SC1_Proj

Alessio

13 January 2020

Description of the dataset and problem

TODO: describe

```
data <- read_csv("../data/data.csv")
```

```
## Warning: Missing column names filled in: 'X33' [33]
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   diagnosis = col_character(),
##   X33 = col_character()
## )
```

```
## See spec(...) for full column specifications.
```

```
## Warning: 569 parsing failures.
## row col   expected     actual               file
##   1  -- 33 columns 32 columns '../data/data.csv'
##   2  -- 33 columns 32 columns '../data/data.csv'
##   3  -- 33 columns 32 columns '../data/data.csv'
##   4  -- 33 columns 32 columns '../data/data.csv'
##   5  -- 33 columns 32 columns '../data/data.csv'
## ... ... ......... ......... ...............
## See problems(...) for more details.
```

Dataset Preprocessing Visualisation and Exploration

```
colSums(is.na(data))
```

```
##                      id               diagnosis             radius_mean
##                       0                       0                       0
##            texture_mean          perimeter_mean               area_mean
##                       0                       0                       0
##         smoothness_mean        compactness_mean          concavity_mean
##                       0                       0                       0
##     concave points_mean          symmetry_mean  fractal_dimension_mean
##                       0                       0                       0
##               radius_se              texture_se            perimeter_se
##                       0                       0                       0
##                 area_se           smoothness_se          compactness_se
##                       0                       0                       0
##            concavity_se       concave points_se             symmetry_se
##                       0                       0                       0
##    fractal_dimension_se            radius_worst           texture_worst
##                       0                       0                       0
##         perimeter_worst              area_worst        smoothness_worst
```

```
##                        0                         0                        0
##       compactness_worst         concavity_worst    concave points_worst
##                        0                         0                        0
##          symmetry_worst fractal_dimension_worst                      X33
##                        0                         0                      569
```

```
data %<>% mutate_at(vars(diagnosis), factor)
```

```
train <- data %>% sample_frac(0.8)
test <- anti_join(data,train, by='id')
```

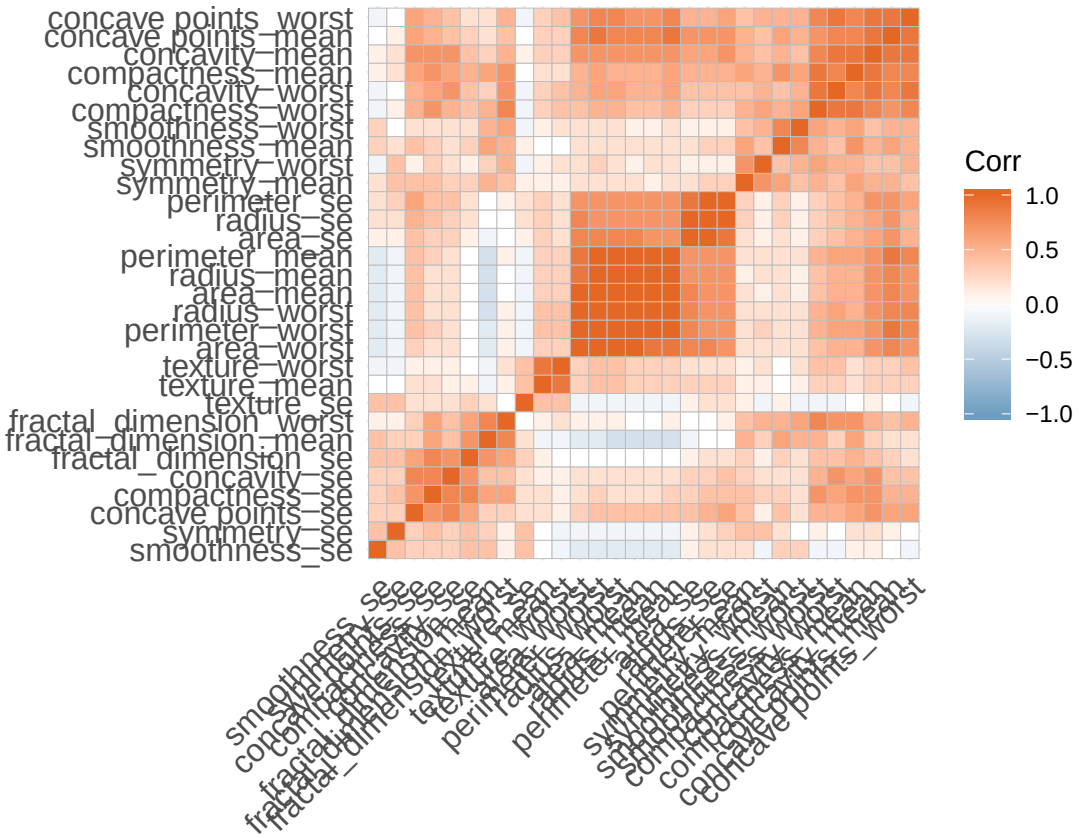
```
data %<>%
  dplyr::select(-c(id, X33))
train %<>%
  dplyr::select(-c(id, X33))
test %<>%
  dplyr::select(-c(id, X33))
```

```
training_data <- train[2:dim(train)[2]]
training_classes <- train[1]

test_data <- test[2:dim(test)[2]]
test_classes <- test[1]
```

```
sum(is.na(data))
```

```
## [1] 0
```

```
corr <- data[,-1] %>%
          cor() %>%
           round(1)
ggcorrplot(corr,
           hc.order = TRUE,
           colors = c("#6D9EC1", "white", "#E46726"),
           ggtheme = ggplot2::theme_minimal)
```



