Final Project Report

Olivia Fan, Alicia Gong

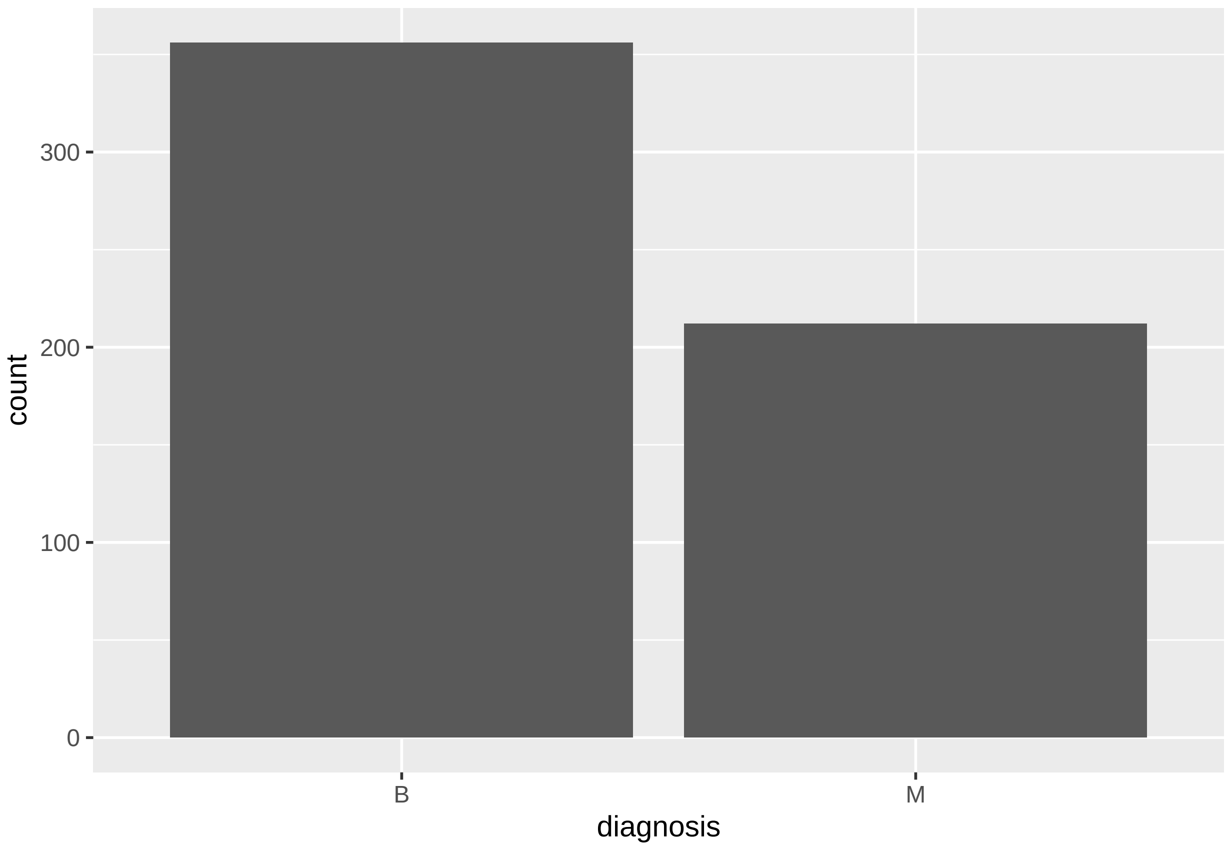
07 December, 2022

Data Processing

In order to fit SVM on the data, we encode the `diagnosis` variable into a factor variable with level 1 and -1:

