# Lasso

*Brooke Anderson*

*March 2, 2016*

```
knitr::opts_knit$set(root.dir = "..") # Reset root directory for analysis
library(lubridate) # To help handle dates
library(dplyr) # Data wrangling
library(tidyr) # Data wrangling
library(ggplot2) # Plotting
library(caret) # machine learning
```

Read in and clean up the data:

```
train <- read.csv("data/train.csv", as.is = TRUE) # `as.is` so `datetime` comes in as
                                                  # character, not factor
test <- read.csv("data/test.csv", as.is = TRUE)

train <- mutate(train,
                datetime = ymd_hms(datetime),
                year = factor(year(datetime)),
                hour = factor(hour(datetime)),
                month = month(datetime),
                yday = yday(datetime),
                weather = factor(weather, levels = c(1, 2, 3, 4),
                                 labels = c("Clear", "Mist", "Light Precip",
                                            "Heavy Precip")),
                season = factor(season, levels = c(1, 2, 3, 4),
                                labels = c("Spring", "Summer", "Fall", "Winter")),
                workingday = factor(workingday, levels = c(0, 1),
                                    labels = c("Holiday / weekend",
                                               "Working day")))

test <- mutate(test,
                datetime = ymd_hms(datetime),
                year = factor(year(datetime)),
                hour = factor(hour(datetime)),
                month = month(datetime),
                yday = yday(datetime),
                weather = factor(weather, levels = c(1, 2, 3, 4),
                                 labels = c("Clear", "Mist", "Light Precip",
                                            "Heavy Precip")),
                season = factor(season, levels = c(1, 2, 3, 4),
                                labels = c("Spring", "Summer", "Fall", "Winter")),
                workingday = factor(workingday, levels = c(0, 1),
                                    labels = c("Holiday / weekend",
                                               "Working day")))

write_test_preds <- function(test_preds, mod_name){
  out_file <- data.frame(datetime = as.character(test$datetime),
                         count = test_preds)
  out_name <- paste0("test_predictions/", mod_name, ".csv")
  write.csv(out_file, file = out_name, row.names = FALSE)
}
```

Lasso with glmnet and caret:

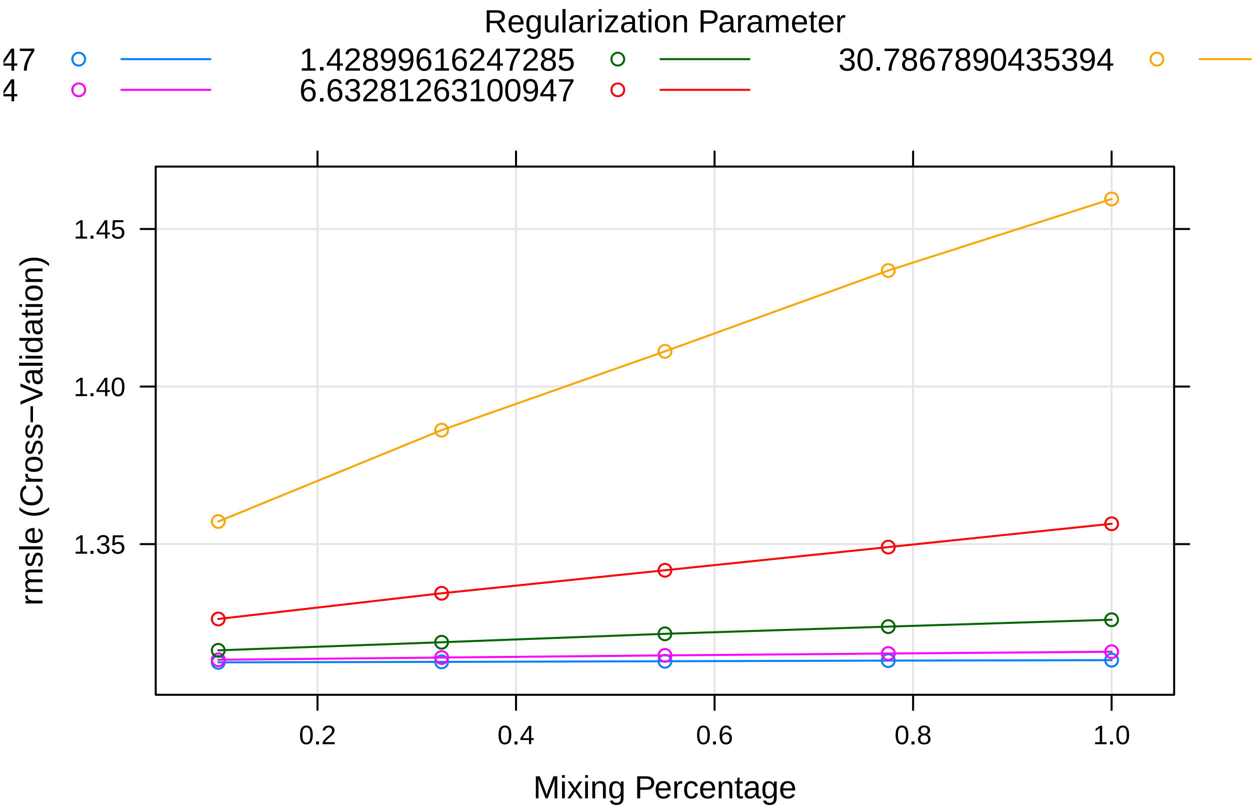
```
rmsle_fun <- function(data, lev = NULL, model = NULL, ...){
  log_p_1 <- log(exp(data$pred) + 1)
  log_a_1 <- log(data$obs + 1)
