Week3_SimonsenHomework

Steven Simonsen

2024-09-15

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(caret)
```

```
## Loading required package: ggplot2
```

```
## Loading required package: lattice
```

```
data <- read.csv("lab_3_data.csv")

training_ind <- createDataPartition(data$lodgepole_pine,
                                    p = 0.75,
                                    list = FALSE,
                                    times = 1)

training_set <- data[training_ind, ]
test_set <- data[-training_ind, ]

unique(training_set$wilderness_area)
```

```
## [1] "wilderness_area_1" "wilderness_area_3" "wilderness_area_4"
## [4] "wilderness_area_2"
```

```
unique(training_set$soil_type)
```

```
##  [1] "soil_type_18" "soil_type_30" "soil_type_12" "soil_type_29" "soil_type_20"
##  [6] "soil_type_23" "soil_type_24" "soil_type_32" "soil_type_10" "soil_type_5"
## [11] "soil_type_33" "soil_type_17" "soil_type_39" "soil_type_31" "soil_type_13"
## [16] "soil_type_1"  "soil_type_3"  "soil_type_11" "soil_type_14" "soil_type_6"
## [21] "soil_type_2"  "soil_type_16" "soil_type_40" "soil_type_35" "soil_type_4"
## [26] "soil_type_38" "soil_type_25" "soil_type_19" "soil_type_22" "soil_type_8"
## [31] "soil_type_9"  "soil_type_28" "soil_type_34" "soil_type_37" "soil_type_21"
## [36] "soil_type_36" "soil_type_26" "soil_type_27"
```

```
top_20_soil_types <- training_set %>%
  group_by(soil_type) %>%
```

```
  summarise(count = n()) %>%
  arrange(desc(count)) %>%
  select(soil_type) %>%
  top_n(20)
```

```
## Selecting by soil_type
```

```
training_set$soil_type <- ifelse(training_set$soil_type %in% top_20_soil_types$soil_type,
                                 training_set$soil_type,
                                 "other")

training_set$wilderness_area <- factor(training_set$wilderness_area)
training_set$soil_type <- factor(training_set$soil_type)

class(training_set$wilderness_area)
```

```
## [1] "factor"
```

```
class(training_set$soil_type)
```

```
## [1] "factor"
```

```
levels(training_set$wilderness_area)
```

```
## [1] "wilderness_area_1" "wilderness_area_2" "wilderness_area_3"
## [4] "wilderness_area_4"
```

```
levels(training_set$soil_type)
```

```
##  [1] "other"        "soil_type_27" "soil_type_28" "soil_type_29" "soil_type_3"
##  [6] "soil_type_30" "soil_type_31" "soil_type_32" "soil_type_33" "soil_type_34"
## [11] "soil_type_35" "soil_type_36" "soil_type_37" "soil_type_38" "soil_type_39"
## [16] "soil_type_4"  "soil_type_40" "soil_type_5"  "soil_type_6"  "soil_type_8"
## [21] "soil_type_9"
```

```
onehot_encoder <- dummyVars(~ wilderness_area + soil_type,
                            training_set[, c("wilderness_area", "soil_type")],
                            levelsOnly = TRUE,
                            fullRank = TRUE)


onehot_enc_training <- predict(onehot_encoder,
                               training_set[, c("wilderness_area", "soil_type")])


training_set <- cbind(training_set, onehot_enc_training)



test_set$soil_type <- ifelse(test_set$soil_type %in% top_20_soil_types$soil_type,
                             test_set$soil_type,
                             "other")


test_set$wilderness_area <- factor(test_set$wilderness_area)
test_set$soil_type <- factor(test_set$soil_type)
```

```
onehot_enc_test <- predict(onehot_encoder, test_set[, c("wilderness_area", "soil_type")])

test_set <- cbind(test_set, onehot_enc_test)

test_set[, -c(11:13)] <- scale(test_set[, -c(11:13)],
                               center = apply(training_set[, -c(11:13)], 2, mean),
                               scale = apply(training_set[, -c(11:13)], 2, sd))

training_set[, -c(11:13)] <- scale(training_set[, -c(11:13)])


training_features <- array(data = unlist(training_set[, -c(11:13)]),
                           dim = c(nrow(training_set), 33))

training_labels <- array(data = unlist(training_set[, 13]),
                         dim = c(nrow(training_set)))

test_features <- array(data = unlist(test_set[, -c(11:13)]),
                       dim = c(nrow(test_set), 33))

test_labels <- array(data = unlist(test_set[, 13]),
                     dim = c(nrow(test_set)))
```

#Exercises

1) After working through all of the code in the lab, run 'head(training_features)' and 'head(test_features)'. Copy and paste the output.

