Mathematical Methods for Artificial Intelligence Lab 4

Vytautas Kraujalis

2021-11-20

Contents

1 Gaussian Process

1.1 Reading Data

```r
set.seed(123)

data_original <- read.csv("satellite_train.csv")
```

1.2 Required packages

```
library(SmartEDA)
library(dplyr)
library(ggplot2)
library(caret)

# Function for 6 class accuracy from confusion matrix
classAcc <- function(confusionMatrix) {
  class1 <- round(confusionMatrix$table[1, 1] / sum(confusionMatrix$table[, 1]) * 100, 1)
  class2 <- round(confusionMatrix$table[2, 2] / sum(confusionMatrix$table[, 2]) * 100, 1)
  class3 <- round(confusionMatrix$table[3, 3] / sum(confusionMatrix$table[, 3]) * 100, 1)
  class4 <- round(confusionMatrix$table[4, 4] / sum(confusionMatrix$table[, 4]) * 100, 1)
  class5 <- round(confusionMatrix$table[5, 5] / sum(confusionMatrix$table[, 5]) * 100, 1)
  class6 <- round(confusionMatrix$table[6, 6] / sum(confusionMatrix$table[, 6]) * 100, 1)
  acc <- c(class1, class2, class3, class4, class5, class6 )
  names(acc) <- colnames(confusionMatrix$table)
  return(acc)
}
```

1.3 Parallel processing

```
library(parallel)
no_cores <- detectCores() - 1
library(doParallel)
cl <- makePSOCKcluster(no_cores)
registerDoParallel(cl)
```

1.4 EDA, first look at the dataset

```
ExpData(data_original,type=1)
```

```
##                                              Descriptions     Value
## 1                                      Sample size (nrow)      4435
## 2                                 No. of variables (ncol)        37
## 3                       No. of numeric/interger variables        37
## 4                                 No. of factor variables         0
## 5                                   No. of text variables         0
## 6                                No. of logical variables         0
## 7                             No. of identifier variables         0
## 8                                   No. of date variables         0
## 9                No. of zero variance variables (uniform)         0
## 10              %. of variables having complete cases 100% (37)
## 11   %. of variables having >0% and <50% missing cases    0% (0)
## 12 %. of variables having >=50% and <90% missing cases    0% (0)
## 13          %. of variables having >=90% missing cases    0% (0)
```

We have a dataset of 4435 observations with 37 variables, all of the variables are numerical. All variables have no missing values. Let's change the response variable to factor type.

```
data <- data_original %>%
   mutate(V37 = as.factor(V37)) %>%
   rename(Target = V37)
```

```
data <- data %>%
   mutate(
      Target = as.factor(case_when(
         Target == "1" ~ "Red Soil",
         Target == "2" ~ "Cotton Crop",
         Target == "3" ~ "Grey Soil",
         Target == "4" ~ "Damp Grey Soil",
         Target == "5" ~ "Soil With Vegetation Stubble",
         Target == "7" ~ "Very Damp Grey Soil",
         TRUE ~ "ERROR"
      ))
   )
```

Let's look at the target variable frequencies

```
data %>%
   group_by(Target) %>%
   summarise(n = n()) %>%
   mutate(n_prop = round(n / sum(n) * 100, 2))
```

```
## # A tibble: 6 x 3
##   Target                           n n_prop
##   <fct>                        <int>  <dbl>
## 1 Cotton Crop                    479  10.8
## 2 Damp Grey Soil                 415   9.36
## 3 Grey Soil                      961  21.7
## 4 Red Soil                      1072  24.2
## 5 Soil With Vegetation Stubble   470  10.6
## 6 Very Damp Grey Soil           1038  23.4
```

Our target variable has 6 classes, the smallest class has 415 (9.4%) observations while the biggest class has 1072 (24.2%) observations.