  sle <- (log_p_1 - log_a_1)^2
  rmsle <- sqrt(mean(sle))
  names(rmsle) <- "rmsle"
  return(rmsle)
}

my_train <- select(train, -datetime, -registered, - casual)
my_train <- model.matrix(count ~ year * season * workingday * hour +
                           holiday + temp + atemp + humidity + windspeed +
                           month + yday + weather,
                         data = my_train)
nzv <- nearZeroVar(my_train)
my_train <- my_train[, -nzv]

fitControl <- trainControl(method = "cv",
                       number = 5,
                       summaryFunction = rmsle_fun)

mod_1 <- train(y = train$count,
               x = my_train,
               preProcess = c("center", "scale"),
               method = "glmnet",
               trControl = fitControl,
               metric = "rmsle",
               maximize = FALSE,
               family = "poisson",
               tuneLength = 5)
```

```
plot(mod_1)
```

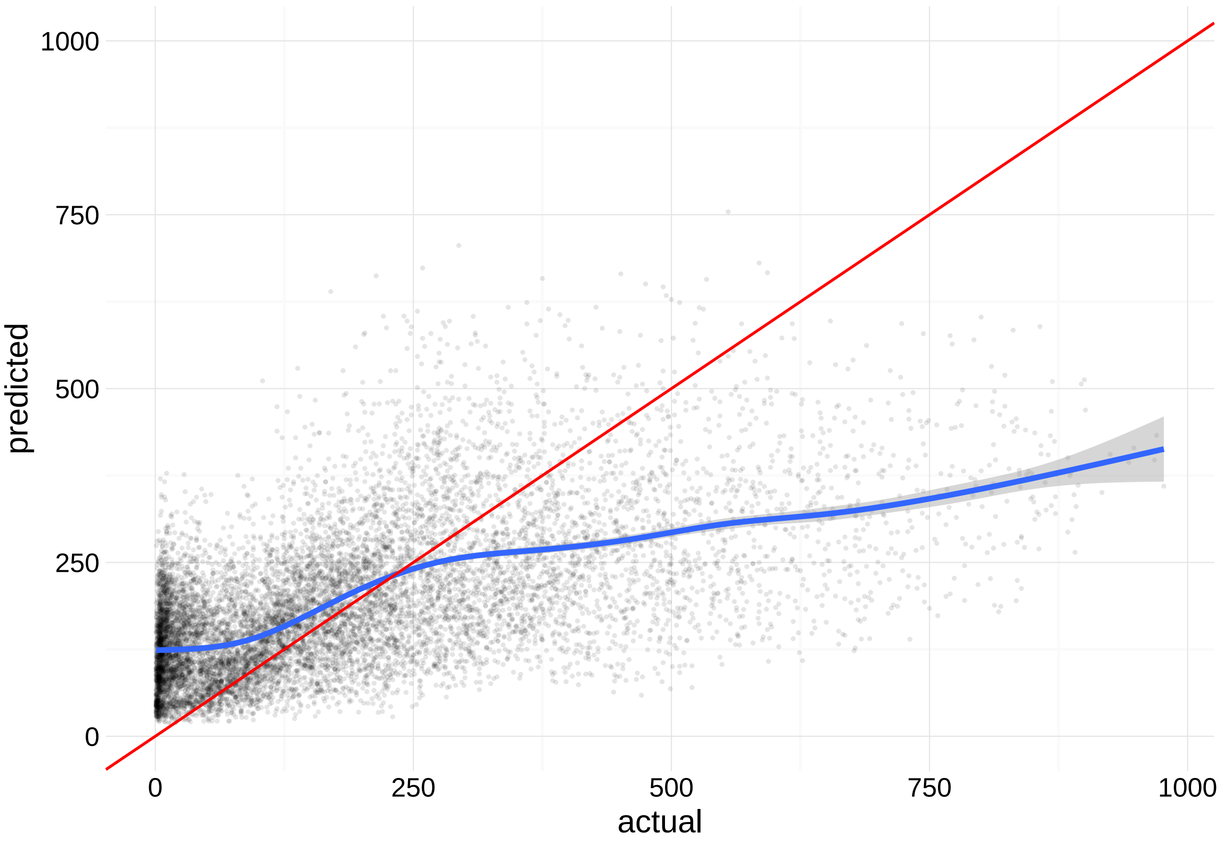


```r
rmsle <- function(train_preds, actual_preds){
  log_p_1 <- log(train_preds + 1)
  log_a_1 <- log(actual_preds + 1)
  sle <- (log_p_1 - log_a_1)^2
  rmsle <- sqrt(mean(sle))
  return(rmsle)
}

train_preds <- predict(mod_1, newdata = my_train)
train_preds <- exp(train_preds)