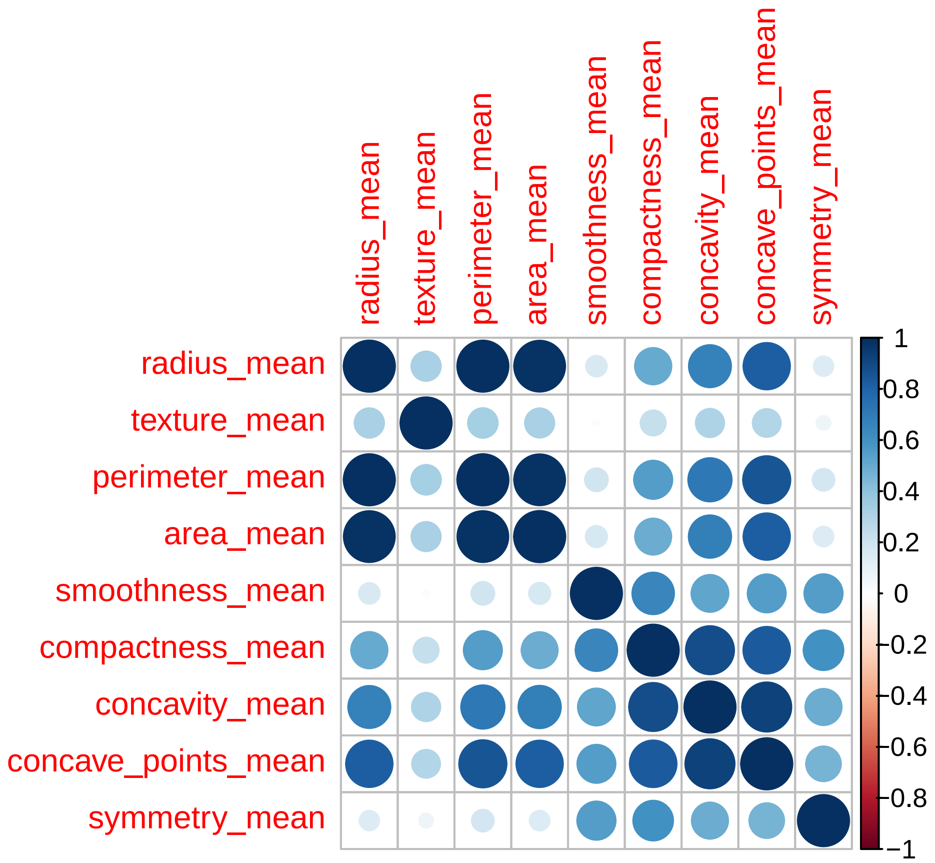
We partition the data into training and testing sets using a 70-30 percentage split(70% of the original data as the training set, and 30% as the testing set):

EDA



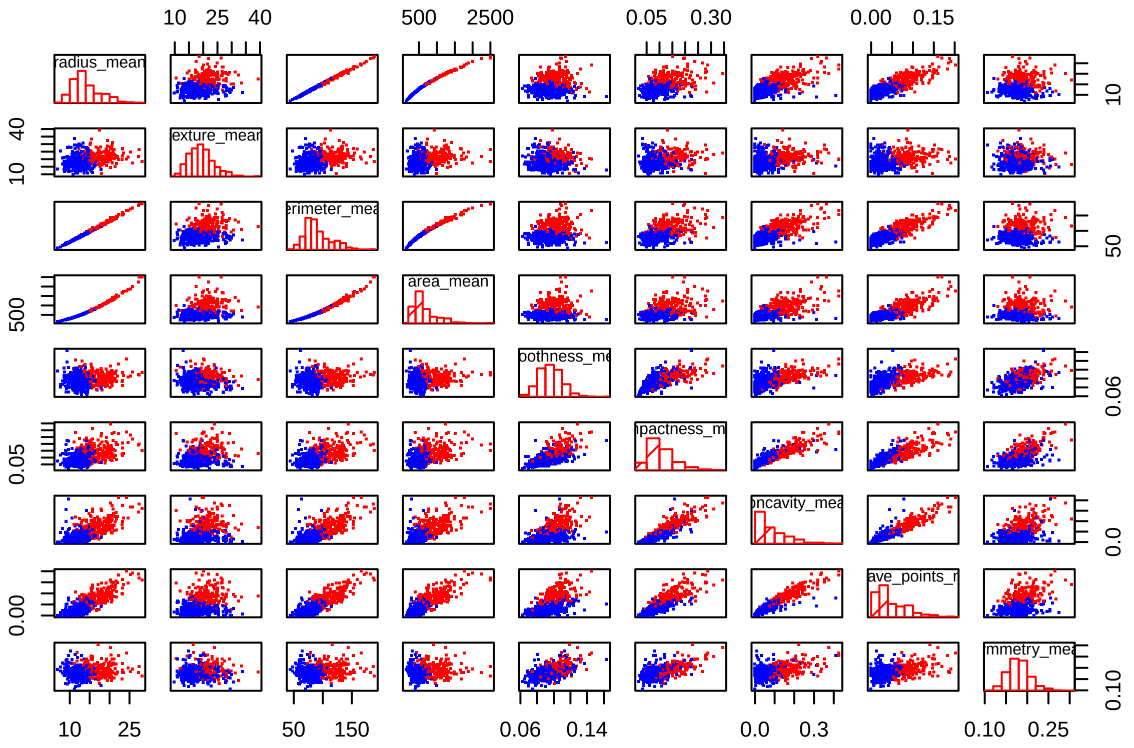
The bar plot shows that there is a larger number of benign than malignant cancer.

