CME 250 Prediction Competition

Jeffrey Barrera

May 16, 2016

SUNetID: barrera2

Kaggle Username: Jeff Barrera

Methodology

I began by trying to tackle the missing observations, since I couldn't build a model with NA values. At first I simply removed observations with NAs, but realized that wouldn't let me predict all 27 observations in the testing set, since some of these contained NAs. I then experimented with multiple imputation, but couldn't get this to work well on observations where people had only answered a handful of the survey questions – there thus weren't enough data points to effectively predict the missing values. Therefore, I eventually settled on imputing all missing observations with the median value for that column, a somewhat rudimentary but apparently fairly effective method.

I then randomly split the training data into a training set (comprised of roughly two-thirds of the observations), and an evaluation set with the remaining observations. I then built a model for each feature (exercise, age, and laptop OS) from the training set with a support vector machine, using cross-validation to select the hyperparameters like the kernel, penalty, and gamma values. I then tried predicting the observations in the evaluation set and checked the mean absolute error for each model.

I made a first submission to Kaggle using this approach, but got an error rate that was worse than simply predicting the median value for each outcome variable. Therefore, I realized I needed a more regularized approach to account for the high number of features relative to the small number of observations. Therefore, I created a second model for each feature using Lasso regression (logistic lasso in the case of laptop OS), cross-validating on the training set to find the optimal penalty value. To take advantage of both the lasso regularization and the higher-dimensional model captured by the SVM kernel, I then calculated the median of the predictions from both the lasso and SVM models, and used this as my final prediction.

