A Predictive Analysis of Doctor Performance Evaluation Data under the Merit-based Incentive Payment System (MIPS)

Xintao Ding (xd2222)
Sarah Yam (sy2806)

explain mips basically?

The Physician Compare website was created by the Centers for Medicare & Medicaid Services (CMS) in December 2010 as required by the Affordable Care Act (ACA) of 2010 to help patients assess and find doctors and hospitals. This dataset contains the information supplied to patients via that website, including patient satisfaction surveys and performance scores across over 100 metrics.

Looking at individual physician scores:

- MIPS
- Performance by measure category
- Organization MIPS

Question/Problem: How can we better help patients assess and find doctors, where the scoring and rating come in a format not easily accessible or understandable by the average individual?

Approach/Methods: Supervised learning for binary classification utilizing the MIPS as a target with other physician scoring methods as predictors (which we know some of the metrics are direct factors of the individual MIPS scoring, such as the IA, ACI, and Quality category scorings). Potential methods outlined below, including generalized linear models and tree methods.

Potential Methods for Binary Classification:
Using overall MIPS for individuals where MIPS>= 75, the positive payment adjustment threshold.

- could apply spline to other MIPS, ACI scorings since they're somewhat discrete in nature.
  - ACI >= 0 : clinician reported ACI category
  - ACI >= 50: clinician achieved base score for ACI
  - MIPS < 30: Negative Payment Adjustment
- Predictive MIPS >= 75, essentially.
- Methods to try:
  - PCA to explore relationship of numeric variables
  - PCA to explor clustering of observations
  - glmnet for binary classification (elastic model/penalized logit)
  - glm logit model with polynomials?
  - tree model if we can make it work? (Single Tree, Random Forest, Boosting, Dbarts???)
  - PLSDA or LDA
  - nnet or MARS

explain why some of these methods?

explain data – cite where it came from

```
suppressPackageStartupMessages(library(dplyr))
suppressPackageStartupMessages(library(stringr))
suppressPackageStartupMessages(library(tidyr))
suppressPackageStartupMessages(library(dmm))
suppressPackageStartupMessages(library(pcaPP))
suppressPackageStartupMessages(library(caret))
suppressPackageStartupMessages(library(splines))
suppressPackageStartupMessages(library(dbarts))
```

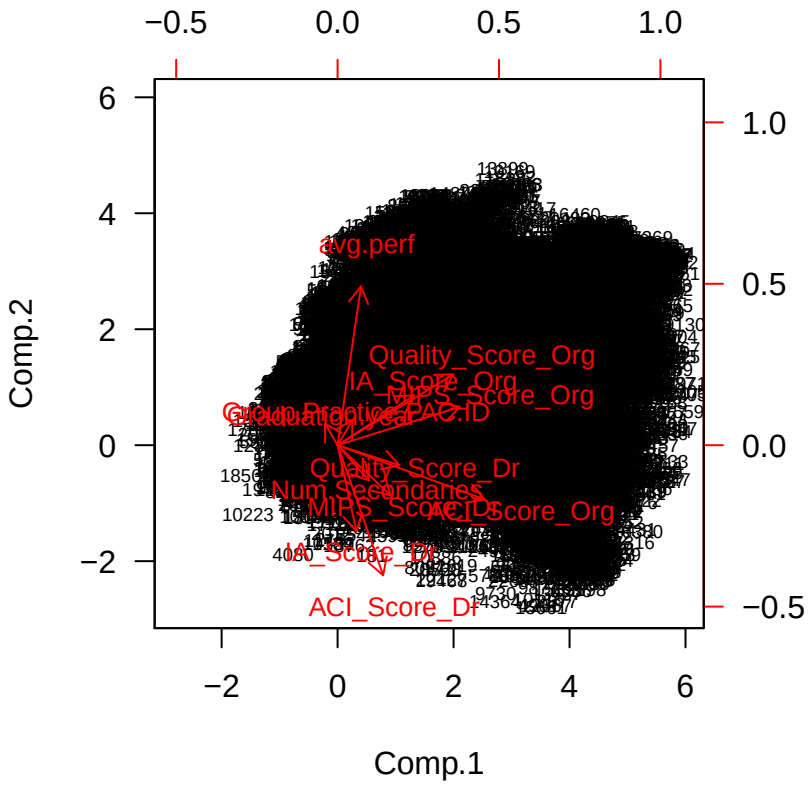
```
set.seed(70856775)
```

```
# Professional.Enrollment.ID is non-predictive, same with PAC_id
dr_scores <- read.csv("full_doctor_scoring.csv",
                      sep = ",", na = c("NA", "N/A"))
# remove majority of missingness by removing each task type
no_tasks <- dr_scores[c(1:23, 57)]
# add a "UKN" for unknown category
no_tasks$Credential <- factor(no_tasks$Credential,
                              levels = c(levels(dr_scores$Credential),
                                         "UKN"))
no_tasks$Credential[is.na(no_tasks$Credential)] <- "UKN"
# -1 for not reporting individual ACI Score (NA value)
no_tasks$ACI_Score_Dr[is.na(no_tasks$ACI_Score_Dr)] <- -1
```

