CIS635-Project

Tyler Reed

4/19/2021

Preliminary Analysis

```r
trainA <- as_tibble(trainA)
# Calculate summary statistics and produce visuals to check for outliers/noise/NAs
trainA %>%
  summary() %>%
  kable(caption = "Summary Table of `trainA`")
```

```r
# Test for duplicate records
length(unique(trainA$id)) == nrow(trainA)
```

```
## [1] TRUE
```

```r
# Test for missing values by row: no more than one to avoid considering removal of instance
train_A_byrow<- rowSums(is.na(trainA))
max(train_A_byrow)
```

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

```r
trainB <- as_tibble(trainB)

# Calculate summary statistics and produce visuals to check for outliers/noise/NAs
trainB %>%
  summary(trainB) %>%
  kable(caption = "Summary Table of `trainB`")
```

Table 1: Summary Table of ‘trainA‘

	id	temp	bpSys	vo2	throat	atRisk
	Min. : 0	Min. : 15.00	Min. : 20.0	Min. : 10.00	Min. : 81	Min. :0.0000
	1st Qu.:1673	1st Qu.: 97.79	1st Qu.:119.0	1st Qu.: 34.00	1st Qu.: 97	1st Qu.:0.0000
	Median :3352	Median : 98.19	Median :124.0	Median : 39.00	Median :100	Median :0.0000
	Mean :3376	Mean : 98.47	Mean :124.6	Mean : 37.76	Mean :100	Mean :0.4652
	3rd Qu.:5084	3rd Qu.: 98.93	3rd Qu.:130.0	3rd Qu.: 42.00	3rd Qu.:103	3rd Qu.:1.0000
	Max. :6780	Max. :198.83	Max. :501.0	Max. :150.00	Max. :122	Max. :1.0000
	NA	NA’s :1	NA’s :1	NA’s :2	NA’s :1	NA

Table 2: Summary Table of ‘trainB‘

	id	headA	bodyA	cough	runny	nausea	diarrhea
	Min. : 0	Min. : 0.000	Min. :1.000	Min. :0.0000	Min. :0.0000	Min. :0.0000	Min. :0.000
	1st Qu.:1673	1st Qu.: 3.000	1st Qu.:4.000	1st Qu.:0.0000	1st Qu.:0.0000	1st Qu.:0.0000	1st Qu.:0.000
	Median :3352	Median : 3.000	Median :4.000	Median :0.0000	Median :0.0000	Median :0.0000	Median :0.000
	Mean :3376	Mean : 3.461	Mean :4.016	Mean :0.3418	Mean :0.1986	Mean :0.2367	Mean :0.102
	3rd Qu.:5084	3rd Qu.: 4.000	3rd Qu.:4.000	3rd Qu.:1.0000	3rd Qu.:0.0000	3rd Qu.:0.0000	3rd Qu.:0.000
	Max. :6780	Max. :100.000	Max. :7.000	Max. :1.0000	Max. :1.0000	Max. :5.0000	Max. :1.000
	NA	NA’s :1	NA	NA	NA’s :1	NA	NA’s :1

```r
# Test for duplicate records
length(unique(trainB$id)) == nrow(trainB)
```

```
## [1] TRUE
```

```r
# Test for missing values by row: no more than one to avoid considering removal of instance
train_B_byrow <- rowSums(is.na(trainB))
max(train_B_byrow)
```

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

Results of `trainA`

No more than one NA per dataset

- id: looks good and no duplicates
- temp: 1 NA, and min and max troublesome, use average
- bbSys: 1 NA, and min and max troublesome, use average
- vo2: 2 NA, max troublesome
- throat: 1 NA, max troublesome
- atRisk: looks good

Results of `trainB`

- id: looks good and no duplicates
- headA: 1 NA, max troublesome
- bodyA: looks good
- cough: looks good
- runny: 1 NA
- nausea: max is troublesome
- diarrhea: 1 NA
- atRisk: looks good