- Class Frequencies
- Density
- box plots

Dimensionality Reduction and Feature Selection

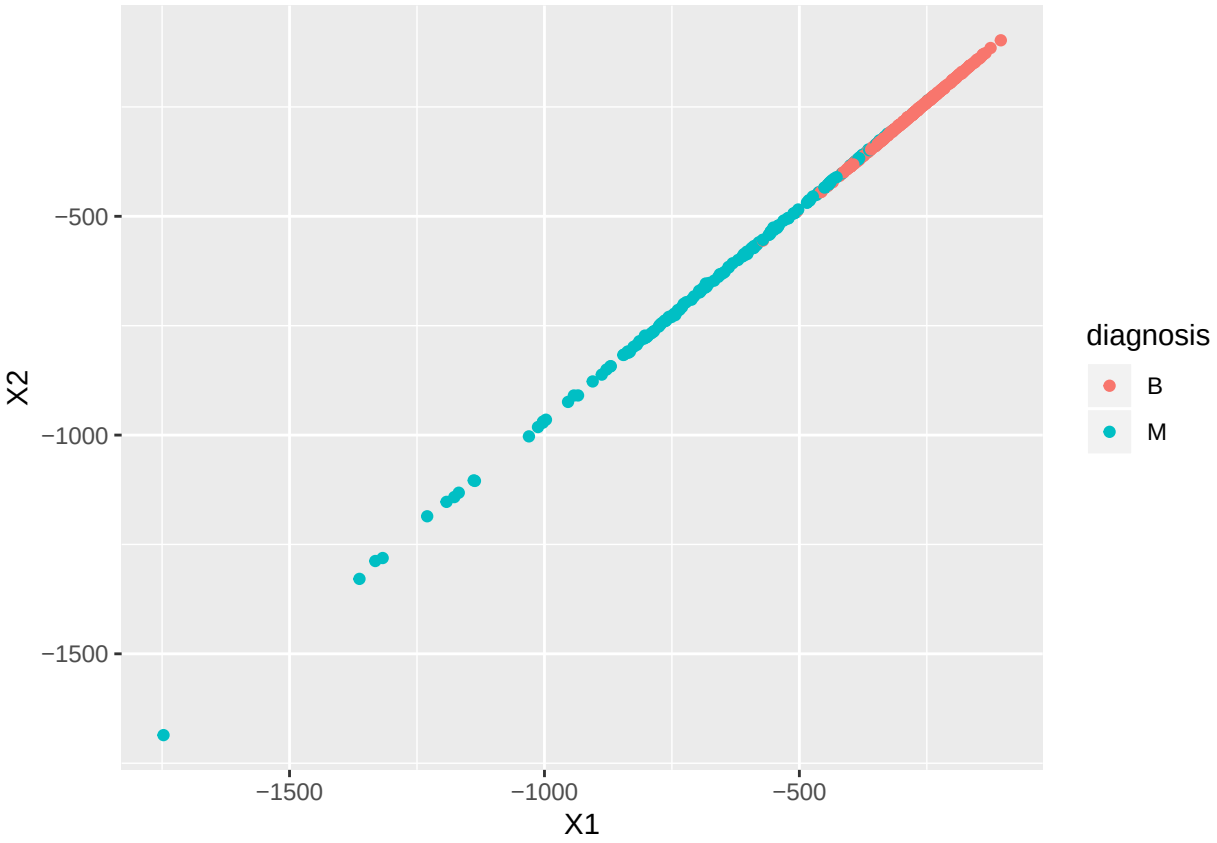
- PCA Code

```
normalise_z <-  function(X){
  mean_cols <- colMeans(X)
  sd_cols <- apply(X, 2, sd)
  mean_normalised_X <- t(apply(X, 1, function(x){x - mean_cols}))
  normalised_X <- t(apply(mean_normalised_X, 1, function(x){x / sd_cols}))
  return(normalised_X)
}


pca <- function(X, number_components_keep) {
  normalised_X <- normalise_z(X)

  corr_mat <- t(normalised_X) %*% normalised_X

  eigenvectors <- eigen(corr_mat, symmetric=TRUE)$vectors

  reduced_data <-  X %*% eigenvectors[,1:number_components_keep]
  relevant_eigs <-  eigenvectors[,1:number_components_keep]
  returnds <- list(reduced_data, relevant_eigs)
```

```
  names(returnds) <- c("reduced_data", "reduction_matrix")
  return(returnds)
}
```