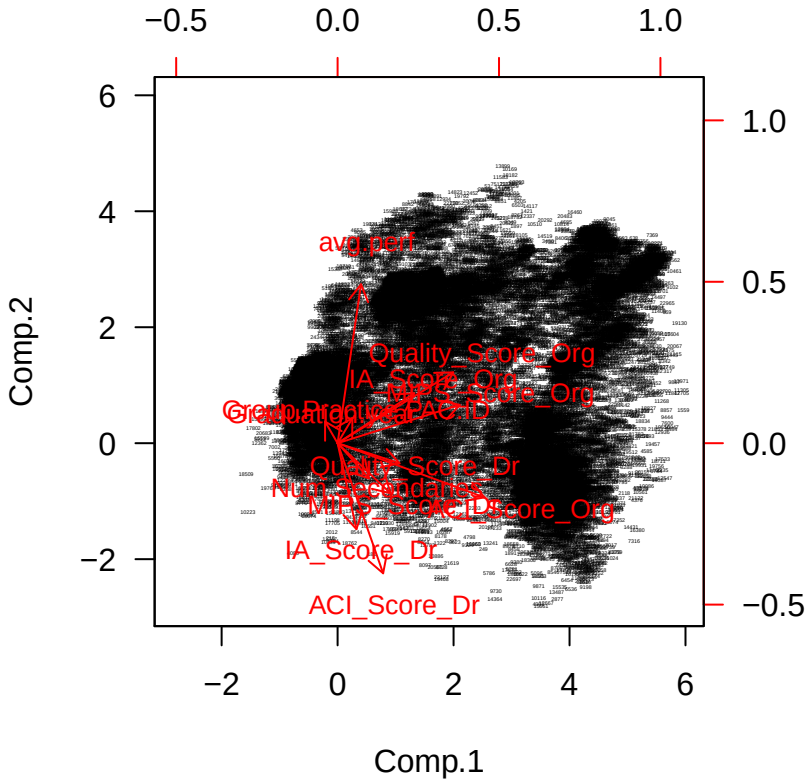
PCA to Explore Correlation of Variables

An initial PCA analysis to look understand how rows cluster based on the column variables. This is without considering the binary classifying aspect of MIPS >= 75„ and is just to understand some of the structure of the scores in the data.

```
# still minor missingness in Graduation Year and Quality Score
numerics <- no_tasks[, c(8, 10:11, 15:18, 20:24)] %>% na.omit(.)
pr_out <- PCAproj(numerics, scale = sd)
par(mar = c(5, 4, 3, 3) + 0.1, las = 1)

# visualization of any immediate outliers and the variables
biplot(pr_out, scale = 0, cex = c(0.6, 0.8))
```




```
# clusters within the observations visible
biplot(pr_out, scale = 0, cex = c(0.13, 0.8))
```



As mostly expected, the MIPS scores for the hospital organizations that each doctor works at are more correlated to each other than they are to the MIPS scores for each individual doctor. While the number of secondary specialties each doctor has is more correlated to the individual doctor MIPS scores, the group practice ID is more correlated to the organization scores. This is also fairly in line with our expectations that measures for the practice organization would cluster separately from the measures for the individual doctors. Interestingly, the average performance for an individual doctor across task categories seems to be more correlated to the organization scores though. Since performance is measured though individual patient reporting, their experience with the organization itself may be taken into consideration and bias their score, even if the doctor-patient interaction itself was positive.

In the second PCA plot, with the observation labels less cluttered from size, we see that there seem to be distinct clusters of observations. This would indicate groupings within the observations with distinguishing measurement characteristics. Many points do deviate from the groupings themselves. Still, overall, it doesn't appear that any observations seems like a major outlier, as seen in the first plot.