We divide the data into 3 categories according to their features.



Major observations:

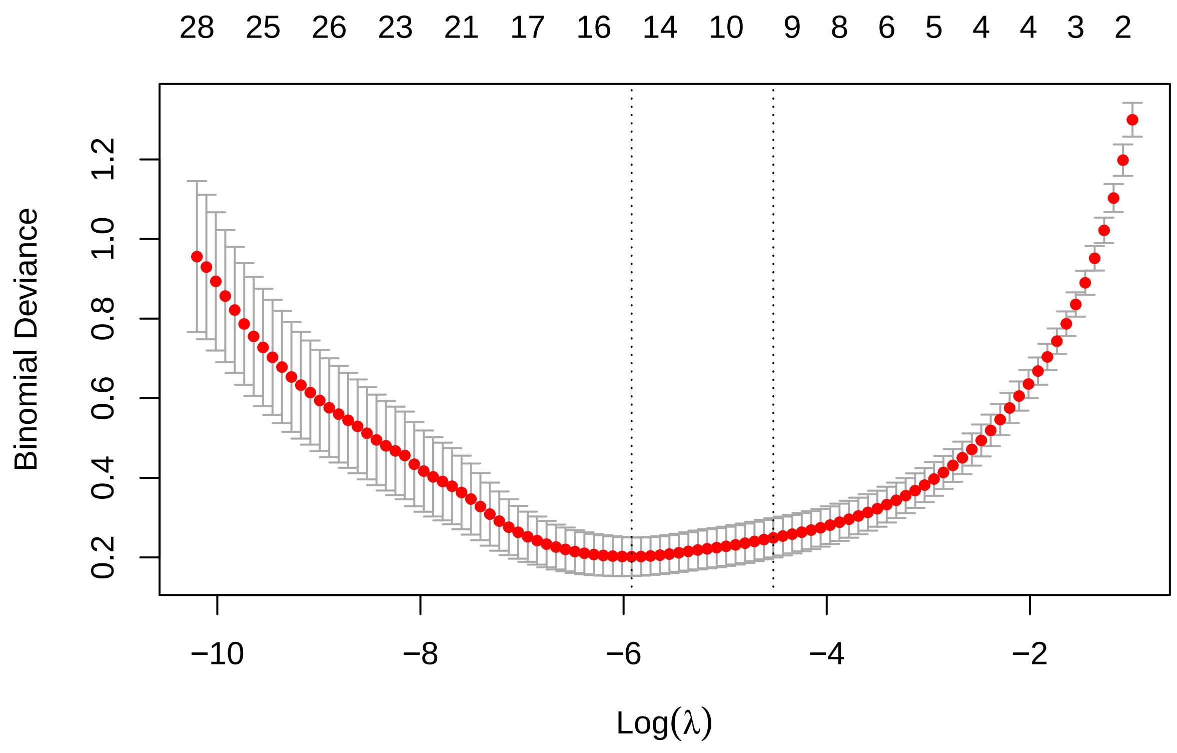
- Radius_mean, perimeter_mean, and area_mean are highly correlated.
- Compactness_mean, concavity_mean and concave_points_mean are highly correlated.