Let's look at descriptive statistics of each variable:

```
ExpNumStat(data,by ="A",round= 2, gp = "Target") %>%
  select(Vname, min, max, mean, median, SD)
```

```
##    Vname min max  mean median    SD
## 1     V1  40 104 69.47     68 13.65
## 10   V10  27 130 83.13     85 22.81
## 11   V11  50 145 98.97    101 16.68
## 12   V12  29 157 82.41     81 18.84
## 13   V13  40 102 69.37     68 13.66
## 14   V14  27 131 83.73     87 22.79
## 15   V15  53 145 99.41    101 16.69
## 16   V16  33 151 82.65     81 18.80
## 17   V17  40 104 69.13     68 13.56
```

```
## 18   V18  27 130 83.43     85 22.82
## 19   V19  56 139 99.24    101 16.73
## 2     V2  27 137 83.86     87 22.73
## 20   V20  34 157 82.62     81 18.84
## 21   V21  39 104 68.92     67 13.47
## 22   V22  27 130 83.14     85 22.83
## 23   V23  50 140 99.00    100 16.72
## 24   V24  29 154 82.48     81 18.92
## 25   V25  40 104 69.25     68 13.65
## 26   V26  27 131 83.67     85 22.77
## 27   V27  53 140 99.32    101 16.70
## 28   V28  34 154 82.67     81 18.88
## 29   V29  39 104 69.03     68 13.53
## 3     V3  56 140 99.32    101 16.67
## 30   V30  27 128 83.43     85 22.81
## 31   V31  50 145 99.18    101 16.74
## 32   V32  29 157 82.64     81 18.94
## 33   V33  40 104 68.80     67 13.44
## 34   V34  27 130 83.15     85 22.76
## 35   V35  50 145 99.06    100 16.66
## 36   V36  29 157 82.58     81 18.90
## 4     V4  33 154 82.56     83 18.70
## 5     V5  40 102 69.21     68 13.55
## 6     V6  27 137 83.50     85 22.81
## 7     V7  50 145 99.17    101 16.63
## 8     V8  29 157 82.48     81 18.71
## 9     V9  40 104 68.96     67 13.50
```

Nothing seems unordinary.

We should look at the correlation between variables

```r
# Correlation

corr_simple <- function(df,sig=0.5){
  corr <- cor(df)
  #prepare to drop duplicates and correlations of 1
  corr[lower.tri(corr,diag=TRUE)] <- NA
  #drop perfect correlations
  corr[corr == 1] <- NA
  #turn into a 3-column table
  corr <- as.data.frame(as.table(corr))
  #remove the NA values from above
  corr <- na.omit(corr)
  #select significant values
  corr <- subset(corr, abs(Freq) > sig)
  #sort by highest correlation
  corr <- corr[order(-abs(corr$Freq)),]
  return(corr)
}

correlation_matrix = cor(data %>% select(-Target))

length(findCorrelation(correlation_matrix, cutoff = 0.99))
```

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

```r
length(findCorrelation(correlation_matrix, cutoff = 0.95))
```

```
## [1] 11
```

```r
length(findCorrelation(correlation_matrix, cutoff = 0.9))
```