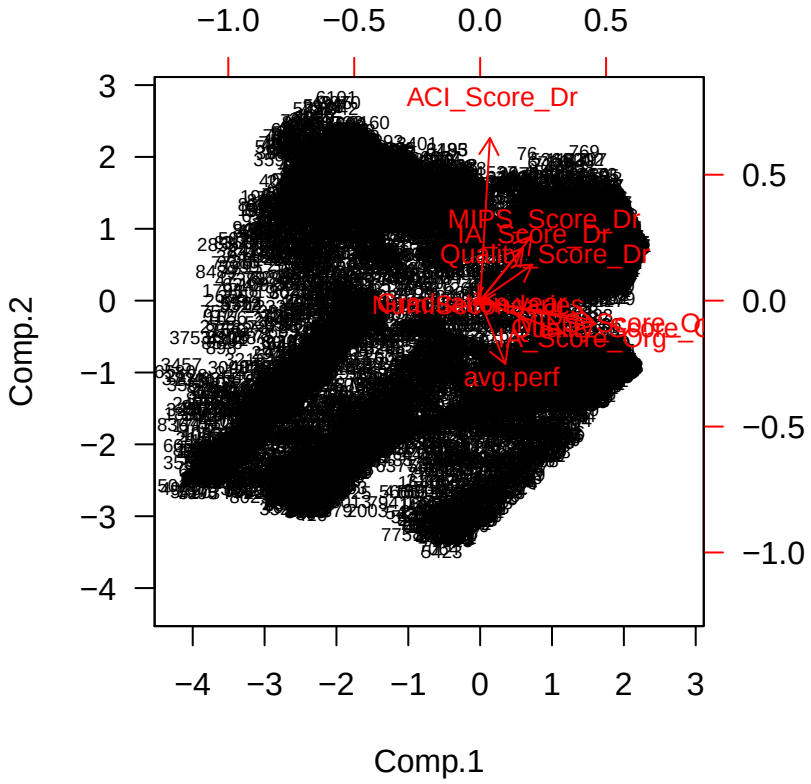
Prepare dataset for training/testing

```
# we do not expect PAC_id or Proefssional.Enrollment.ID to be predictive
# create factor for classification prediction (Individual Doctor MIPS Score >= 75)
# Score_source_org has no variance -- they're all group

no_tasks <- na.omit(no_tasks)[, c(3:13, 15:ncol(no_tasks))]
no_tasks$MIPS75_Dr <- factor(no_tasks$MIPS_Score_Dr >= 75.0,
                             levels = c(TRUE, FALSE),
                             labels = c("yes", "no"))
```

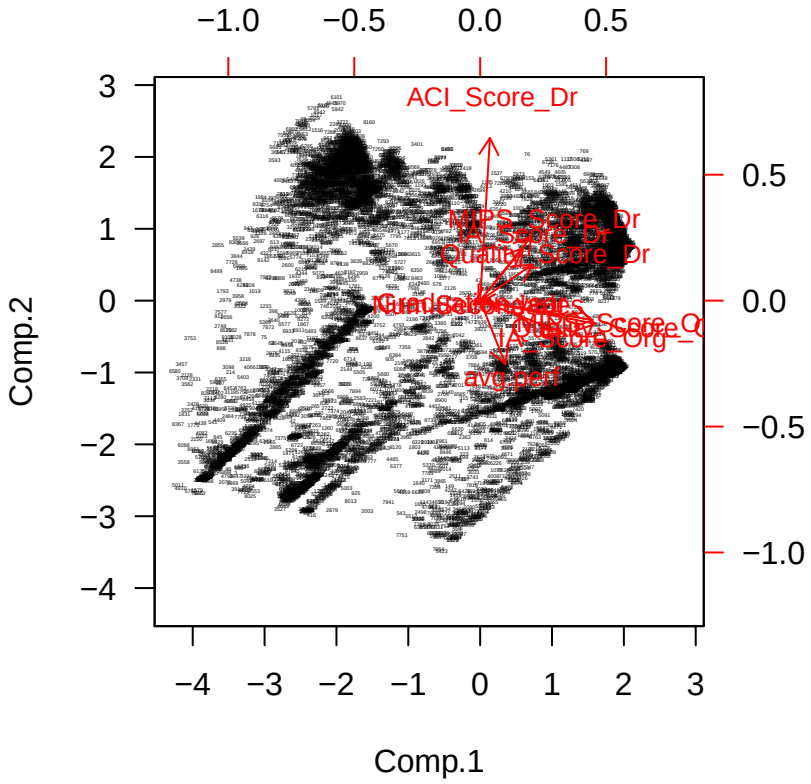
```r
# too many levels for partitioning separately
no_tasks$Prim.Schl <- interaction(no_tasks$Primary.specialty,
                                  no_tasks$Medical.school.name,
                                  sep = ":", drop = TRUE)



# variables we expect to be predictive
preds <- c("Gender", "Prim.Schl", "MIPS_Score_Org",
           "Num.Secondaries", "Quality_Score_Org",
           "IA_Score_Org", "IA_Score_Dr", "Graduation.year",
           "ACI_Score_Dr", "Quality_Score_Dr",
           "avg.perf", "MIPS75_Dr", "MIPS_Score_Dr")
no_tasks <- no_tasks[, preds]


remain <- group_by(no_tasks, Prim.Schl) %>%
  summarise(., count = n()) %>%
  filter(., count > 50) %>%
  .$Prim.Schl
no_tasks <- subset(no_tasks, Prim.Schl %in% remain)

numerical <-  no_tasks[, c("MIPS_Score_Org", "Num.Secondaries",
                           "Quality_Score_Org", "IA_Score_Org",
                           "IA_Score_Dr", "Graduation.year",
                           "ACI_Score_Dr", "Quality_Score_Dr",
                           "avg.perf",  "MIPS_Score_Dr")]

no_tasks <- no_tasks[, c("Gender", "Prim.Schl", "MIPS_Score_Org",
           "Num.Secondaries", "Quality_Score_Org", "IA_Score_Org",
           "IA_Score_Dr", "Graduation.year", "ACI_Score_Dr",
           "Quality_Score_Dr", "avg.perf", "MIPS75_Dr")]
```