Methodology

SVM

Model Selection (Lasso penalized logistic regression)



```
## [1] 0.002682998
```

```
## 31 x 1 sparse Matrix of class "dgCMatrix"
##                                  s0
## (Intercept)             -28.77675317
## radius_mean               .
## texture_mean              0.05593184
## perimeter_mean            .
## area_mean                 .
## smoothness_mean           .
## compactness_mean          .
## concavity_mean            .
## concave_points_mean      26.70558695
## symmetry_mean             .
## fractal_dimension_mean    .
## radius_se                 4.68352247
## texture_se               -0.53071651
## perimeter_se              .
## area_se                   0.04732961
## smoothness_se            88.30404283
## compactness_se          -42.98049312
## concavity_se              .
## concave_points_se         .
## symmetry_se               .
## fractal_dimension_se    -85.32165947
## radius_worst              0.58890672
## texture_worst             0.22994413
## perimeter_worst           .
## area_worst                .
## smoothness_worst         17.82352005
## compactness_worst         .
## concavity_worst           4.35034427
## concave_points_worst     21.35593118
## symmetry_worst            7.99935194
## fractal_dimension_worst   .
```

Linear Kernel SVM

We use the predictors selected by the LASSO penalized logistic regression as predictors for the support vector machine model:

If two predictors have high correlation, only use one of them:

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost
##   0.1
##
## - best performance: 0.03025641
##
## - Detailed performance results:
##    cost      error dispersion
## 1 1e-03 0.10807692 0.03865124
## 2 1e-02 0.05038462 0.02042181
## 3 1e-01 0.03025641 0.02307059
## 4 1e+00 0.03775641 0.02700681
## 5 5e+00 0.03775641 0.02429977
## 6 1e+01 0.03525641 0.02112037
## 7 1e+02 0.03275641 0.02904010
```

```
##
## Call:
## best.tune(METHOD = svm, train.x = diagnosis_binary ~ concavity_mean +
##     concave_points_mean + radius_se + texture_se + smoothness_se +
##     compactness_se + fractal_dimension_se + radius_worst + texture_worst +
##     smoothness_worst + concavity_worst + concave_points_worst + symmetry_worst +
##     fractal_dimension_worst, data = cancer_train, ranges = list(cost = c(0.001,
##     0.01, 0.1, 1, 5, 10, 100)), kernel = "linear")
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  0.1
##
## Number of Support Vectors:  62
##
##  ( 30 32 )
##
##
## Number of Classes:  2
##
## Levels:
##  -1 1
```

```
##        truth
## predict -1  1
##      -1 97  2
##      1   1 71
```

```
## [1] 0.01754386
```

The misclassification rate is 0.0467.

Radial Kernel SVM

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost gamma
##     1   0.5
##
## - best performance: 0.05051282
##
## - Detailed performance results:
```