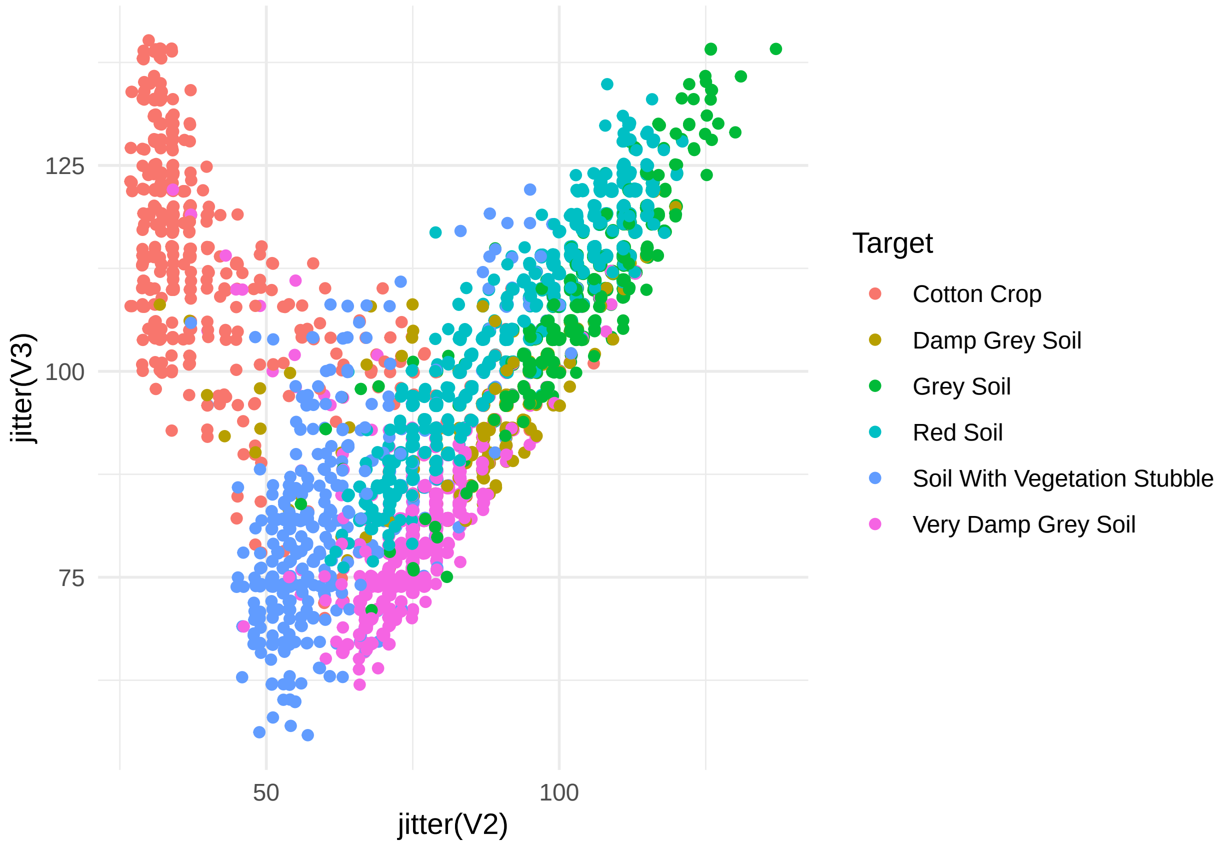
```
## [1] 23
```

We have 0 variables with higher than .99 correlation. We have 11 variables with higher than .95 correlation. We have 23 variables with higher than .9 correlation.

We'll look through random scatter plots and see how our target variable is seperated across. We'll save some interesting combinations for later. P.S. we'll use jitter() function to overcome observation overlap.