Confirm outliers/missing data are cleaned

Table 3: New Summary Statistics to Confirm Cleaned Training Data

	id	temp	bpSys	vo2	throat	headA	bodyA	cough	run
	Min. : 0	Min. : 96.18	Min. : 97.0	Min. :10.00	Min. : 81	3 :2970	1: 7	0:3570	0:43
	1st Qu.:1673	1st Qu.: 97.79	1st Qu.:119.0	1st Qu.:34.00	1st Qu.: 97	5 : 906	2: 91	1:1854	1:10
	Median :3352	Median : 98.19	Median :124.0	Median :39.00	Median :100	4 : 715	3: 709	NA	NA
	Mean :3376	Mean : 98.47	Mean :124.5	Mean :37.74	Mean :100	2 : 544	4:3745	NA	NA
	3rd Qu.:5084	3rd Qu.: 98.93	3rd Qu.:130.0	3rd Qu.:42.00	3rd Qu.:103	6 : 172	5: 753	NA	NA
	Max. :6780	Max. :101.40	Max. :149.0	Max. :58.00	Max. :116	1 : 91	6: 110	NA	NA
	NA	NA	NA	NA	NA	(Other): 26	7: 9	NA	NA

```r
kable(summary(xTrain), caption = "New Summary Statistics to Confirm Cleaned Training Data")
```

Rationale The table above provides confirmation of the cleaned dataset as no more missing values are detected and all variables are within established ranges.

Selecting a Classifier

```r
normalize <- function(x) {
return ((x - min(x)) / (max(x) - min(x)))
}

xTrain_norm <- c("")
xTrain_norm <- xTrain
for (i in 2:5) {
    xTrain_norm[, i] <- normalize(xTrain[, i])
}

xTest_norm <- c("")
xTest_norm <- xTest
for (i in 2:5) {
    xTest_norm[, i] <- normalize(xTest[, i])
}

xTrain_norm <-xTrain_norm %>%
  dummy_cols(select_columns = c("headA", "bodyA")) %>%
  select(-headA, -bodyA) %>%
  relocate(atRisk, .after = last_col()) %>%
  mutate(across(where(is.factor), unfactor)) %>%
  mutate(across(where(is.integer), as.numeric))

xTest_norm <-xTest_norm %>%
  dummy_cols(select_columns = c("headA", "bodyA")) %>%
  select(-headA, -bodyA) %>%
  relocate(atRisk, .after = last_col()) %>%
  mutate(across(where(is.factor), unfactor)) %>%
  mutate(across(where(is.integer), as.numeric))
```

```r
# Create formula for factor variables depending on how many levels are used in data
xTrain_norm_formula <- c("")
for (i in 2:(ncol(xTrain_norm) - 1)) {
  if (i < (ncol(xTrain_norm) - 1)) {
    xTrain_norm_formula <- paste0(xTrain_norm_formula, names(xTrain_norm[, i]), "+")
  } else {
      xTrain_norm_formula <- paste0(xTrain_norm_formula, names(xTrain_norm[, i]))
      xTrain_norm_formula <- paste0("atRisk~", xTrain_norm_formula)
  }
}
```

Rationale The code chunk above includes normalizing the dataset which was essential to testing the ANN classifier due to the wide range of scales among the variables.

Best ANN Results after a few iterations

- 85.41% accuracy with 1 hidden nodes
- 84.19% recall with 1 hidden nodes
- 85.41% accuracy with 2 hidden nodes
- 83.87% recall with 2 hidden nodes
- 70.00% accuracy with 3 hidden nodes
- 75.41% recall with 3 hidden nodes
- 83.79% accuracy with 4 hidden nodes
- 83.50% recall with 4 hidden nodes
- 28.08% accuracy with 5 hidden nodes
- 32.98% recall with 5 hidden nodes

