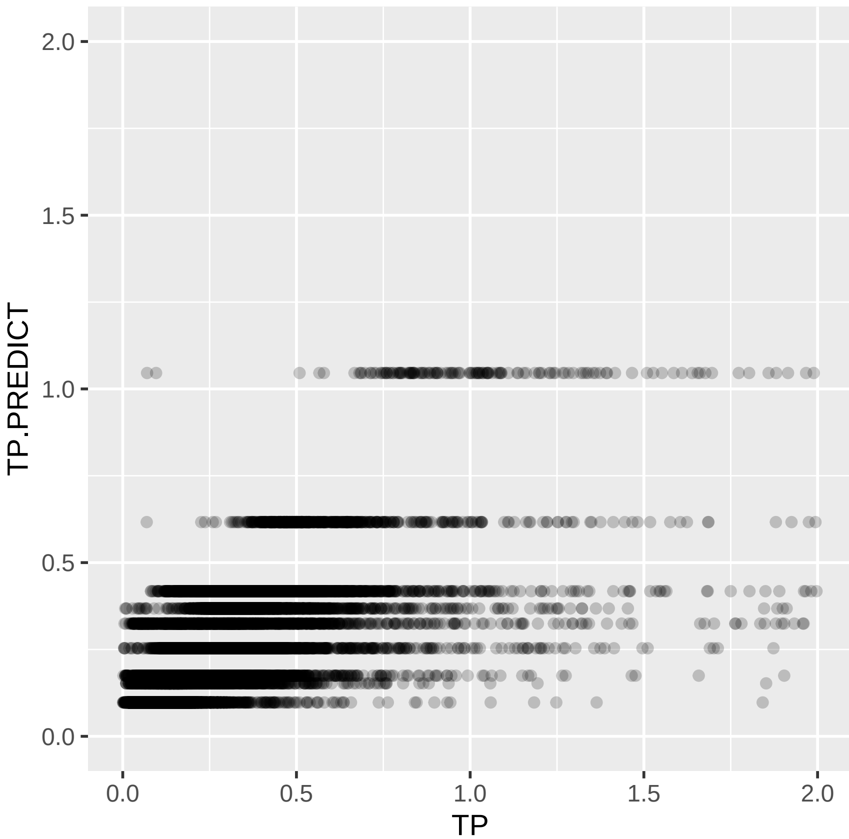
```r
watertrain1$TP.PREDICT <- predict(tr.TP, data = watertrain1)

watertrain1 <- watertrain1 %>%
  mutate(TP.SQ.ERROR = (TP - TP.PREDICT)^2)

print("RSME is")
```

```
## [1] "RSME is"
```

```r
sqrt(sum(watertrain1$TP.SQ.ERROR)/dim(watertrain1)[1])
```

```
## [1] 0.1137467
```

```r
watertrain1 %>%
  ggplot(aes(x = TP, y = TP.PREDICT)) +
  geom_point(alpha = 0.2) +
  coord_equal() +
  theme(aspect.ratio = 1) + xlim(0, 2) + ylim(0, 2)
```




```
ggsave("tree_training_predictvsactual.png")
```

