Appendix

Christian Pascual

5/16/2019

```
set.seed(8106)
library(tidyverse)
library(visdat)
library(caret)
mice = read_csv('./data.csv') %>%
  mutate(MouseID = 1:length(MouseID))

final.mice = mice %>%
  filter(!(MouseID %in% c(988, 989, 990))) %>%
  select_if(function(col) !any(is.na(col))) %>%
  select(DYRK1A_N:Genotype) %>%
  mutate(
    genotype = ifelse(Genotype == "Control", "Control", "DS"),
    genotype = as.factor(genotype)
    ) %>%
  select(-Genotype)

proteins = final.mice %>%
  select(DYRK1A_N:CaNA_N) %>%
  map_df(.x = ., function(col) (col - mean(col))/sd(col))

final.mice = cbind(proteins, genotype = final.mice$genotype)

proteins = final.mice %>% select(-genotype)
pca = prcomp(proteins)
reduced.pca.mice = as_tibble(cbind(pca$x[,1:27], genotype = final.mice$genotype)) %>%
  mutate(
    genotype = ifelse(genotype == 1, "Control", "DS"),
    genotype = as.factor(genotype)
    )

train.idx = createDataPartition(final.mice$genotype, p = 0.80, list = FALSE, times = 1)
train = final.mice[train.idx,]
test = final.mice[-train.idx,]
pca.train = reduced.pca.mice[train.idx,]
pca.test = reduced.pca.mice[-train.idx,]

vis_miss(mice) + theme(axis.text = element_text(size = 3))

hist.data = final.mice %>%
  gather(., key = "protein", value = "level", DYRK1A_N:CaNA_N) %>%
  mutate(Genotype = ifelse(Genotype == 0, "Control", "DS"))

ggplot(data = hist.data, aes(x  = protein, y = level, color = Genotype)) +
  geom_boxplot() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1),
        legend.position = "bottom",
        plot.title = element_text(hjust = 0.5)) +
  labs(
    title = "Histograms of protein levels by case vs control",
    x = "Protein",
    y = "Expression Level"
  )

hist.data2 = final.mice %>%
  gather(., key = "protein", value = "level", DYRK1A_N:CaNA_N) %>%
  mutate(Genotype = ifelse(Genotype == 0, "Control", "DS")) %>%
  group_by(protein, Genotype) %>%
  summarise(mean = mean(level)) %>%
  spread(., key = Genotype, value = mean) %>%
  mutate(mean_diff  = abs(DS - Control)) %>%
  arrange(-mean_diff)

ggplot(data = subset(hist.data2, mean_diff > 0.25),
       aes(x  = reorder(protein, -mean_diff), y = mean_diff)) +
  geom_point() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1),
        legend.position = "bottom",
        plot.title = element_text(hjust = 0.5)) +
  labs(
    title = "Absolute difference of protein levels by case vs control",
    x = "Protein",
    y = "Expression Level"
  )

library(corrplot)
proteins = final.mice %>% select(-Genotype)
corrplot(cor(proteins), method = "color", tl.cex = 0.5)

pca = prcomp(proteins)

# Plot out the variance explained by each PC
tib = tibble(
  PC = 1:ncol(pca$rotation),
  var = pca$sdev^2 / sum(pca$sdev^2),
  cumvar = cumsum(var)
)

ggplot(data = tib, aes(x = PC, y = cumvar)) +
  geom_line() +
  geom_point() +
  geom_hline(yintercept = 0.8, color = "blue") +
  geom_hline(yintercept = 0.9, color = "green") +
  geom_hline(yintercept = 0.95, color = "red") +
  labs(
    title = "Percentage of variance explained by principle components",
    x = "Principal Component",
    y = "% Variance Explained (cumulative)"
  ) +
  theme(plot.title = element_text(hjust = 0.5))
```

```r
pca.combined = as_tibble(cbind(pca$x, factor(final.mice$genotype))) %>%
  mutate(genotype = factor(ifelse(V69 == 1, "Control", "DS")))
ggplot(data = pca.combined, aes(x = PC1, y = PC2, color = genotype)) +
  geom_point() +
  theme(legend.position = "bottom", plot.title = element_text(hjust = 0.5)) +
  labs(
    title = "Scatterplot of PC1 vs PC2 by case vs control",
    x = "PC1",
    y = "PC2"
  )

trainControl = trainControl(method = "cv", number = 10, classProbs = TRUE)

modelFit = train(genotype ~ ., data = train,
                 method = "glmnet",
                 trControl = trainControl,
                 family = "binomial",
                 tuneGrid = expand.grid(.alpha = 1, .lambda = exp(seq(-8, -2, length.out = 100))))

pca.modelFit = train(genotype ~ ., data = pca.train,
                     method = "glmnet",
                     trControl = trainControl,
                     family = "binomial",
                     tuneGrid = expand.grid(.alpha = 1, .lambda = exp(seq(-8, -2, length.out = 100))))

preds = predict(modelFit, newdata = test)
pca.preds =  predict(pca.modelFit, newdata = pca.test)