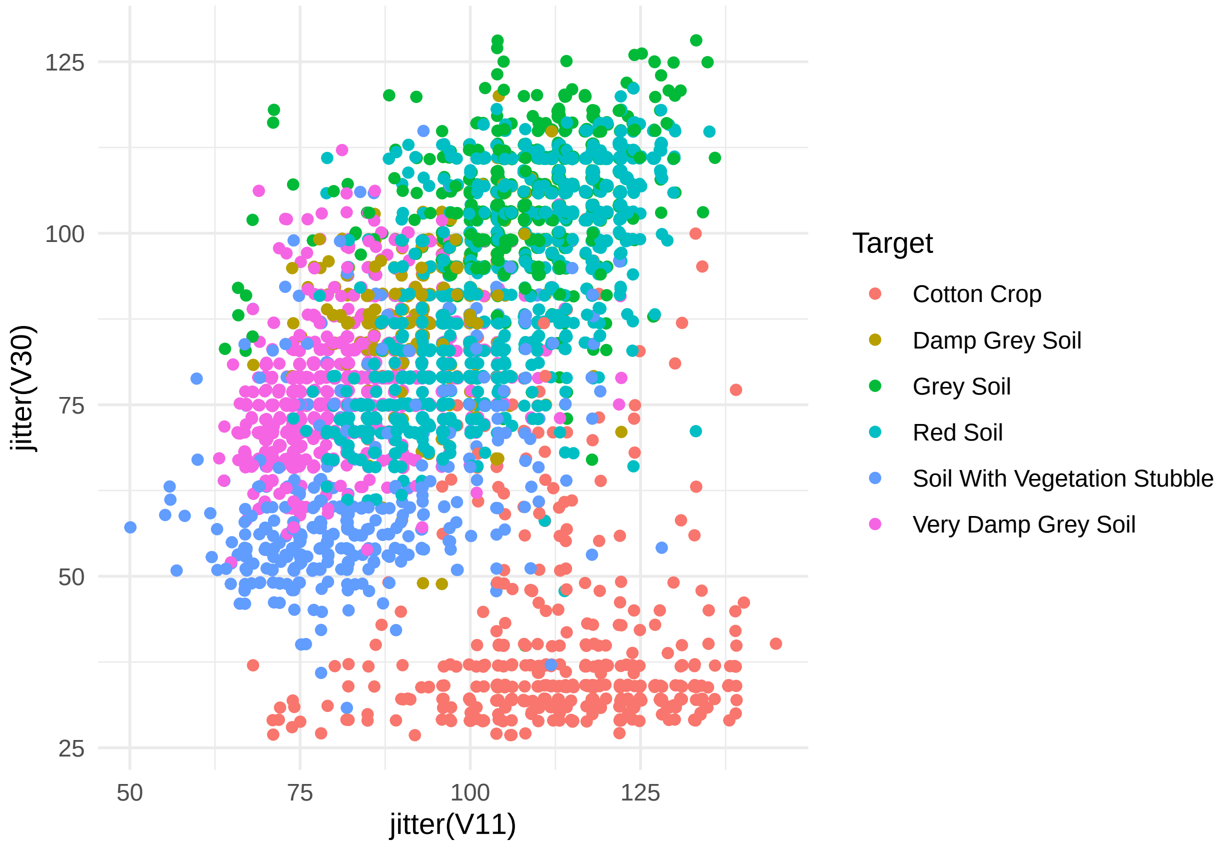
```r
# data_plot <- data %>%
#    select(sample(0:36, 1), sample(0:36, 1), Target)
# data_plot_colnames <- colnames(data_plot)
# colnames(data_plot) <- c("V_1", "V_2", "Target")
# data_plot %>%
#    ggplot(aes(x = jitter(V_1), y = jitter(V_2), color = Target)) +
#    geom_point() +
#    xlab(data_plot_colnames[1]) +
#    ylab(data_plot_colnames[2])

#combinations <- c("V2 - V3", "V30 - V11", "V7 - V3", "V3 - V24", "V31 - V32")
```

```r
data %>%
   ggplot(aes(x = jitter(V2), y = jitter(V3), color = Target)) +
   geom_point() +
   theme_minimal()
```