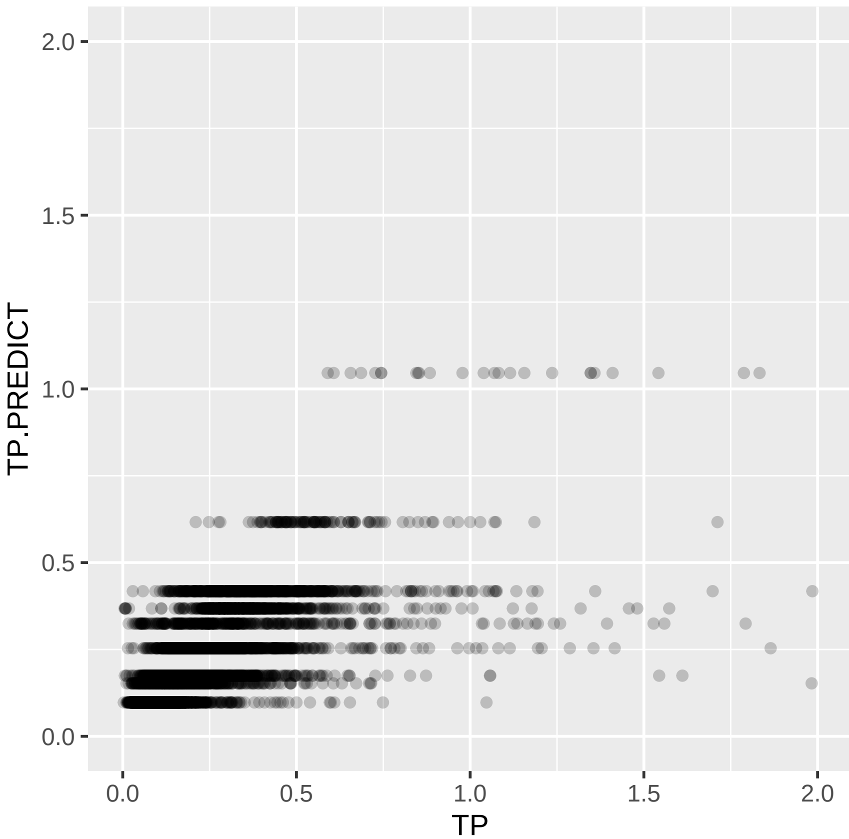
```
watertest1$TP.PREDICT <- predict(tr.TP.prune, watertest1)

watertest1 <- watertest1 %>%
  mutate(TP.SQ.ERROR = (TP - TP.PREDICT)^2)

print("RSME is")
```

```
## [1] "RSME is"
```

```
sqrt(sum(watertest1$TP.SQ.ERROR)/dim(watertest1)[1])
```

```
## [1] 0.1120495
```

```
watertest1 %>%
  ggplot(aes(x = TP, y = TP.PREDICT)) +
  geom_point(alpha = 0.2) +
  coord_equal() +
  theme(aspect.ratio = 1) + xlim(0, 2) + ylim(0, 2)
```




```r
ggsave("tree_test_predictvsactual.png")
```

```r
watertrain2 <- fullTP[train_idx[,2],]
watertest2 <- fullTP[-train_idx[,2],]

# remove all na's because that's how GLM works
watertrain2 <- watertrain2 %>%
  filter_all(all_vars(!is.na(.)))
watertest2 <- watertest2 %>%
  filter_all(all_vars(!is.na(.)))

linreg <- glm(TP ~ ., data = watertrain2)

summary(linreg)
```

multivariate linear regression

```
##
## Call:
## glm(formula = TP ~ ., data = watertrain2)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -0.66410  -0.03954  -0.00907   0.02492   1.48913
##
## Coefficients:
```

```
##                        Estimate Std. Error t value Pr(>|t|)    
## (Intercept)           2.784e+00  6.813e-01   4.086 4.41e-05 ***
## LATITUDE             -8.436e-03  9.145e-03  -0.922  0.35629    
## LONGITUDE             2.661e-02  6.501e-03   4.092 4.29e-05 ***
## FLDNUMBrighton, IL   -1.447e-02  2.925e-02  -0.495  0.62090    
## FLDNUMHavana, IL      3.452e-02  1.811e-02   1.906  0.05663 .  
## FLDNUMJackson, MO    -9.317e-02  4.357e-02  -2.139  0.03249 *  
## FLDNUMLake City, MN   6.572e-02  2.331e-02   2.819  0.00483 ** 
## FLDNUMOnalaska, WI    4.499e-02  1.587e-02   2.834  0.00460 ** 
## STRATUM2              1.404e-03  3.816e-03   0.368  0.71299    
## STRATUM3              3.993e-03  4.072e-03   0.981  0.32676    
## STRATUM4              1.613e-01  7.570e-03  21.306  < 2e-16 ***
## STRATUM5             -8.705e-03  4.544e-03  -1.916  0.05544 .  
## STRATUM6              6.495e-02  8.190e-03   7.930 2.35e-15 ***
## STRATUM9             -1.816e-02  2.827e-02  -0.642  0.52071    
## TN                    4.787e-04  5.116e-04   0.936  0.34947    
## TEMP                  2.061e-03  1.343e-04  15.353  < 2e-16 ***
## DO                   -8.912e-03  3.194e-04 -27.902  < 2e-16 ***
## TURB                  1.126e-03  4.109e-05  27.405  < 2e-16 ***
## COND                  2.027e-04  9.164e-06  22.116  < 2e-16 ***
## VEL                  -1.560e-03  2.682e-03  -0.582  0.56082    
## SS                   -2.453e-05  3.627e-05  -0.676  0.49891    
## WDP                  -2.772e-04  5.456e-04  -0.508  0.61140    
## CHLcal                1.047e-03  3.281e-05  31.899  < 2e-16 ***
## SECCHI               -3.912e-04  3.000e-05 -13.039  < 2e-16 ***
## year                  1.457e-05  1.125e-04   0.130  0.89691    
## quarter               1.672e-02  1.007e-03  16.599  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.009330335)
##
##     Null deviance: 320.12  on 14281  degrees of freedom
## Residual deviance: 133.01  on 14256  degrees of freedom
## AIC: -26202
##
## Number of Fisher Scoring iterations: 2
```

```r
tidy(linreg) %>%
  filter(p.value < 0.05) %>%
  mutate(estimate = round(estimate, digits = 3),
         p.value = round(p.value, digits = 3)) %>%
  select(term, estimate, p.value) %>%
  kbl(booktabs = T)
```