Maybe explore PCA again for the things we are directly looking at and using the data that we've trimmed down to

```r
pca_out <- PCAproj(numerical, scale = sd)

par(mar = c(5, 4, 3, 3) + 0.1, las = 1)

# visualization of any immediate outliers and the variables
biplot(pca_out, scale = 0, cex = c(0.6, 0.8))
```




```
# clusters within the observations visible
biplot(pca_out, scale = 0, cex = c(0.13, 0.8))
```



When comparing just the numeric components of the preditors we will be training and testing on, scores measuring performance of individual doctors cluster apart from scores measuring performance of health care organizations. This makes sense as the criteria for evaluating an organization are likely more related to each other than they would be to the criteria for evaluating individual health care providers. Included in the doctor score cluster is the MIPS Score for individual dcotors, which is target variable we are trying to predict. The MIPS Score for individual doctors though, still seems to be fairly correlated with the organization-based scores. Overall though, we don't really appear to see any major outliers from the observations.

Similar to the original PCA plot as well, the average performance seems more correlated to the organization scores than the actual invidividual doctor scores, despite it being a measurement of average performance for individual doctors. Performance though, was measured across a large variety of different tasks, including e-Prescribing, Preventative Care and Screening, Diabetic Care, Nuclear Medicine, Patient Portal Access. Given the vast range of topics covered, it's apparent that many of these tasks, such as online patient portal access or disease screening, are more dependent on services provided and resources of the organization, rather than the individual caregiver.

The plot with smaller points again also shows us that the observations do a appear to group in distinctive patterns still, hopefully indicating that some of the underlying structure and characteristics of the data were still retained after further processing of the data.

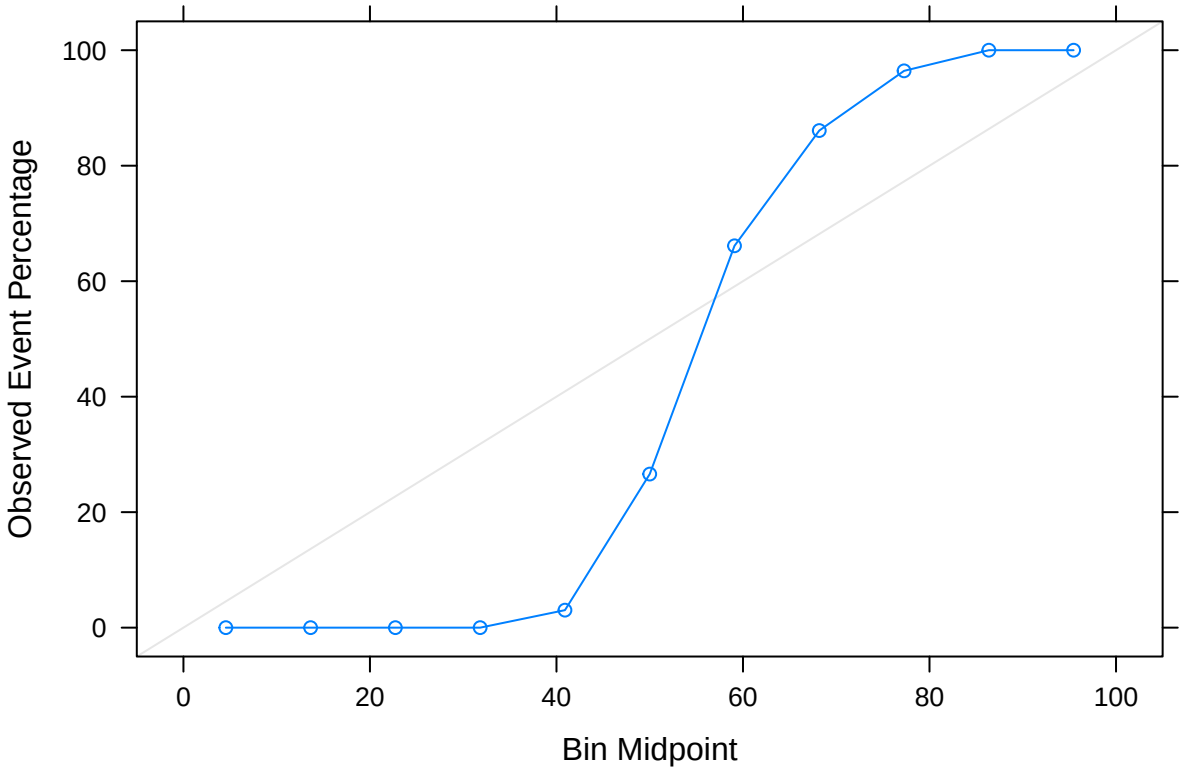
```
in_train <- createDataPartition(no_tasks$Prim.Schl,
                                p = 0.8, list = F)

training <- no_tasks[in_train, ]
testing <- no_tasks[-in_train, ]
```