summary(train_preds)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   19.28  110.70  170.70  191.60  247.10  754.30
```

```r
to_plot <- data.frame(actual = train$count, predicted = train_preds)
ggplot(to_plot, aes(x = actual, y = predicted)) +
  geom_point(alpha = 0.1, size = 0.2) +
  geom_smooth() +
  theme_minimal() +
  geom_abline(intercept = 0, slope = 1, color = "red") +
  ylim(c(0, 1000))
```



```r
rmsle(train_preds, train$count)
```

```
## [1] 1.311138
```

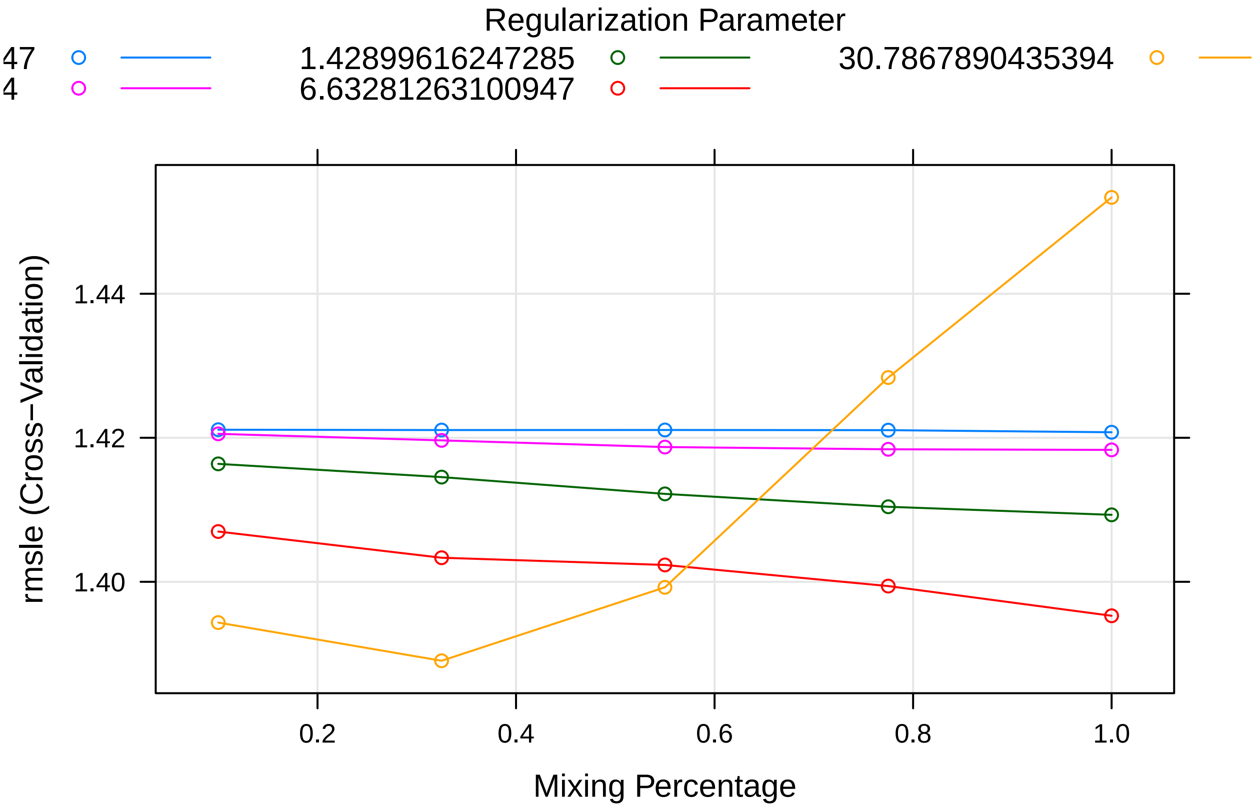
```r
my_test <- select(test, -datetime)
my_test <- model.matrix( ~ 1 + year * season * workingday * hour +
                           holiday + temp + atemp + humidity + windspeed +
                           month + yday + weather,
                         data = my_test)
my_test <- my_test[, -nzv]
test_preds <- predict(mod_1, newdata = my_test)
test_preds <- exp(test_preds)
write_test_preds(test_preds, mod = "elastic net, poisson")
```

```r
rmsle_fun <- function(data, lev = NULL, model = NULL, ...){
  data$pred[data$pred < 0] <- 0
  log_p_1 <- log(data$pred + 1)
  log_a_1 <- log(data$obs + 1)
  sle <- (log_p_1 - log_a_1)^2
  rmsle <- sqrt(mean(sle))
  names(rmsle) <- "rmsle"
  return(rmsle)
}

fitControl <- trainControl(method = "cv",
                       number = 5,
                       summaryFunction = rmsle_fun)