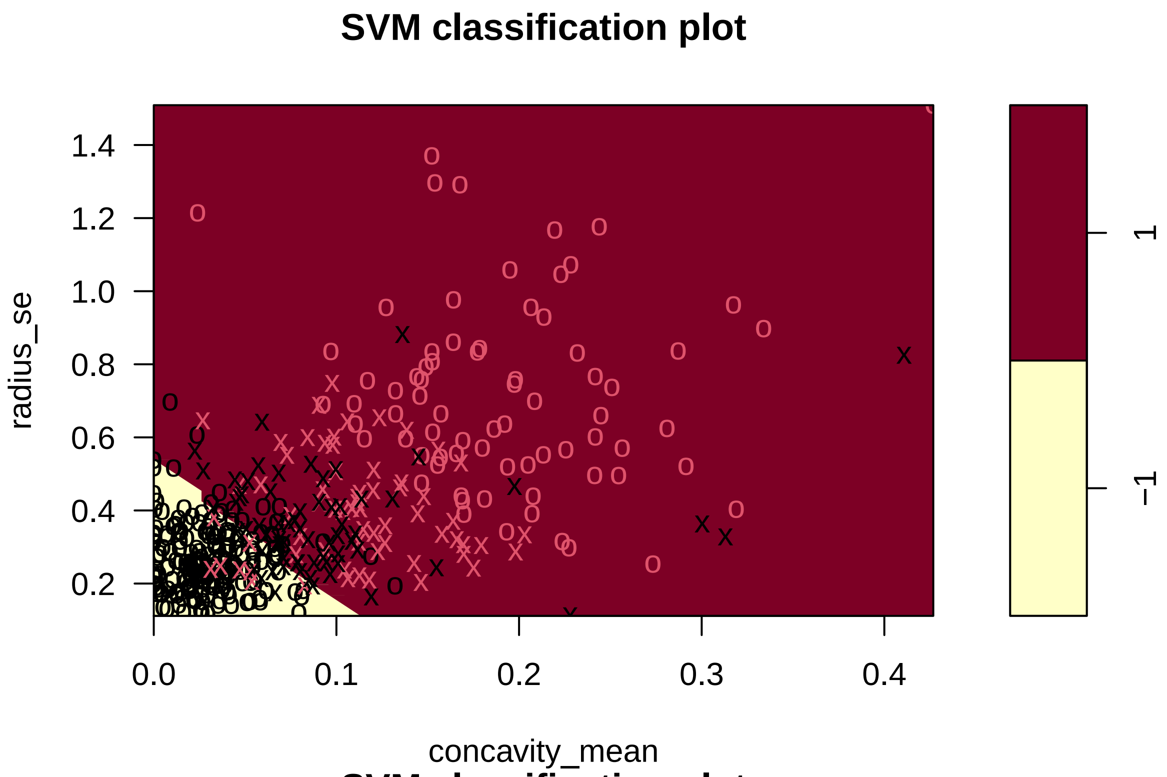
```
##     cost gamma      error dispersion
## 1  1e-01   0.5 0.35019231 0.06877558
## 2  1e+00   0.5 0.05051282 0.03765876
## 3  1e+01   0.5 0.05294872 0.03007618
## 4  1e+02   0.5 0.05294872 0.03007618
## 5  1e+03   0.5 0.05294872 0.03007618
## 6  1e-01   1.0 0.35019231 0.06877558
## 7  1e+00   1.0 0.16628205 0.07124087
## 8  1e+01   1.0 0.14115385 0.05604713
## 9  1e+02   1.0 0.14115385 0.05604713
## 10 1e+03   1.0 0.14115385 0.05604713
## 11 1e-01   2.0 0.35019231 0.06877558
## 12 1e+00   2.0 0.34012821 0.05872625
## 13 1e+01   2.0 0.33506410 0.05581714
## 14 1e+02   2.0 0.33506410 0.05581714
## 15 1e+03   2.0 0.33506410 0.05581714
## 16 1e-01   3.0 0.35019231 0.06877558
## 17 1e+00   3.0 0.35019231 0.06877558
## 18 1e+01   3.0 0.35019231 0.06877558
## 19 1e+02   3.0 0.35019231 0.06877558
## 20 1e+03   3.0 0.35019231 0.06877558
## 21 1e-01   4.0 0.35019231 0.06877558
## 22 1e+00   4.0 0.35019231 0.06877558
## 23 1e+01   4.0 0.35019231 0.06877558
## 24 1e+02   4.0 0.35019231 0.06877558
## 25 1e+03   4.0 0.35019231 0.06877558
```