Let's test with a basic linear model:

```
# testing without stratification, med school name, primary speciality
ols <- lm(MIPS75_Dr == "yes" ~ ., data = training)
yhat <- predict(ols, newdata = testing)
z_ols <- factor(yhat > 0.5, levels = c(TRUE, FALSE),
                labels = c("yes", "no"))
confusionMatrix(z_ols, reference = testing$MIPS75_Dr)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  yes   no
##        yes 1186   99
##        no    13  475
##
##                Accuracy : 0.9368
##                  95% CI : (0.9245, 0.9477)
##     No Information Rate : 0.6763
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8499
##
##  Mcnemar's Test P-Value : 9.61e-16
##
##             Sensitivity : 0.9892
##             Specificity : 0.8275
##          Pos Pred Value : 0.9230
##          Neg Pred Value : 0.9734
##              Prevalence : 0.6763
##          Detection Rate : 0.6689
##    Detection Prevalence : 0.7248
##       Balanced Accuracy : 0.9083
##
##        'Positive' Class : yes
##
```

```
calibration(MIPS75_Dr ~ yhat, data = testing) %>%
  plot(.)
```



Where, an ideally calibrated model should see 20% of observations being successful that have a predicted probability of about 0.2 if it finds that there's a 0.2 probability of success, our model seems more extreme, in almost a binary manner. In this case, when our models finds less than 0.5 probability of success, the proportion seen of observations being successful is far lower than that probability of success. In fact, none of the observation with a predicted probability greater than 0 and less than ~0.4 are successful. On the other hand, we see that that the proportion of observations being successful is far larger than the probability of success when the model says there's greater than ~0.6 probability of success. At a 0.8 probability of success, we actually see nearly 100% of those observations being successful. While the accuracy from this model is still quite good (not entirely unexpected since our dataset is fairly large and comprehensive), our data does not appear to quite fit an exact linear model, and would likely perform better with more flexible models.

GLM Models – Logit and GLMnet Penalized

```
# continous measurements can be scaled
# ACI is really a mixture of discrete and continuous

pp_names <- c("Num.Secondaries", "Quality_Score_Org",
              "Quality_Score_Dr", "IA_Score_Org",
              "IA_Score_Dr", "avg.perf")
pp <- list(center = pp_names, scale = pp_names)
```

```
in_train <- createDataPartition(no_tasks$MIPS75_Dr,
                                p = 0.6, list = F)
training <- no_tasks[in_train, ]
testing <- no_tasks[-in_train, ]
```

```r
logit <- glm(MIPS75_Dr ~ ., data = training,
             family = binomial(link = "logit"))
z <- predict(logit, newdata = testing,
             type = "response") > 0.5
z <- factor(z, levels = c(TRUE, FALSE),
            labels = c("no", "yes"), order = T)
confusionMatrix(z, testing$MIPS75_Dr)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  yes   no
##        yes 2313   86
##        no    78 1108
##
##                Accuracy : 0.9543
##                  95% CI : (0.9469, 0.9609)
##     No Information Rate : 0.6669
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8969
##
##  Mcnemar's Test P-Value : 0.5846
##
##             Sensitivity : 0.9674
##             Specificity : 0.9280
##          Pos Pred Value : 0.9642
##          Neg Pred Value : 0.9342
##              Prevalence : 0.6669
##          Detection Rate : 0.6452
##    Detection Prevalence : 0.6692
##       Balanced Accuracy : 0.9477
##
##        'Positive' Class : yes
##
```

```r
# glmnet PENALIZATION
ctrl <- trainControl(method = "cv", number = 3)
enet <- train(formula(logit), data = training,
              method = "glmnet", trControl = ctrl,
              tuneLength = 10, preProcess = pp)
enet_hat <- predict(enet, newdata = testing)
confusionMatrix(enet_hat, reference = testing$MIPS75_Dr)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  yes   no
##        yes 2311   95
##        no    80 1099
##
##                Accuracy : 0.9512
```

```
##                  95% CI : (0.9436, 0.958)
##     No Information Rate : 0.6669
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8898
##
##  Mcnemar's Test P-Value : 0.2899
##
##             Sensitivity : 0.9665
##             Specificity : 0.9204
##          Pos Pred Value : 0.9605
##          Neg Pred Value : 0.9321
##              Prevalence : 0.6669
##          Detection Rate : 0.6446
##    Detection Prevalence : 0.6711
##       Balanced Accuracy : 0.9435
##
##        'Positive' Class : yes
##
```

Analysis of GLM models, with and without penalization. Between each other and compared to linear.