term	estimate	p.value
(Intercept)	2.784	0.000
LONGITUDE	0.027	0.000
FLDNUMJackson, MO	-0.093	0.032
FLDNUMLake City, MN	0.066	0.005
FLDNUMOnalaska, WI	0.045	0.005
STRATUM4	0.161	0.000
STRATUM6	0.065	0.000
TEMP	0.002	0.000
DO	-0.009	0.000
TURB	0.001	0.000
COND	0.000	0.000
CHLcal	0.001	0.000
SECCHI	0.000	0.000
quarter	0.017	0.000

```r
tidy(linreg) %>%
  filter(p.value >= 0.05) %>%
  mutate(estimate = round(estimate, digits = 3),
         p.value = round(p.value, digits = 3)) %>%
  select(term, estimate, p.value) %>%
  kbl(booktabs = T)
```

term	estimate	p.value
LATITUDE	-0.008	0.356
FLDNUMBrighton, IL	-0.014	0.621
FLDNUMHavana, IL	0.035	0.057
STRATUM2	0.001	0.713
STRATUM3	0.004	0.327
STRATUM5	-0.009	0.055
STRATUM9	-0.018	0.521
TN	0.000	0.349
VEL	-0.002	0.561
SS	0.000	0.499
WDP	0.000	0.611
year	0.000	0.897

```r
# watertrain2.selected <- fullTP[train_idx[,2],] %>%
#   select(all_of(c("TEMP", "DO", "TURB", "COND", "FLDNUM",
#                   "CHLcal", "quarter"))) %>%
#   filter_all(all_vars(!is.na(.)))
# TODO remove the missing values here and try to run the regression again, see if it applies to more va
```

```r
sum(fitted(linreg) == predict(linreg))
```

```
## [1] 14282
```

```r
watertrain2$TP.PREDICT <- predict(linreg)

watertrain2 <- watertrain2 %>%
  mutate(TP.SQ.ERROR = (TP - TP.PREDICT)^2)
```

```r
print("RSME is")
```