```
head(training_features)
```

```
##            [,1]       [,2]       [,3]         [,4]       [,5]       [,6]
## [1,] -1.6426185 -0.5703296 -0.9715763 -0.415721884 -0.4958759 -1.0522558
## [2,] -0.3001001 -0.6240062 -0.2863090 -1.134355361 -0.7549933  1.2991364
## [3,] -0.3801418 -0.2751081 -0.8345228 -0.009126627 -0.6858954  0.6021650
## [4,] -0.5147575 -0.7492517  0.8101187 -0.850684252 -0.4613270  0.3449109
## [5,] -1.3442811 -1.3307486 -0.8345228 -0.874323511 -0.8931893 -1.3017728
## [6,] -1.3260898  1.5320051 -0.9715763 -1.134355361 -0.8068168 -1.1425203
##             [,7]        [,8]       [,9]      [,10]      [,11]      [,12]
## [1,]  0.70765017  0.29776091 -0.2940898  2.6356814 -0.2312943 -0.8898425
## [2,]  0.93560794 -0.11246340 -0.7661563  2.9456742 -0.2312943 -0.8898425
## [3,]  0.78363609  0.60542914 -0.2154120  0.7847535 -0.2312943 -0.8898425
## [4,]  0.97360091 -1.29185829 -1.5791598  0.1926071 -0.2312943 -0.8898425
## [5,] -0.05220908  0.09264876  0.2304286 -0.5357255 -0.2312943 -0.8898425
## [6,] -0.31815982  0.50287306  0.6762692 -0.3829863 -0.2312943 -0.8898425
##           [,13]      [,14]       [,15]      [,16]       [,17]     [,18]
## [1,] -0.2501434 -0.0410525 -0.03273853 -0.5065499 -0.08778692 -0.229084
## [2,] -0.2501434 -0.0410525 -0.03273853 -0.5065499 -0.08778692  4.364543
## [3,] -0.2501434 -0.0410525 -0.03273853 -0.5065499 -0.08778692 -0.229084
## [4,] -0.2501434 -0.0410525 -0.03273853  1.9738372 -0.08778692 -0.229084
## [5,] -0.2501434 -0.0410525 -0.03273853  1.9738372 -0.08778692 -0.229084
## [6,] -0.2501434 -0.0410525 -0.03273853 -0.5065499 -0.08778692 -0.229084
##           [,19]      [,20]      [,21]       [,22]       [,23]       [,24]
## [1,] -0.2223479 -0.3149516 -0.2914897 -0.05254272 -0.05676569 -0.01749279
## [2,] -0.2223479 -0.3149516 -0.2914897 -0.05254272 -0.05676569 -0.01749279
## [3,] -0.2223479 -0.3149516 -0.2914897 -0.05254272 -0.05676569 -0.01749279
```

```
## [4,] -0.2223479 -0.3149516 -0.2914897 -0.05254272 -0.05676569 -0.01749279
## [5,] -0.2223479 -0.3149516 -0.2914897 -0.05254272 -0.05676569 -0.01749279
## [6,] -0.2223479 -0.3149516 -0.2914897 -0.05254272 -0.05676569 -0.01749279
##            [,25]      [,26]      [,27]      [,28]     [,29]       [,30]
## [1,] -0.01236832 -0.1628953 -0.1516614 -0.1413112 -0.125262 -0.04632411
## [2,] -0.01236832 -0.1628953 -0.1516614 -0.1413112 -0.125262 -0.04632411
## [3,] -0.01236832 -0.1628953 -0.1516614 -0.1413112 -0.125262 -0.04632411
## [4,] -0.01236832 -0.1628953 -0.1516614 -0.1413112 -0.125262 -0.04632411
## [5,] -0.01236832 -0.1628953 -0.1516614 -0.1413112 -0.125262 -0.04632411
## [6,] -0.01236832 -0.1628953 -0.1516614 -0.1413112 -0.125262 -0.04632411
##           [,31]       [,32]       [,33]
## [1,] -0.1055234 -0.01236832 -0.04288122
## [2,] -0.1055234 -0.01236832 -0.04288122
## [3,] -0.1055234 -0.01236832 -0.04288122
## [4,] -0.1055234 -0.01236832 -0.04288122
## [5,] -0.1055234 -0.01236832 -0.04288122
## [6,] -0.1055234 -0.01236832 -0.04288122
```