Linear Models with Polynomials and Splines – how linear is our data? Do splines make it better?

explain why the splines and polynomials

Potential polynomials:

- Num.Secondaries

Potential splines:

- ACI: spline at -1:0, 0:50, 50:
- raw MIPS: spline at 0:30, 30:75, 75:

```
# with polynomials and splines
poly <- glm(MIPS75_Dr ~ . +
              bs(MIPS_Score_Org, knots = c(30, 75)) +
              bs(ACI_Score_Dr, knots = c(0, 50)) +
              poly(Num.Secondaries, degree = 2),
            data = training,
            family = binomial(link = "logit"))
poly_z <- predict(poly, newdata = testing,
                  type = "response") > 0.5
poly_z <- factor(poly_z, levels = c(TRUE, FALSE),
                 labels = c("no", "yes"), order = T)
confusionMatrix(poly_z, testing$MIPS75_Dr)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  yes   no
##        yes 2384   10
##        no     7 1184
##
##                Accuracy : 0.9953
##                  95% CI : (0.9924, 0.9972)
##     No Information Rate : 0.6669
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9893
##
##  Mcnemar's Test P-Value : 0.6276
##
##             Sensitivity : 0.9971
##             Specificity : 0.9916
##          Pos Pred Value : 0.9958
##          Neg Pred Value : 0.9941
##              Prevalence : 0.6669
##          Detection Rate : 0.6650
##    Detection Prevalence : 0.6678
##       Balanced Accuracy : 0.9943
##
##        'Positive' Class : yes
##
```

```
# glmnet PENALIZATION
# with polynomials and splines
poly_el <- train(formula(poly), data = training,
                 method = "glmnet", trControl = ctrl,
                 tuneLength = 10, preProcess = pp)
poly_yh <- predict(poly_el, newdata = testing)
confusionMatrix(poly_yh, reference = testing$MIPS75_Dr)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  yes   no
##        yes 2388   14
##        no     3 1180
##
##                Accuracy : 0.9953
##                  95% CI : (0.9924, 0.9972)
##     No Information Rate : 0.6669
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.9893
##
##  Mcnemar's Test P-Value : 0.01529
##
##             Sensitivity : 0.9987
##             Specificity : 0.9883
```

```
##          Pos Pred Value : 0.9942
##          Neg Pred Value : 0.9975
##              Prevalence : 0.6669
##          Detection Rate : 0.6661
##    Detection Prevalence : 0.6700
##       Balanced Accuracy : 0.9935
##
##        'Positive' Class : yes
##
```

Tree Methods

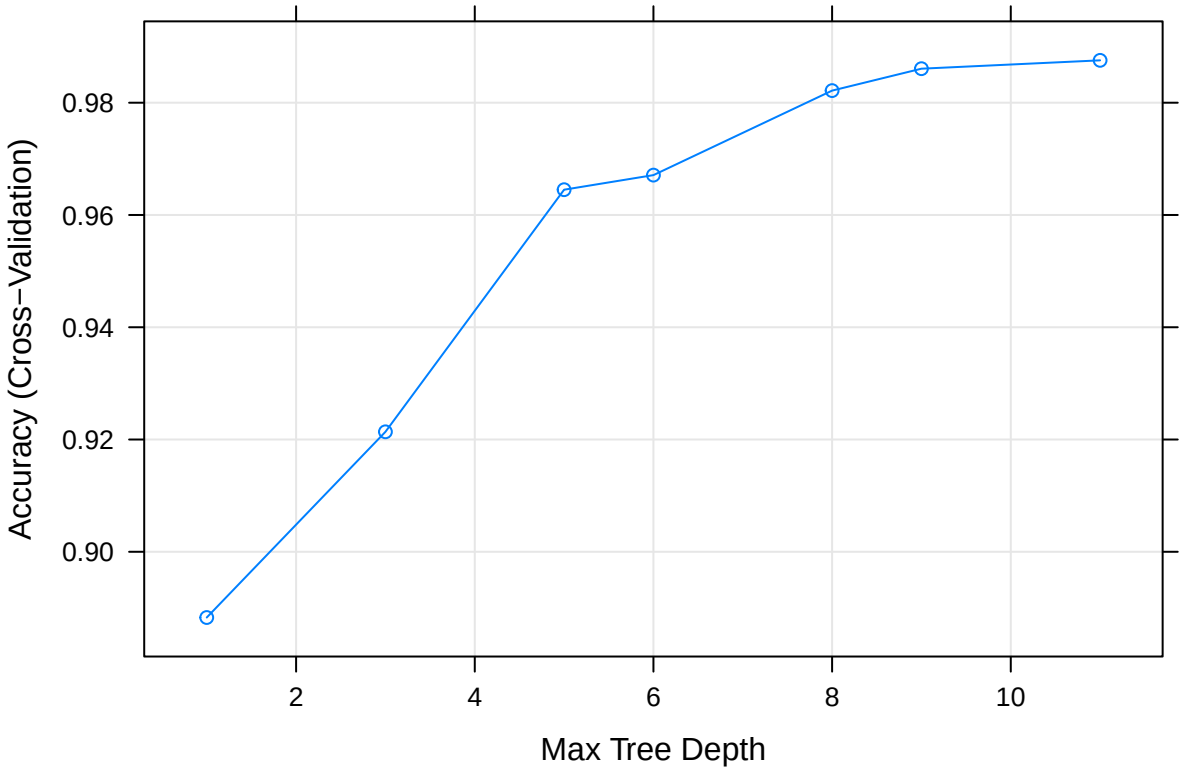
```
# training = 3587 yes, 1793 no (yes = .66)
# testing = 2391 yes, 1194 no (yes = 0.66)
# at random, it would be gussing 50% right, at only predicting yes or no, it would be guessing 33 or 66
# 80-90% are great trajectories then
```