R Code

```
library(e1071)
library(plyr)
library(mice)
library(Hmisc)
```

```r
library(glmnet)

setwd("~//Documents/School/Stanford/Classes/CME 250/Competition")

train_inputs <- read.csv("train_inputs.csv")
train_outputs <- read.csv("train_outputs.csv")

# rename vars
train_X <- rename(train_inputs, c(Do.you.own.a.car. = "own_car",
    What.is.your.primary.method.of.transportation.for.commuting.to.campus. = "commute_mo
    How.many.parking.or.traffic.tickets.have.you.received.in.the.last.year. = "parking_t
    How.many.news.articles.do.you.read.per.week. = "new_articles",
    What.is.your.height.in.inches. = "height_inches", What.is.your.shoe.size..using.US.M
    How.many.course.credits..excluding.research..are.you.enrolled.in.this.quarter = "cre
    What.is.the.area.code.of.your.personal.phone.number = "area_code",
    How.many.first.dates.have..you.been.on.in.the.last.year. = "first_dates",
    How.many.weddings.have.you.attended..as.a.guest..in.the.past.2.years. = "weddings_at
    How.many.days.a.week.do.you.cook.dinner. = "days_cook", How.many.cups.of.coffee.do.y
    How.many.servings.of.alcohol.do.you.consume.in.a.typical.week. = "alcohol",
    How.many.hours.of.sleep.do.you.get.on.weeknights...Sunday.Thursday. = "hours_sleep",
    Have.you.ever.voted.in.a.national..presidential..election. = "voted",
    Have.you.ever.had.a.full.time.job. = "job", How.many.years.total.have.you.live.in.th
    Have.you.ever.checked.your.credit.score. = "checked_credit",
    Did.you.get.a.seasonal.flu.vaccine.this.year..last.12.months.. = "flu_vaccine",
    How.old.were.you.when.you.got.your.first.cell.phone. = "age_first_phone",
    How.old.were.you.when.you.got.your.first.smart.pone. = "age_first_smartphone",
    Which.mobile.operating.system.does.your.cellphone.use. = "phone_os",
    Are.you.more.of.a.cat.person.or.a.dog.person. = "cat_dog",
    Which.best.describes.your.primary.field.of.study.research. = "study_field"))

train_Y = rename(train_outputs, c(How.many.hours.do.you.exercise.per.week. = "exercise",
    What.is.your.age. = "age", Which.type.of.personal.laptop.do.you.use. = "laptop_os"))

train_all <- merge(train_X, train_Y, by = "Id")
train_all$laptop_os = as.factor(train_all$laptop_os)

# try imputing missing data

train_all$age_first_smartphone <- with(train_all, impute(age_first_smartphone,
    mean))
train_all$age_first_phone <- with(train_all, impute(age_first_phone,
    median))
train_all$years_in_us <- with(train_all, impute(years_in_us,
    median))
```

```
train_all$hours_sleep <- with(train_all, impute(hours_sleep,
    median))
train_all$alcohol <- with(train_all, impute(alcohol, median))
train_all$coffee_drunk <- with(train_all, impute(coffee_drunk,
    median))
train_all$days_cook <- with(train_all, impute(days_cook, median))
train_all$weddings_attended <- with(train_all, impute(weddings_attended,
    median))
train_all$first_dates <- with(train_all, impute(first_dates,
    median))
train_all$area_code <- with(train_all, impute(area_code, median))
train_all$credits_taking <- with(train_all, impute(credits_taking,
    median))
train_all$shoe_size <- with(train_all, impute(shoe_size, median))
train_all$height_inches <- with(train_all, impute(height_inches,
    median))
train_all$new_articles <- with(train_all, impute(new_articles,
    median))
train_all$parking_tickets <- with(train_all, impute(parking_tickets,
    median))
train_all$exercise <- with(train_all, impute(exercise, median))
train_all$age <- with(train_all, impute(age, median))
train_all$laptop_os <- with(train_all, impute(laptop_os, median))

# split out X feature set for lasso models
train_features <- subset(train_all, select = -c(exercise, age,
    laptop_os, X.x, X.y, Id))

###### exercise

# subset df
exercise_df <- subset(train_all, select = -c(age, laptop_os,
    X.x, X.y, Id))
dim(exercise_df)

# create training and evaluation sets
train_rows <- sample(nrow(exercise_df), 40)
ex_train_df <- exercise_df[train_rows, ]
ex_eval_df <- exercise_df[-train_rows, ]

exercise_all_features <- data.matrix(train_features)
exercise_all_Y <- train_all$exercise

exercise_train_features <- data.matrix(train_features[train_rows,
```

```
    ])
exercise_eval_features <- data.matrix(train_features[-train_rows,
    ])

exercise_train_Y <- train_all$exercise[train_rows]
exercise_eval_Y <- train_all$exercise[-train_rows]

# try an SVM model

svr.tune.exercise <- tune(svm, exercise ~ ., data = exercise_df,
    ranges = list(cost = c(0.001, 0.01, 0.1, 1, 5, 10, 100),
        gamma = c(0.001, 0.01, 0.5, 1, 2, 3, 4), kernel = c("linear",
            "radial")))

svr.tune.exercise$best.model