Apply to dataset

```
pca_result <- pca(as.matrix(training_data), 2)
reduced_training_data <- data.frame(cbind(pca_result$reduced_data, training_classes))

ggplot(data=reduced_training_data, aes(x=X1, y=X2)) + geom_point(aes(colour=diagnosis))
```



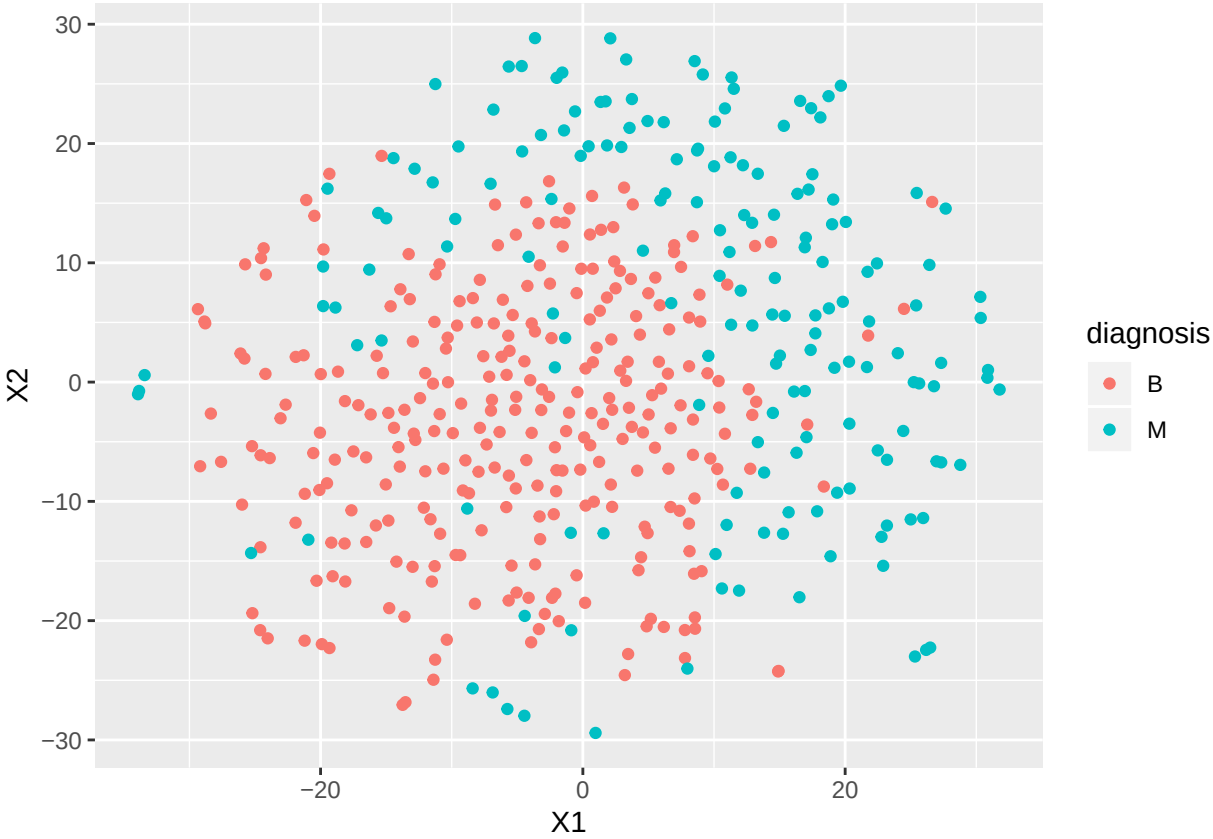
tSNE TODO: try different perplexity parameters

```
reduced_training_data <- tsne::tsne(training_data)
```

```
## sigma summary: Min. : 0.298811060497677 |1st Qu. : 0.508190776389993 |Median : 0.544599500039341 |Mea
## Epoch: Iteration #100 error is: 19.294534890338
## Epoch: Iteration #200 error is: 1.44411660626388
## Epoch: Iteration #300 error is: 1.42258724003965
## Epoch: Iteration #400 error is: 1.41859021960501
## Epoch: Iteration #500 error is: 1.41238411938403
## Epoch: Iteration #600 error is: 1.40975349010736
## Epoch: Iteration #700 error is: 1.40931369048512
## Epoch: Iteration #800 error is: 1.40928976264231
```

```
## Epoch: Iteration #900 error is: 1.40928206021092
## Epoch: Iteration #1000 error is: 1.40927924695282
```

```
reduced_training_data <- data.frame(cbind(reduced_training_data, training_classes))
ggplot(data=reduced_training_data, aes(x=X1, y=X2)) + geom_point(aes(colour=diagnosis))
```



- Correlation Feature Selection
- LDA

Classification

- SVM

TODO: Mathematical description

TODO: Basic Code describing implementation

TODO: Application

- Naive Bayes
- Logistic Regression
- Lasso

Conclusion

- Evaluation of results
- Discuss outliers