Basic Single Tree Model on the same predictors

```
one_tree <- train(MIPS75_Dr ~ ., data = training, method = "rpart2",
                  tuneLength = 10, trControl = ctrl, preProcess = pp)
```

```
## note: only 7 possible values of the max tree depth from the initial fit.
##  Truncating the grid to 7 .
```

```
plot(one_tree)
```




```
confusionMatrix(testing$MIPS75_Dr, predict(one_tree, newdata = testing))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  yes   no
##        yes 2373   18
##        no    16 1178
##
##                Accuracy : 0.9905
##                  95% CI : (0.9868, 0.9934)
##     No Information Rate : 0.6664
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9787
##
##  Mcnemar's Test P-Value : 0.8638
##
##             Sensitivity : 0.9933
##             Specificity : 0.9849
##          Pos Pred Value : 0.9925
##          Neg Pred Value : 0.9866
##              Prevalence : 0.6664
##          Detection Rate : 0.6619
##    Detection Prevalence : 0.6669
```

```
##       Balanced Accuracy : 0.9891
##
##        'Positive' Class : yes
##
```

analysis of graph?

DBarts: Bayesian Additive Regression Trees

```
bart_out <- bart2(MIPS75_Dr == "yes" ~ ., data = training,
                  test = testing, k = 5, base = 0.4,
                  power = 1.3, keepTrees =  T)
```

```
##
## Running BART with binary y
##
## number of trees: 75
## number of chains: 4, number of threads 2
## Prior:
##  k: 5.000000
##  power and base for tree prior: 1.300000 0.400000
##  use quantiles for rule cut points: false
## data:
##  number of training observations: 5380
##  number of test observations: 3585
##  number of explanatory variables: 57
##
## Cutoff rules c in x<=c vs x>c
## Number of cutoffs: (var: number of possible c):
## (1: 100) (2: 100) (3: 100) (4: 100) (5: 100)
## (6: 100) (7: 100) (8: 100) (9: 100) (10: 100)
## (11: 100) (12: 100) (13: 100) (14: 100) (15: 100)
## (16: 100) (17: 100) (18: 100) (19: 100) (20: 100)
## (21: 100) (22: 100) (23: 100) (24: 100) (25: 100)
## (26: 100) (27: 100) (28: 100) (29: 100) (30: 100)
## (31: 100) (32: 100) (33: 100) (34: 100) (35: 100)
## (36: 100) (37: 100) (38: 100) (39: 100) (40: 100)
## (41: 100) (42: 100) (43: 100) (44: 100) (45: 100)
## (46: 100) (47: 100) (48: 100) (49: 100) (50: 100)
## (51: 100) (52: 100) (53: 100) (54: 100) (55: 100)
## (56: 100) (57: 100)
## Running mcmc loop:
## [1] iteration: 100 (of 500)
## [2] iteration: 100 (of 500)
## [1] iteration: 200 (of 500)
## [2] iteration: 200 (of 500)
## [1] iteration: 300 (of 500)
## [2] iteration: 300 (of 500)
## [1] iteration: 400 (of 500)
## [2] iteration: 400 (of 500)
## [1] iteration: 500 (of 500)
## [2] iteration: 500 (of 500)
```

```
## [3] iteration: 100 (of 500)
## [4] iteration: 100 (of 500)
## [3] iteration: 200 (of 500)
## [4] iteration: 200 (of 500)
## [3] iteration: 300 (of 500)
## [4] iteration: 300 (of 500)
## [3] iteration: 400 (of 500)
## [4] iteration: 400 (of 500)
## [3] iteration: 500 (of 500)
## [4] iteration: 500 (of 500)
## total seconds in loop: 16.734078
##
## Tree sizes, last iteration:
## [1] 2 2 2 3 1 4 3 1 2 5 2 2 5 1 2 1 2 2
## 2 2 2 2 3 1 1 2 3 2 3 2 1 1 1 3 4 1 4 2
## 2 4 1 2 1 2 1 2 2 1 1 2 1 2 1 2 1 3 1 1
## 2 1 1 3 1 2 1 2 3 1 2 2 2 2 3 3 2
## [2] 3 3 1 1 4 2 2 3 1 2 1 2 2 2 4 3 1 2
## 1 1 2 3 2 1 2 1 2 2 2 2 3 1 3 2 3 2 3 3
## 2 2 3 1 1 2 2 2 3 1 1 2 1 2 1 2 1 2 3 1
## 3 3 5 3 2 1 2 2 2 2 2 3 2 1 2 3 4
## [3] 2 1 2 2 1 3 1 2 2 2 2 1 1 2 2 3 1 1
## 3 1 2 5 3 3 2 2 1 3 2 2 1 1 4 2 2 2 2 1
## 2 2 3 2 3 1 2 2 2 2 2 2 4 1 1 1 3 4 1 3
## 4 3 2 3 1 2 3 2 1 1 3 1 2 4 2 3 2
## [4] 2 1 4 2 2 2 4 1 2 2 2 1 3 1 1 1 2 4
## 2 2 1 2 2 2 3 1 1 2 1 2 2 2 2 3 2 2 1 3
## 2 1 3 2 2 1 3 3 2 3 1 1 1 1 2 3 3 2 2 3
## 2 1 2 1 4 2 4 2 1 3 1 1 2 2 1 4 2
##
## Variable Usage, last iteration (var:count):
## (1: 0) (2: 2) (3: 5) (4: 1) (5: 2)
## (6: 7) (7: 1) (8: 2) (9: 3) (10: 3)
## (11: 2) (12: 4) (13: 1) (14: 9) (15: 1)
## (16: 4) (17: 1) (18: 5) (19: 2) (20: 2)
## (21: 5) (22: 1) (23: 3) (24: 2) (25: 4)
## (26: 6) (27: 3) (28: 0) (29: 7) (30: 2)
## (31: 2) (32: 1) (33: 2) (34: 4) (35: 2)
## (36: 1) (37: 1) (38: 3) (39: 4) (40: 5)
## (41: 1) (42: 1) (43: 2) (44: 4) (45: 3)
## (46: 4) (47: 4) (48: 1) (49: 32) (50: 0)
## (51: 15) (52: 3) (53: 20) (54: 3) (55: 35)
## (56: 71) (57: 1)
## DONE BART
```

```
confusionMatrix(factor(
  apply(pnorm(bart_out$yhat.test), 3, mean) > 0.5,
  levels = c(T, F), labels = levels(testing$MIPS75_Dr)),
  reference = testing$MIPS75_Dr)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  yes   no
```

```
##        yes 2348   87
##        no    43 1107
##
##                Accuracy : 0.9637
##                  95% CI : (0.9571, 0.9696)
##     No Information Rate : 0.6669
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9176
##
##  Mcnemar's Test P-Value : 0.0001624
##
##             Sensitivity : 0.9820
##             Specificity : 0.9271
##          Pos Pred Value : 0.9643
##          Neg Pred Value : 0.9626
##              Prevalence : 0.6669
##          Detection Rate : 0.6550
##    Detection Prevalence : 0.6792
##       Balanced Accuracy : 0.9546
##
##        'Positive' Class : yes
##
```