svm.preds.exercise <- round(predict(svr.tune.exercise$best.model,
    ex_eval_df))
error <- ex_eval_df$exercise - svm.preds.exercise
mean(abs(error))


# try a lasso model
lasso.exercise <- glmnet(x = exercise_all_features, y = exercise_all_Y)
lasso_cv.exercise <- cv.glmnet(x = exercise_all_features, y = exercise_all_Y,
    nfolds = 10)
lasso_preds.exercise <- round(predict(lasso.exercise, newx = exercise_eval_features,
    s = lasso_cv.exercise$lambda.min))
lasso_preds.exercise

lasso.error <- ex_eval_df$exercise - lasso_preds.exercise
mean(abs(lasso.error))


# combine models
combined_preds.exercise <- data.frame(svm = svm.preds.exercise,
    lasso = lasso_preds.exercise)
combined_preds.exercise$median <- round(apply(combined_preds.exercise,
    1, median))

combined_preds.exercise

combined.error <- ex_eval_df$exercise - combined_preds.exercise$median
mean(abs(combined.error))
```

```
get_predictions <- function(svm_model, lasso_model, lasso_cv,
    svm_features, lasso_features) {

    svm_preds <- round(as.numeric(as.character(predict(svm_model$best.model,
        svm_features))))
    lasso_preds <- round(predict(lasso_model, newx = lasso_features,
        s = lasso_cv$lambda.min))
    combined_preds <- data.frame(svm = svm_preds, lasso = lasso_preds)
    combined_preds$median <- round(apply(combined_preds, 1, median))

    return(combined_preds)
}

get_predictions(svr.tune.exercise, lasso.exercise, lasso_cv.exercise,
    ex_eval_df, exercise_eval_features)

###### age

# subset df
age_df <- subset(train_all, select = -c(exercise, laptop_os,
    X.x, X.y, Id))
dim(age_df)

# create training and evaluation sets
train_rows <- sample(nrow(age_df), 40)
age_train_df <- age_df[train_rows, ]
age_eval_df <- age_df[-train_rows, ]

age_train_features <- data.matrix(train_features[train_rows,
    ])
age_eval_features <- data.matrix(train_features[-train_rows,
    ])

age_train_Y <- train_all$age[train_rows]
age_eval_Y <- train_all$age[-train_rows]

age_all_features <- data.matrix(train_features)
age_all_Y <- train_all$age

# try a SVM model

svr.tune.age <- tune(svm, age ~ ., data = age_train_df, ranges = list(cost = c(0.001,
    0.01, 0.1, 1, 5, 10, 100), gamma = c(0.001, 0.01, 0.5, 1,
    2, 3, 4), kernel = c("linear", "radial")))
```

```
svr.tune.age$best.model

svm.preds.age <- round(predict(svr.tune.age$best.model, age_eval_df))
svm.error <- age_eval_df$age - preds
mean(abs(svm.error))
sd(age_eval_df$age)


# try a lasso model
lasso.age <- glmnet(x = age_train_features, y = age_train_Y)
lasso_cv.age <- cv.glmnet(x = age_train_features, y = age_train_Y,
    nfolds = 10)
lasso_preds.age <- round(predict(lasso.age, newx = age_eval_features,
    s = lasso_cv.age$lambda.min))
lasso_preds.age

lasso.error <- age_eval_df$age - lasso_preds.age
mean(abs(lasso.error))


combined.error <- age_eval_df$age - combined_preds.age$median
mean(abs(combined.error))

# train final model on full dataset
svr.tune.age.all <- tune(svm, age ~ ., data = age_df, ranges = list(cost = c(0.001,
    0.01, 0.1, 1, 5, 10, 100), gamma = c(0.001, 0.01, 0.5, 1,
    2, 3, 4), kernel = c("linear", "radial")))
lasso.age.all <- glmnet(x = age_all_features, y = age_all_Y)
lasso_cv.age.all <- cv.glmnet(x = age_all_features, y = age_all_Y,
    nfolds = 10)

get_predictions(svr.tune.age.all, lasso.age.all, lasso_cv.age.all,
    age_eval_df, age_eval_features)


###### laptop_os

# subset df
laptop_os_df <- subset(train_all, select = -c(exercise, age,
    X.x, X.y, Id))
dim(laptop_os_df)

# create training and evaluation sets
train_rows <- sample(nrow(laptop_os_df), 40)
```

```
laptop_os_train_df <- laptop_os_df[train_rows, ]
laptop_os_eval_df <- laptop_os_df[-train_rows, ]

laptop_os_train_features <- data.matrix(train_features[train_rows,
    ])
laptop_os_eval_features <- data.matrix(train_features[-train_rows,
    ])

laptop_os_train_Y <- train_all$laptop_os[train_rows]
laptop_os_eval_Y <- train_all$laptop_os[-train_rows]

laptop_os_all_features <- data.matrix(train_features)
laptop_os_all_Y <- train_all$laptop_os

# try an SVM model

svr.tune.laptop_os <- tune(svm, laptop_os ~ ., data = laptop_os_train_df,
    ranges = list(cost = c(0.001, 0.01, 0.1, 1, 5, 10, 100),
        gamma = c(0.001, 0.01, 0.5, 1, 2, 3, 4), kernel = c("linear",
            "radial")))

svr.tune.laptop_os$best.model

preds <- predict(svr.tune.laptop_os$best.model, laptop_os_eval_df)
error <- ifelse(laptop_os_eval_df$laptop_os == preds, 0, 1)
mean(abs(error))

# try a lasso model
lasso.laptop_os <- glmnet(x = laptop_os_train_features, y = laptop_os_train_Y,
    family = "binomial")