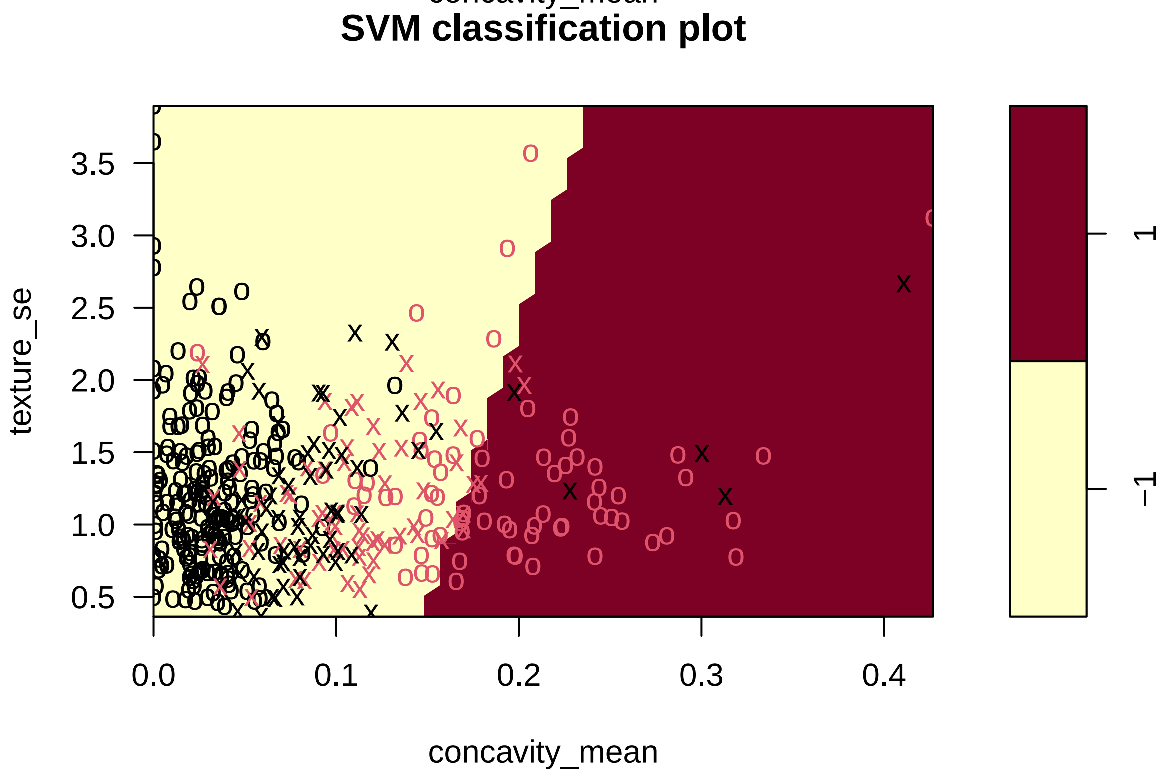
```
##
## Call:
## best.tune(METHOD = svm, train.x = diagnosis_binary ~ concavity_mean +
##     concave_points_mean + radius_se + texture_se + smoothness_se +
##     compactness_se + radius_worst + texture_worst + smoothness_worst +
##     concavity_worst + concave_points_worst + symmetry_worst + fractal_dimension_worst,
##     data = cancer_train, ranges = list(cost = c(0.1, 1, 10, 100,
##         1000), gamma = c(0.5, 1, 2, 3, 4)), kernel = "radial")
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  1
##
## Number of Support Vectors:  269
##
##  ( 127 142 )
##
##
## Number of Classes:  2
##
## Levels:
##  -1 1
```

```
##        truth
## predict -1  1
##      -1 94  3
##       1   4 70
```