ll.err.rate = 1 - sum(preds == test$genotype)/length(test$genotype)
ll.pca.err.rate = 1 - sum(pca.preds == test$genotype)/length(test$genotype)

trainControl = trainControl(method = "repeatedcv", classProbs = TRUE, verboseIter = TRUE)

set.seed(8106)
model.knn = train(genotype ~ ., data = train,
                  method = "knn",
                  tuneGrid = data.frame(k = seq(1, 50)),
                  trControl = trainControl,
                  preProcess = c("center", "scale"))
ggplot(model.knn)

knn.pred = predict(model.knn, newdata = test)

confusionMatrix(knn.pred, test$genotype) # 99.07% accuracy

#test error rate
knn.err.rate = 1 - sum(knn.pred == test$genotype)/length(test$genotype) #0.009302326


#pca
set.seed(8106)
model.knn.pca = train(genotype ~ ., data = pca.train,
                  method = "knn",
                  tuneGrid = data.frame(k = seq(1, 100, by = 5)),
                  trControl = trainControl)
ggplot(model.knn.pca)

knn.pred.pca = predict(model.knn.pca, newdata = pca.test)

confusionMatrix(knn.pred.pca, test$genotype) #99.07% accuracy

#pca test error rate
knn.pca.err.rate = 1 - sum(knn.pred.pca == test$genotype)/length(test$genotype) #0.009302326

gbmB.grid = expand.grid(n.trees = c(2000, 3000, 4000),
                        interaction.depth = 1:6,
                        shrinkage = c(0.001, 0.003, 0.005),
                        n.minobsinnode = 1)

# Binomial loss function

set.seed(8106)
gbmB.fit = train(genotype ~ ., data = train,
                 tuneGrid = gbmB.grid,
                 trControl = trainControl,
                 method = "gbm",
                 distribution = "bernoulli",
                 verbose = FALSE)

ggplot(gbmB.fit, highlight = TRUE)

gbmB.pred = predict(gbmB.fit, newdata = test)

#test error rate
gdmB.err.rate = 1 - sum(gbmB.pred == test$genotype)/length(test$genotype) #0.009302326


#pca
set.seed(8106)
gbmB.fit.pca = train(genotype ~ ., data = pca.train,
                 tuneGrid = gbmB.grid,
                 trControl = trainControl,
                 method = "gbm",
                 distribution = "bernoulli",
                 verbose = FALSE)

ggplot(gbmB.fit.pca, highlight = TRUE)

gbmB.pred.pca = predict(gbmB.fit.pca, newdata = pca.test)

#pca test error rate
gdmB.pca.err.rate = 1 - sum(gbmB.pred.pca == test$genotype)/length(test$genotype) #0.04186047

trainControl = trainControl(method = "repeatedcv",
                            summaryFunction = twoClassSummary,
                            classProbs = TRUE,
```

```r
                              verboseIter = TRUE)

grid = expand.grid(mfinal = (1:40), maxdepth = c(1,3,6,10,20))

bagging.fit = train(genotype ~ ., data = train,
                    method = "AdaBag",
                    trControl = trainControl,
                    tuneGrid = grid,
                    metric = "ROC")

plot(bagging.fit)


pca.bagging.fit = train(genotype ~ ., data = pca.train,
                    method = "AdaBag",
                    trControl = trainControl,
                    tuneGrid = grid,
                    metric = "ROC")


bagging.pred = predict(bagging.fit, newdata = test)
pca.bagging.pred = predict(pca.bagging.fit, newdata = pca.test)


bagging.err.rate = 1 - sum(bagging.pred == test$genotype)/length(test$genotype) #0.05116
bagging.pca.err.rate = 1 - sum(pca.bagging.pred == pca.test$genotype)/length(pca.test$genotype) #0.0604

set.seed(8106)

### radial kernel
ctr1 = trainControl(method = "cv")

svmr.grid = expand.grid(C = exp(seq(-4, 5, len=10)),
                        sigma = exp(seq(-8, -3, len=5)))

svmr.fit = train(genotype ~ ., data = train,
                  method = "svmRadial",
                  preProcess = c("center", "scale"),
                  tuneGrid = svmr.grid,
                  trControl = ctr1)

ggplot(svmr.fit, highlight = TRUE)


pca.svmr.fit = train(genotype ~ ., data = pca.train,
                  method = "svmRadial",
                  preProcess = c("center", "scale"),
                  tuneGrid = svmr.grid,
                  trControl = ctr1)



svmr.pred = predict(svmr.fit, newdata = test)
```

```
pca.svmr.pred = predict(pca.svmr.fit, newdata = pca.test)

svm.err.rate = 1 - sum(svmr.pred == test$genotype)/length(test$genotype) #0.004651
svm.pca.err.rate = 1 - sum(pca.svmr.pred == pca.test$genotype)/length(pca.test$genotype) #0.004651
```