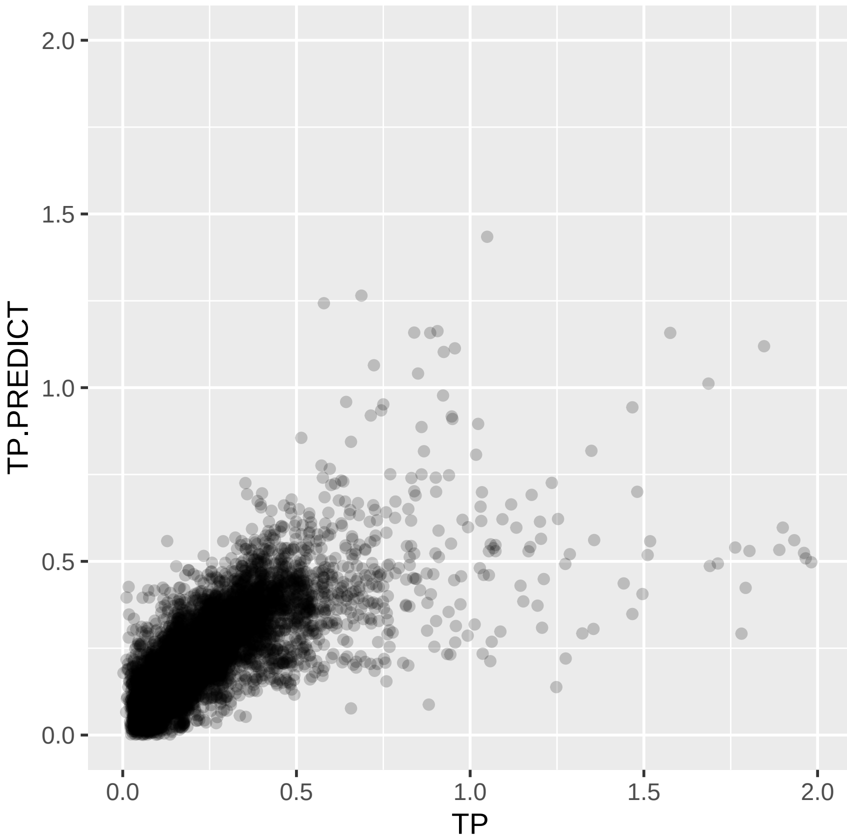
```
## [1] "RSME is"
```

```r
sqrt(sum(watertrain2$TP.SQ.ERROR)/dim(watertrain2)[1])
```

```
## [1] 0.0965057
```

```r
watertrain2 %>%
  ggplot(aes(x = TP, y = TP.PREDICT)) +
  geom_point(alpha = 0.2) +
  coord_equal() +
  theme(aspect.ratio = 1) + xlim(0, 2) + ylim(0, 2)
```

```
## Warning: Removed 32 rows containing missing values (geom_point).
```




```r
ggsave("glm_train_predictvsactual.png")
```

```
## Warning: Removed 32 rows containing missing values (geom_point).
```

```r
watertest2$TP.PREDICT <- predict(linreg, watertest2)

watertest2 <- watertest2 %>%
  mutate(TP.SQ.ERROR = (TP - TP.PREDICT)^2)

print("RSME is")
```

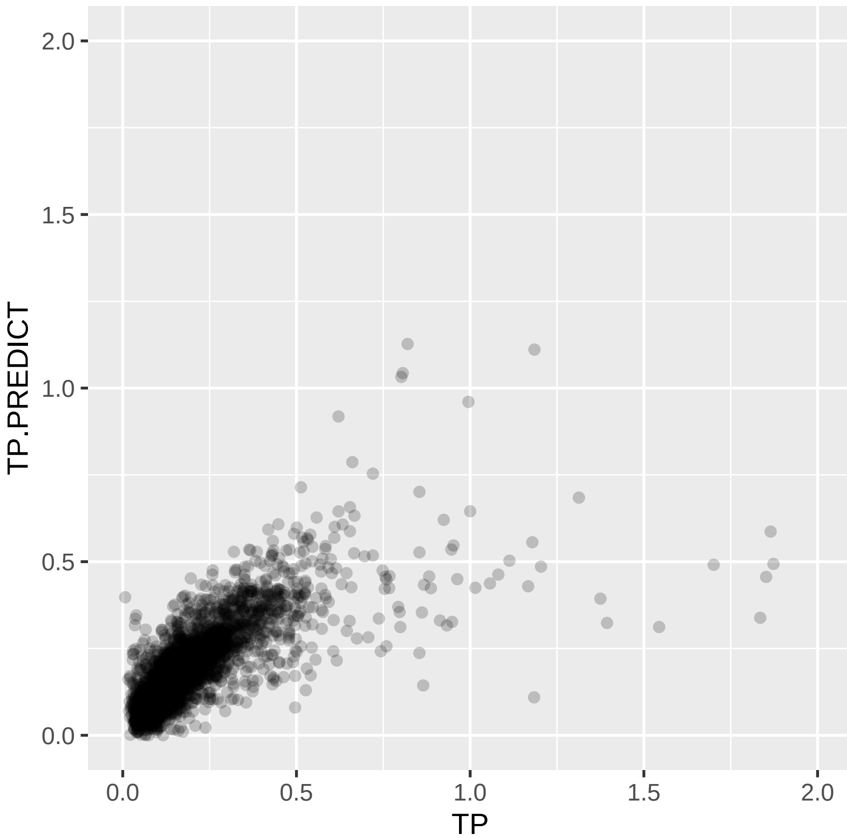
```
## [1] "RSME is"
```

```r
sqrt(sum(watertest2$TP.SQ.ERROR)/dim(watertest2)[1])
```

```
## [1] 0.1048744
```

```r
watertest2 %>%
  ggplot(aes(x = TP, y = TP.PREDICT)) +
  geom_point(alpha = 0.2) +
  coord_equal() +
  theme(aspect.ratio = 1) + xlim(0, 2) + ylim(0, 2)
```

```
## Warning: Removed 9 rows containing missing values (geom_point).
```




```r
ggsave("glm_test_predictvsactual.png")
```

```
## Warning: Removed 9 rows containing missing values (geom_point).
```