lasso_cv.laptop_os <- cv.glmnet(x = laptop_os_train_features,
    y = laptop_os_train_Y, nfolds = 10, family = "binomial")
lasso_preds.laptop_os <- round(predict(lasso.laptop_os, newx = laptop_os_eval_features,
    s = lasso_cv.laptop_os$lambda.min))
lasso_preds.laptop_os


# train final model on full dataset
svr.tune.laptop_os.all <- tune(svm, laptop_os ~ ., data = laptop_os_df,
    ranges = list(cost = c(0.001, 0.01, 0.1, 1, 5, 10, 100),
        gamma = c(0.001, 0.01, 0.5, 1, 2, 3, 4), kernel = c("linear",
            "radial")))
lasso.laptop_os.all <- glmnet(x = laptop_os_all_features, y = laptop_os_all_Y,
    family = "binomial")
```

```
lasso_cv.laptop_os.all <- cv.glmnet(x = laptop_os_all_features,
    y = laptop_os_all_Y, nfolds = 10, family = "binomial")

get_predictions(svr.tune.laptop_os.all, lasso.laptop_os.all,
    lasso_cv.laptop_os.all, laptop_os_eval_df, laptop_os_eval_features)



########## Predict test set

test_inputs <- read.csv("test_inputs.csv")

test_X <- rename(test_inputs, c(Do.you.own.a.car. = "own_car",
    What.is.your.primary.method.of.transportation.for.commuting.to.campus. = "commute_mo
    How.many.parking.or.traffic.tickets.have.you.received.in.the.last.year. = "parking_t
    How.many.news.articles.do.you.read.per.week. = "new_articles",
    What.is.your.height.in.inches. = "height_inches", What.is.your.shoe.size..using.US.M
    How.many.course.credits..excluding.research..are.you.enrolled.in.this.quarter = "cre
    What.is.the.area.code.of.your.personal.phone.number = "area_code",
    How.many.first.dates.have..you.been.on.in.the.last.year. = "first_dates",
    How.many.weddings.have.you.attended..as.a.guest..in.the.past.2.years. = "weddings_at
    How.many.days.a.week.do.you.cook.dinner. = "days_cook", How.many.cups.of.coffee.do.y
    How.many.servings.of.alcohol.do.you.consume.in.a.typical.week. = "alcohol",
    How.many.hours.of.sleep.do.you.get.on.weeknights...Sunday.Thursday. = "hours_sleep",
    Have.you.ever.voted.in.a.national..presidential..election. = "voted",
    Have.you.ever.had.a.full.time.job. = "job", How.many.years.total.have.you.live.in.th
    Have.you.ever.checked.your.credit.score. = "checked_credit",
    Did.you.get.a.seasonal.flu.vaccine.this.year..last.12.months.. = "flu_vaccine",
    How.old.were.you.when.you.got.your.first.cell.phone. = "age_first_phone",
    How.old.were.you.when.you.got.your.first.smart.pone. = "age_first_smartphone",
    Which.mobile.operating.system.does.your.cellphone.use. = "phone_os",
    Are.you.more.of.a.cat.person.or.a.dog.person. = "cat_dog",
    Which.best.describes.your.primary.field.of.study.research. = "study_field"))


# impute missing data using median values -- too many missing
# variables for effective multiple imputation

test_X$age_first_smartphone <- with(test_X, impute(age_first_smartphone,
    median))
test_X$age_first_phone <- with(test_X, impute(age_first_phone,
    median))
test_X$years_in_us <- with(test_X, impute(years_in_us, median))
test_X$hours_sleep <- with(test_X, impute(hours_sleep, median))
```

```
test_X$alcohol <- with(test_X, impute(alcohol, median))
test_X$coffee_drunk <- with(test_X, impute(coffee_drunk, median))
test_X$days_cook <- with(test_X, impute(days_cook, median))
test_X$weddings_attended <- with(test_X, impute(weddings_attended,
    median))
test_X$first_dates <- with(test_X, impute(first_dates, median))
test_X$area_code <- with(test_X, impute(area_code, median))
test_X$credits_taking <- with(test_X, impute(credits_taking,
    median))
test_X$shoe_size <- with(test_X, impute(shoe_size, median))
test_X$height_inches <- with(test_X, impute(height_inches, median))
test_X$new_articles <- with(test_X, impute(new_articles, median))
test_X$parking_tickets <- with(test_X, impute(parking_tickets,
    median))

# is.na(test_X)

# resolve factor disparities by cbinding and then splitting
# the df
combined <- rbind(train_X, test_X)
test_X <- combined[(nrow(train_X) + 1):nrow(combined), ]
ids <- test_X$Id
test_X <- subset(test_X, select = -c(X, Id))

# predict testing data
pred_exercise <- get_predictions(svr.tune.exercise, lasso.exercise,
    lasso_cv.exercise, test_X, data.matrix(test_X))$median
pred_age <- get_predictions(svr.tune.age.all, lasso.age.all,
    lasso_cv.age.all, test_X, data.matrix(test_X))$median
pred_laptop_os <- get_predictions(svr.tune.laptop_os.all, lasso.laptop_os.all,
    lasso_cv.laptop_os.all, test_X, data.matrix(test_X))$median

# output
preds_df <- data.frame(Id = ids, Exercise = pred_exercise, Age = pred_age,
    Laptop = pred_laptop_os)

head(preds_df)
write.csv(preds_df, "submission5-14-16.csv", row.names = FALSE)
```