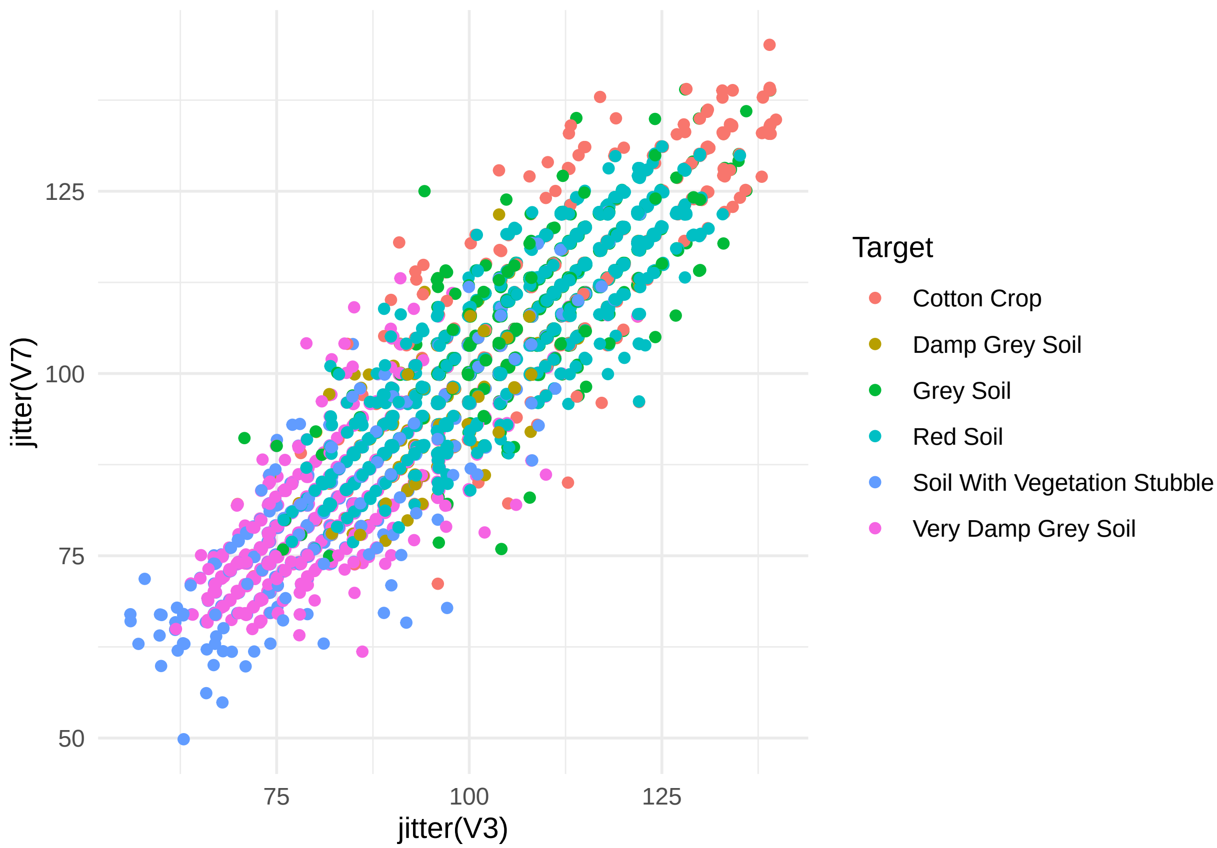
Combination of V2 and V3 features shows quite significant separation of "Cotton Crop" group, while "Damp Grey Soil" has almost no separation. All other classes could be visually separated.

```
data %>%
  ggplot(aes(x = jitter(V11), y = jitter(V30), color = Target)) +
  geom_point() +
  theme_minimal()
```



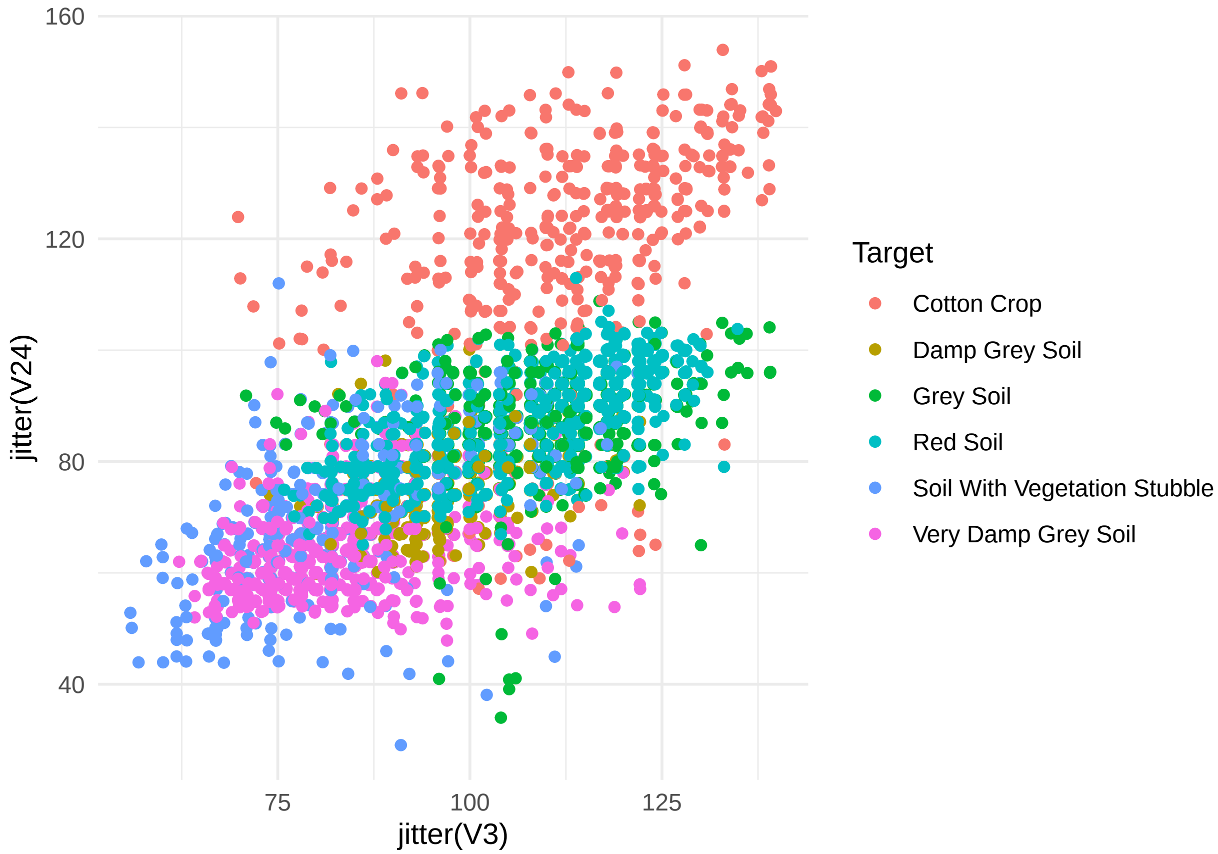
once again, the “Cotton Crop” group seems separatable quite good, but now there’s a mix of “Red Soil”, “Grey Soil” and “Damp Grey Soil” all in one cluster, which could be hard to distinguish.

```
data %>%
  ggplot(aes(x = jitter(V3), y = jitter(V7), color = Target)) +
  geom_point() +
  theme_minimal()
```