TP by year and season

```r
make_year_touples <- function(index, year_partition){
  # index goes from 1 to length(year_partition) - 1

  return(c(year_partition[index], year_partition[index+1]))

}

tree_by_years <- function(year_touple, water_data){
```

```r
  # filter for specific group of years
  water_data <- water_data %>% filter(year >= year_touple[1] &
                                      year <= year_touple[2])

  fitControl <- caret::trainControl(method="cv")

  tr.TP <- caret::train(TP ~ .,
                        data = water_data,
                        # what is method?
                        method = "rpart2",
                        trControl = fitControl,
                        # don't quite understand maxdepth
                        tuneGrid = data.frame(maxdepth=1:20))

  return(tr.TP)

}

tree_by_season <- function(season, min_year, max_year, year_interval, water_data){
  # season can be 1, 2, 3, 4 with 1 being spring
  ## this is already processed in line 70

  year_partition <- seq(min_year, max_year, year_interval)
  # make a list of each year interval
  year_touples <- lapply(1:(length(year_partition)-1), make_year_touples, year_partition)

  water_data <- water_data %>% filter(quarter == season)

  tree_models <- lapply(year_touples, tree_by_years, water_data)

  return(tree_models)

}
```

```r
# model for spring, 2000-2005, ..., 2015-2020
trees.sp.2000.2020 <- tree_by_season(1, # 1 stands for spring
                                     2000, # minimum year
                                     2020, # maximum year
                                     5, # 5 year intervals
                                     narm_water20) # dataset

# the 4 comes from (2020 - 2000) / 5
lapply(1:4, function(x) return(fancyRpartPlot(trees.sp.2000.2020[[x]]$finalModel,
                                              main = paste(as.character(x)) )))

lapply(1:4, function(x) return(plot(trees.sp.2000.2020[[x]],
                                    main = paste(as.character(x)) )))
```

TP spring

```r
# model for summer, 2000-2020
trees.su.2000.2020 <- tree_by_season(2, 2000, 2020, 5, narm_water20)

# the 4 comes from (2020 - 2000) / 5
lapply(1:4, function(x) return(fancyRpartPlot(trees.su.2000.2020[[x]]$finalModel,
                                              main = paste(as.character(x)) )))

lapply(1:4, function(x) return(plot(trees.su.2000.2020[[x]],
                                    main = paste(as.character(x)) )))
```

TP summer

```r
# model for fall, 2000-2020
trees.fa.2000.2020 <- tree_by_season(3, 2000, 2020, 5, narm_water20)

# the *2 is to insert line breaks
lapply(1:4*2, function(x)
  if (x %% 2 == 1) { # if x is even
    asis_output("\\\\[10cm]")
    } else {
      return(fancyRpartPlot(trees.fa.2000.2020[[x/2]]$finalModel,
                            main = paste(as.character(x/2)) ))
    } )

lapply(1:4, function(x) return(plot(trees.fa.2000.2020[[x]],
                                    main = paste(as.character(x)) )))
```

TP fall

```r
# model for winter, 2000-2020
trees.wi.2000.2020 <- tree_by_season(4, 2000, 2020, 5, narm_water20)

lapply(1:4, function(x) return(fancyRpartPlot(trees.wi.2000.2020[[x]]$finalModel,
                                              main = paste(as.character(x)) )))

lapply(1:4, function(x) return(plot(trees.wi.2000.2020[[x]],
                                    main = paste(as.character(x)) )))
```

TP winter