mod_1 <- train(y = train$count,
               x = my_train,
               preProcess = c("center", "scale"),
               method = "glmnet",
               trControl = fitControl,
               metric = "rmsle",
               maximize = FALSE,
               family = "gaussian",
               tuneLength = 5)
```

```
plot(mod_1)
```

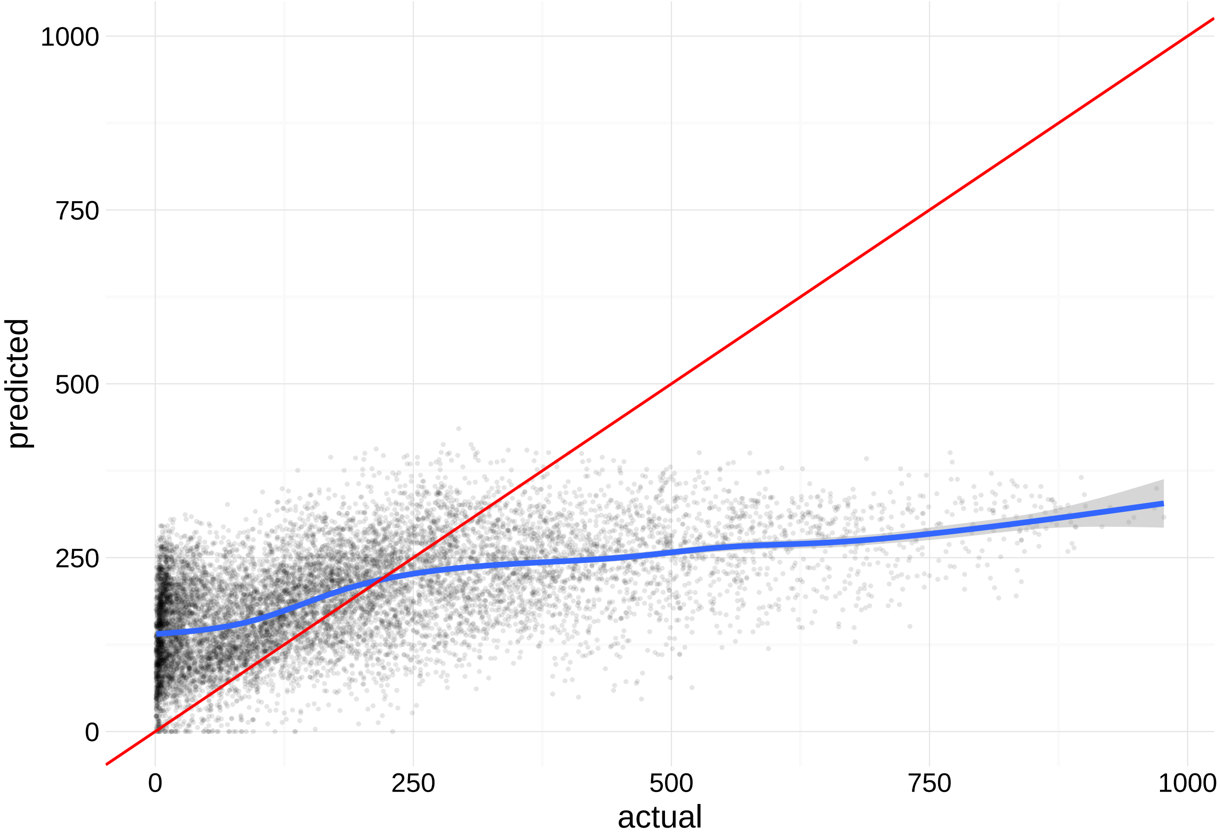


```
rmsle <- function(train_preds, actual_preds){
  train_preds[train_preds < 0] <- 0
  log_p_1 <- log(train_preds + 1)
  log_a_1 <- log(actual_preds + 1)
  sle <- (log_p_1 - log_a_1)^2
  rmsle <- sqrt(mean(sle))
  return(rmsle)
}

train_preds <- predict(mod_1, newdata = my_train)
train_preds[train_preds < 0] <- 0
summary(train_preds)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     0.0   135.5   191.6   191.6   246.5   435.6
```

```
to_plot <- data.frame(actual = train$count, predicted = train_preds)
ggplot(to_plot, aes(x = actual, y = predicted)) +
  geom_point(alpha = 0.1, size = 0.2) +
  geom_smooth() +
  theme_minimal() +
  geom_abline(intercept = 0, slope = 1, color = "red") +
  ylim(c(0, 1000))
```



```
rmsle(train_preds, train$count)
```

```
## [1] 1.388928
```

```
test_preds <- predict(mod_1, newdata = my_test)
write_test_preds(test_preds, mod = "elastic net, gaussian")
```