Beijing PM2.5 Data

Hourly data of PM2.5 from the US Embassy in Beijing. PM2.5 is a measure of particulate matter that have a diameter of less than 2.5 micrometers. They are an important measure of air quality for humans.

The data was downloaded from the UCI Machine Learning Repository and is calleed the Beijing PM2.5 Data Set

Load data

Loading the data and restricting to complete cases leaves about 41K observations. We will remove the No, day, and hour columns. Remaining columns are the year and month, pm2.5, temperature, pressure, combined wind direction. cumulated wind speed, cumulated hours of snow and cumulated hours of rain.

```
library(keras)
```

```
## Warning: package 'keras' was built under R version 3.4.3
```

```
df <- read.csv("PRSA_data.csv", header=TRUE)
df <- df[complete.cases(df), c(3, 6:13)]
head(df)
```

```
##    month pm2.5 DEWP TEMP PRES cbwd  Iws Is Ir
## 25     1   129  -16   -4 1020   SE 1.79  0  0
## 26     1   148  -15   -4 1020   SE 2.68  0  0
## 27     1   159  -11   -5 1021   SE 3.57  0  0
## 28     1   181   -7   -5 1022   SE 5.36  1  0
## 29     1   138   -7   -5 1022   SE 6.25  2  0
## 30     1   109   -7   -6 1022   SE 7.14  3  0
```

```
str(df)
```

```
## 'data.frame':    41757 obs. of  9 variables:
##  $ month: int  1 1 1 1 1 1 1 1 1 1 ...
##  $ pm2.5: int  129 148 159 181 138 109 105 124 120 132 ...
##  $ DEWP : int  -16 -15 -11 -7 -7 -7 -7 -7 -8 -7 ...
##  $ TEMP : num  -4 -4 -5 -5 -5 -6 -6 -5 -6 -5 ...
##  $ PRES : num  1020 1020 1021 1022 1022 ...
##  $ cbwd : Factor w/ 4 levels "cv","NE","NW",..: 4 4 4 4 4 4 4 4 4 4 ...
##  $ Iws  : num  1.79 2.68 3.57 5.36 6.25 ...
##  $ Is   : int  0 0 0 1 2 3 4 0 0 0 ...
##  $ Ir   : int  0 0 0 0 0 0 0 0 0 0 ...
```

```
N <- nrow(df)
p <- ncol(df)
t <- 2

X <- df[, -t]
Y <- df[, t]
```

Train/test split

```
set.seed(1234)
i <- sample(1:nrow(df), 0.8*nrow(df), replace=FALSE)
```

```
X_train <- data.matrix(X[i,])
Y_train <- Y[i]
X_test <- data.matrix(X[-i,])
Y_test <- Y[-i]
```

normalize data

```
means <- apply(X_train, 2, mean)
stdvs <- apply(X_train, 2, sd)
X_train <- scale(X_train, center=means, scale=stdvs)
X_test <- scale(X_test, center=means, scale=stdvs)
```

Try neural network

```
# build a model
model <- keras_model_sequential()
model %>%
  layer_dense(units=16, activation='relu', input_shape = dim(X_train)[[2]]) %>%
  layer_dense(units=16, activation='relu') %>%
  layer_dense(units=1)

model %>% compile(
  loss = 'mse',
  optimizer = 'rmsprop',
  metrics = c("mae")
)

model %>% fit(X_train, Y_train, epochs=100, batch_size=100, verbose=0)

results <- model %>% evaluate(X_test, Y_test, verbose=0)
results$mean_absolute_error
```

```
## [1] 44.33662
```

```
# how do you get predictions?
pred <- predict(model, X_test)
cor(pred, Y_test)  # 0.7
```

```
##           [,1]
## [1,] 0.7055536
```

```
mse <- mean((pred - Y_test)^2)   # 4211
sqrt(mse)  # 64.9
```

```
## [1] 65.2924
```

```
mae <- mean(abs(pred - Y_test))  # 44.4
```