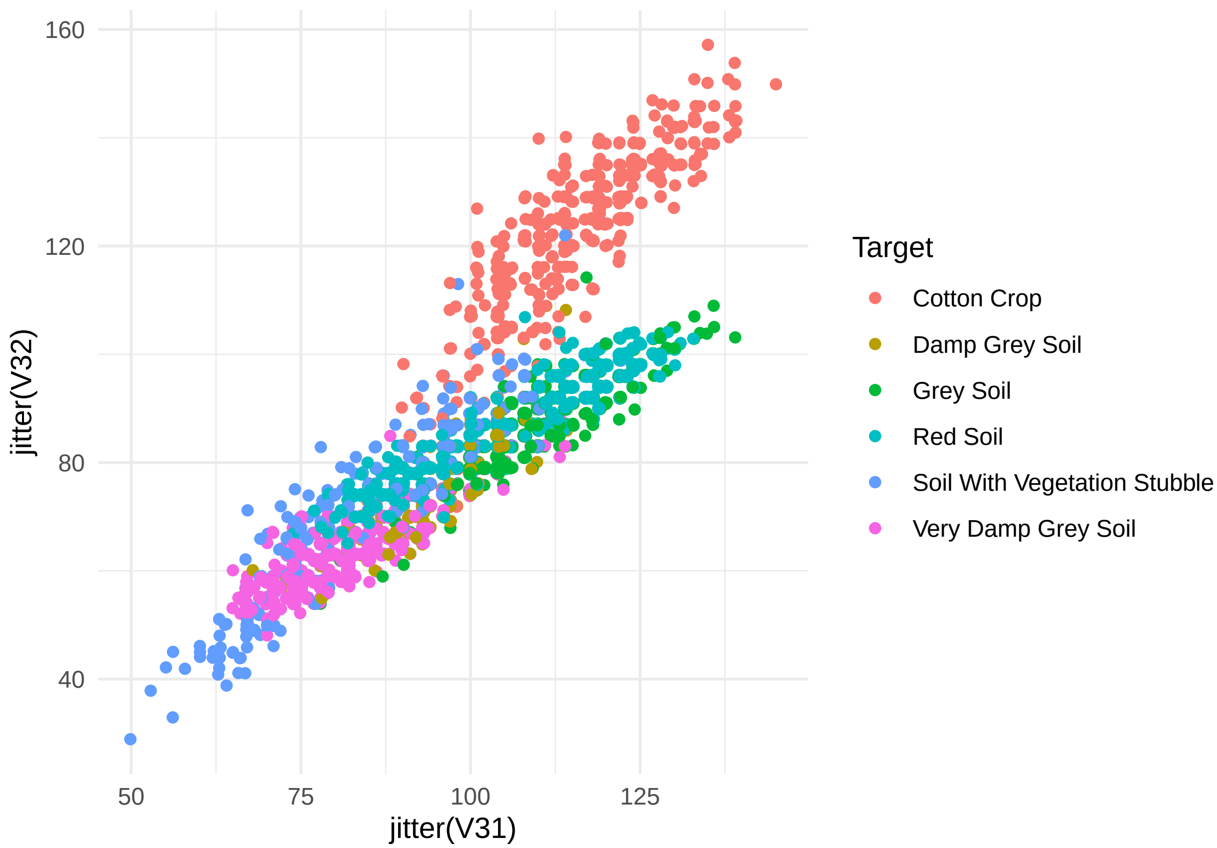
Looking at the combination of V3 and V7 features, we see a clear linear combination, but the target variable is mixed all over the place.

```
data %>%
  ggplot(aes(x = jitter(V3), y = jitter(V24), color = Target)) +
  geom_point() +
  theme_minimal()
```