analysis comparison of trees, difference in trees (no need to interactions, etc because learned)

NNET

```
nnetGrid <- expand.grid(.decay = c(0, 0.01, .01),
                        .size = c(1:3))
nn <- train(MIPS75_Dr ~ . -Prim.Schl, data = training, method = "nnet",
            tuneLength = 10, trControl = ctrl, tuneGrid = nnetGrid,
            preProcess = pp, trace = FALSE)

## prim.schl was taken out as it is a zero-variance predictor

confusionMatrix(testing$MIPS75_Dr, predict(nn, newdata = testing))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  yes   no
##        yes 2295   96
##        no   121 1073
##
##                Accuracy : 0.9395
##                  95% CI : (0.9312, 0.9471)
##     No Information Rate : 0.6739
##     P-Value [Acc > NIR] : <2e-16
##
```

```
##                  Kappa : 0.863
##
##  Mcnemar's Test P-Value : 0.1033
##
##             Sensitivity : 0.9499
##             Specificity : 0.9179
##          Pos Pred Value : 0.9598
##          Neg Pred Value : 0.8987
##              Prevalence : 0.6739
##          Detection Rate : 0.6402
##    Detection Prevalence : 0.6669
##       Balanced Accuracy : 0.9339
##
##        'Positive' Class : yes
##
```

MARS

```
marsGrid <- expand.grid(.degree = 1:3, .nprune = 1:10)
MARS <- train(MIPS75_Dr ~ . -Prim.Schl, data = training, method = "earth",
                  tuneLength = 10, trControl = ctrl, preProcess = pp, trace = FALSE)
## prim.schl was taken out as it is a zero-variance predictor

confusionMatrix(testing$MIPS75_Dr, predict(MARS, newdata = testing))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  yes   no
##        yes 2381   10
##        no     7 1187
##
##                Accuracy : 0.9953
##                  95% CI : (0.9924, 0.9972)
##     No Information Rate : 0.6661
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9893
##
##  Mcnemar's Test P-Value : 0.6276
##
##             Sensitivity : 0.9971
##             Specificity : 0.9916
##          Pos Pred Value : 0.9958
##          Neg Pred Value : 0.9941
##              Prevalence : 0.6661
##          Detection Rate : 0.6642
##    Detection Prevalence : 0.6669
##       Balanced Accuracy : 0.9944
##
##        'Positive' Class : yes
##
```

"perfecrt separation happened, doesn't matter in an earth context since earth doesn't use t-value and other statistics that will be unrealiable for subsequent inference. Mostly likely happens in a cross-validation model since we are looking at smaller datasets"

final nnet/MARS analysis

overview of models and what worked best. why?

what did this tell us and how did this help solve problem?