SVM Visualization

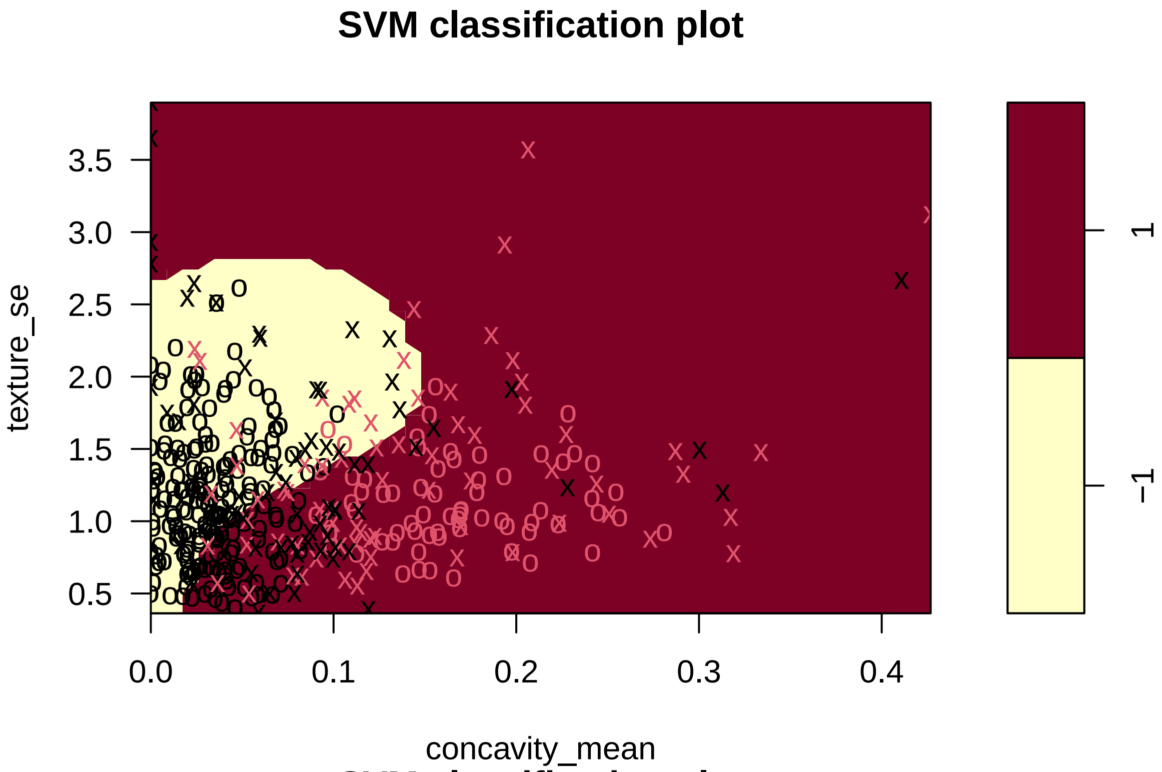
Linear

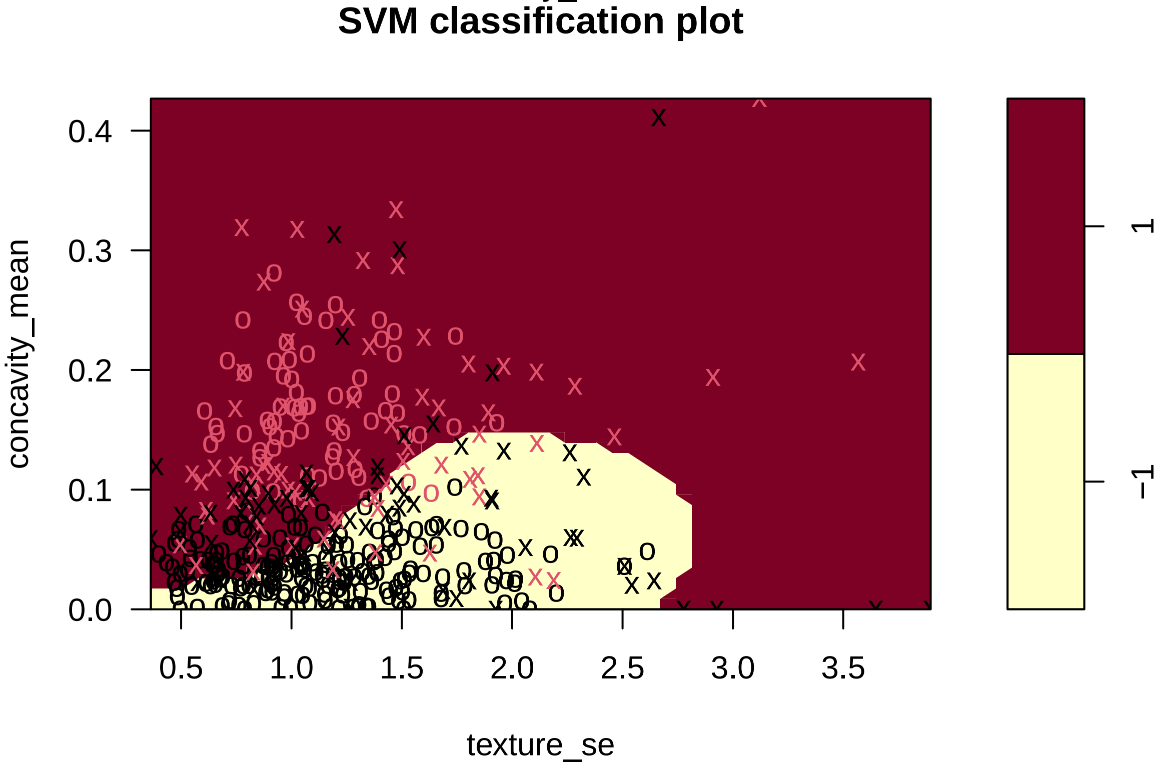




##

Radial





Random Forest

```
##                                MeanDecreaseGini
## radius_mean                    8.5616407
## texture_mean                   3.0559818
## perimeter_mean                 5.9008019
## area_mean                      9.8692050
## smoothness_mean                1.1478325
## compactness_mean               1.6773967
## concavity_mean                10.1532690
## concave_points_mean           18.7904551
## symmetry_mean                  0.6545792
## fractal_dimension_mean         0.7849650
## radius_se                      2.4824456
## texture_se                     0.8773616
## perimeter_se                   2.8646560
## area_se                        6.6899822
## smoothness_se                  1.1900223
## compactness_se                 0.9471109
## concavity_se                   1.1550698
## concave_points_se              1.0486457
## symmetry_se                    0.8049179
## fractal_dimension_se           1.1728247
## radius_worst                  18.0287062
## texture_worst                  3.6374436
## perimeter_worst               20.0049651
## area_worst                    17.7615481
## smoothness_worst               2.9276817
## compactness_worst              3.0046846
## concavity_worst                6.6002452
## concave_points_worst          25.2111650
## symmetry_worst                 1.8577452
## fractal_dimension_worst        1.4519769
```

mean:

```
##
## Call:
##  randomForest(formula = diagnosis_binary ~ radius_mean + perimeter_mean +      area_mean + concavity_
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 7.3%