Rationale As you can see above, several iterations with different parameters were used in order to produce the best results from the ANN classifier, but as hidden nodes reached 5, performance plummeted.

```r
forest <- rando_forest(xTrain_noFactors, t = 10, n = 5000, d = 8)
pred_forest <- pred(forest, xTest_noFactors)
table_forest <- table(pred_forest, xTest_noFactors$atRisk)

# Best with t = 10, n = 3000, d = 5
# 86.37% accuracy
# 87.35% recall

# Best with t = 10, n = 3000, d = 8
# 78.70% accuracy
# 87.93% recall

# Best with t = 10, n = 5000, d = 5
# 85.26% accuracy
```

Table 4: Classifier Performance

Classifier	Recall	Accuracy	Comments
Random Forests	88.56	84.97	highest values from the following parameters: t = 10, n = 5000, d = 8
Decision Tree	85.15	85.92	took highest values after several iterations
Naive Bayes	84.50	84.67	
ANN	84.19	85.41	highest values from 1 hidden node, iterated up to 5 nodes
SVM: polynomial	60.32	63.67	
SVM: linear	29.75	38.61	

Table 5: Numeric Size Comparison of Clusters

Cluster 1 Size	Cluster 2 Size	Cluster 3 Size
256	339	762

```
# 87.81% recall

# Best with t = 10, n = 5000, d = 8
# 84.97% accuracy
# 88.56% recall
```

Rationale Above you will notice several iterations had to be performed with the random forest classifier in order to find a good balance between overfitting with too many trees and higher performance.

```
comparisons
```

Rationale The employee vitals dataset includes many features which are categorical and 4 which are continuous. Decision Trees and random forests can more can accurately divide the data based on categorical variables than many other classifiers. Additionally, the random forest had the highest Recall accuracy at 88.56%. I think it would be appropriate to emphasize Recall over Accuracy as health risks demand erring towards false positives over that of false negatives. Even so, the random forest model is bouyed by a descent Accuracy as well in comparison to the other classifiers.

Conclusions and Plots

```
# Build k-means cluster with scaled data
res.km <- kmeans(scale(xTest_norm[, c(-1, -ncol(xTest_norm))]), 3, nstart =  25)
# Table of cluster sizes
kable(tibble("Cluster 1 Size" = res.km$size[[1]], "Cluster 2 Size" = res.km$size[[2]], "Cluster 3 Size"
```

```
# Dimension reduction using PCA
res.pca <- prcomp(xTest_norm[, c(-1, -ncol(xTest_norm))],  scale = TRUE)
# Coordinates of individuals
ind.coord <- as.data.frame(get_pca_ind(res.pca)$coord)
# Add clusters obtained using the K-means algorithm
ind.coord$cluster <- factor(res.km$cluster)
```

```r
# Add ‘headA‘ groups from the original data set
ind.coord$atRisk <- xTest$atRisk

# Plot K-means clusters
ggscatter(
  ind.coord, x = "Dim.1", y = "Dim.2",
  color = "cluster", palette = "npg", ellipse = TRUE, ellipse.type = "convex",
  shape = "atRisk", size = 3,  legend = "right", ggtheme = theme_bw(),
  xlab = paste0("Dim 1 (", variance.percent[1], "% )" ),
  ylab = paste0("Dim 2 (", variance.percent[2], "% )" ) +
  stat_mean(aes(color = cluster), size = 8),
  title = "K-means Clusters by ‘At Risk‘ Employees")
```

```
## Error in paste0("Dim 1 (", variance.percent[1], "% )"): object 'variance.percent' not found
```

```r
## ATTRIBUTION: the kmeans clusting code above is adapted from the following site,
#https://www.datanovia.com/en/blog/k-means-clustering-visualization-in-r-step-by-step-guide/
```

K-means Clustering Plot

Clustering reveals more about how much overlap and non-linear the data is, which makes random forest a descent choice as a classifier; however, the fact that there are several discrete and continuous variables in the dataset makes the random forest stand out the most.

Continuous Variables Plots

The plot of the several continuous variables above gives another illustration of how non-linear the data tends to be. Random forests do quite well with these kinds of distributions. One interesting aspect of the data is the tendency for those within the upper or lower ends of the thresholds for each vital to be more at risk. Naturally, this makes sense and is encouraging as this pattern may aid in health-related decisions.