```
head(test_features)
```

```
##             [,1]       [,2]       [,3]        [,4]        [,5]       [,6]
## [1,] -0.83492448 -0.2035393  0.2619048 -0.19824070  0.55786804 -0.4926154
## [2,] -0.79490360 -0.3019465  0.6730652 -1.13435536 -0.78954230  0.2501331
## [3,] -0.16912266  0.6463407 -0.2863090 -0.95942484 -0.68589535  1.6898531
## [4,] -0.03086873 -1.0534193 -1.2456832 -1.13435536 -0.70316985  1.4906263
## [5,]  1.01331226  0.4137420 -0.2863090  0.03342404  0.05690778  0.7846284
## [6,] -0.85311578 -0.6061140 -0.0122021 -0.14150648 -0.11583713 -1.3617343
##            [,7]        [,8]        [,9]      [,10]      [,11]      [,12]
## [1,]  1.2015587  0.40031699 -0.81860815 -0.8374418 -0.2312943 -0.8898425
## [2,]  1.3535305 -0.06118536 -1.26444878  0.1813210 -0.2312943 -0.8898425
## [3,] -0.3941457  1.42587776  1.12210988 -0.3641761 -0.2312943 -0.8898425
## [4,]  0.2897276  0.24648287  0.04684719  0.2211987 -0.2312943 -0.8898425
## [5,]  0.0617698  1.42587776  0.70249517  0.9126632 -0.2312943 -0.8898425
## [6,]  1.0875798 -0.42013163 -1.08086734  0.3257836 -0.2312943 -0.8898425
##           [,13]      [,14]       [,15]      [,16]       [,17]     [,18]
## [1,] -0.2501434 -0.0410525 -0.03273853 -0.5065499 -0.08778692  4.364543
## [2,] -0.2501434 -0.0410525 -0.03273853  1.9738372 -0.08778692 -0.229084
## [3,] -0.2501434 -0.0410525 -0.03273853 -0.5065499 -0.08778692  4.364543
## [4,] -0.2501434 -0.0410525 -0.03273853 -0.5065499 -0.08778692 -0.229084
## [5,] -0.2501434 -0.0410525 -0.03273853 -0.5065499 -0.08778692  4.364543
## [6,] -0.2501434 -0.0410525 -0.03273853  1.9738372 -0.08778692 -0.229084
##           [,19]      [,20]      [,21]       [,22]       [,23]       [,24]
## [1,] -0.2223479 -0.3149516 -0.2914897 -0.05254272 -0.05676569 -0.01749279
## [2,] -0.2223479 -0.3149516 -0.2914897 -0.05254272 -0.05676569 -0.01749279
## [3,] -0.2223479 -0.3149516 -0.2914897 -0.05254272 -0.05676569 -0.01749279
## [4,] -0.2223479 -0.3149516 -0.2914897 -0.05254272 -0.05676569 -0.01749279
## [5,] -0.2223479 -0.3149516 -0.2914897 -0.05254272 -0.05676569 -0.01749279
## [6,] -0.2223479 -0.3149516 -0.2914897 -0.05254272 -0.05676569 -0.01749279
##            [,25]      [,26]      [,27]      [,28]     [,29]       [,30]
## [1,] -0.01236832 -0.1628953 -0.1516614 -0.1413112 -0.125262 -0.04632411
## [2,] -0.01236832 -0.1628953 -0.1516614 -0.1413112 -0.125262 -0.04632411
## [3,] -0.01236832 -0.1628953 -0.1516614 -0.1413112 -0.125262 -0.04632411
## [4,] -0.01236832 -0.1628953 -0.1516614 -0.1413112 -0.125262 -0.04632411
## [5,] -0.01236832 -0.1628953 -0.1516614 -0.1413112 -0.125262 -0.04632411
## [6,] -0.01236832 -0.1628953 -0.1516614 -0.1413112 -0.125262 -0.04632411
```

```
##           [,31]       [,32]       [,33]
## [1,] -0.1055234 -0.01236832 -0.04288122
## [2,] -0.1055234 -0.01236832 -0.04288122
## [3,] -0.1055234 -0.01236832 -0.04288122
## [4,] -0.1055234 -0.01236832 -0.04288122
## [5,] -0.1055234 -0.01236832 -0.04288122
## [6,] -0.1055234 -0.01236832 -0.04288122
```

2) What is the rank of the tensor 'training_features'? What is the shape of 'training_features'? How many dimensions does 'training_features' have along the second axis?

```
dim(training_features)
```

```
## [1] 6537   33
```

The rank of the tensor 'training_features' is rank 2. The shape is (6537, 33). The 'training_features' tensor has 33 dimensions along its second axis.

3) State two situations where scaling the numerical variables is important.

First, it is important to scale numerical variables when using machine learning methods that rely on gradient-descent for optimization (such as neural networks) for purposes of model stability, convergence speed, and to be sure there is equal contribution of features.

The second situation where it is important to scale numerical variables is when a model is built upon the notion of distance (such as k-means clustering, principal component analysis, or k-nearest neighbor). The reason for this is to prevent one numerical variable from dominating the others simply because they are measured in different units (e.g., meters vs miles).