## Confusion matrix:
##     -1   1 class.error
## -1 246  12  0.04651163
## 1   17 122  0.12230216
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction -1  1
##         -1 92  7
##         1   6 66
##
##                Accuracy : 0.924
##                  95% CI : (0.8735, 0.9589)
##     No Information Rate : 0.5731
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8444
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.9388
##             Specificity : 0.9041
##          Pos Pred Value : 0.9293
##          Neg Pred Value : 0.9167
##              Prevalence : 0.5731
##          Detection Rate : 0.5380
##    Detection Prevalence : 0.5789
##       Balanced Accuracy : 0.9214
##
##        'Positive' Class : -1
##
```

```
##
## Call:
##  randomForest(formula = diagnosis_binary ~ radius_worst + perimeter_worst +      area_worst + concav
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 5.79%
## Confusion matrix:
##     -1   1 class.error
## -1 248  10  0.03875969
## 1   13 126  0.09352518
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction -1  1
##         -1 94  7
##         1   4 66
##
##                Accuracy : 0.9357
##                  95% CI : (0.8878, 0.9675)
##     No Information Rate : 0.5731
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8678
##
##  Mcnemar's Test P-Value : 0.5465
##
##             Sensitivity : 0.9592
##             Specificity : 0.9041
##          Pos Pred Value : 0.9307
##          Neg Pred Value : 0.9429
##              Prevalence : 0.5731
##          Detection Rate : 0.5497
##    Detection Prevalence : 0.5906
##       Balanced Accuracy : 0.9316
##
##        'Positive' Class : -1
##
```