Using a combination of V3 and V24 features we can separate "Cotton Crop" easily, but the rest of the classes won't be separated so easily.

```
data %>%
  ggplot(aes(x = jitter(V31), y = jitter(V32), color = Target)) +
  geom_point() +
  theme_minimal()
```



Similar results as the plot before, but now the other classes could be separatable a bit better, only the "Damp Grey Soil" and "Soil With Vegetation Stubble" classes are not separatable that good.

It seems that our model could have a problem at detecting "Damp Grey Soil" class.

1.5 Fitting models

Tried fitting Linear Gaussian Process and Gaussian Process with Polynomial Kernel but both methods took too long to compute...

1.5.1 Gaussian Process - Variational Bayesian Multinomial Probit Regression

```
fitControl <- trainControl(
  method = "cv",
  number = 2,
  classProbs = TRUE,
  savePredictions="all",
  verboseIter = TRUE)

gp.vbmp.fit <- train(Target ~ ., data = data %>%
                   mutate(Target = factor(Target, labels = make.names(levels(Target)))),
                 method = "vbmpRadial",
                 trControl = fitControl,
                 preProcess=c("center", "scale","pca"))
```

```
## Aggregating results
## Fitting final model on full training set
```

```r
resamp_vbmp = gp.vbmp.fit$pred[gp.vbmp.fit$pred$estimateTheta == gp.vbmp.fit$bestTune[1,1],]
confusion_matrix <- confusionMatrix(resamp_vbmp$pred, resamp_vbmp$obs)

confusion_matrix
```

```
## Confusion Matrix and Statistics
##
##                                Reference
## Prediction                       Cotton.Crop Damp.Grey.Soil Grey.Soil Red.Soil
##   Cotton.Crop                            461              5         2        2
##   Damp.Grey.Soil                           1            261        40        7
##   Grey.Soil                                3             72       889       16
##   Red.Soil                                 0              5         7     1037
##   Soil.With.Vegetation.Stubble            11             10         3       10
##   Very.Damp.Grey.Soil                      3             62        20        0
##                                Reference
## Prediction                       Soil.With.Vegetation.Stubble Very.Damp.Grey.Soil
##   Cotton.Crop                                               9                   5
##   Damp.Grey.Soil                                            4                  73
##   Grey.Soil                                                 4                  31
##   Red.Soil                                                 21                   0
##   Soil.With.Vegetation.Stubble                            394                  30
##   Very.Damp.Grey.Soil                                      38                 899
##
## Overall Statistics
##
##                Accuracy : 0.8886
##                  95% CI : (0.879, 0.8977)
##     No Information Rate : 0.2417
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.862
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Cotton.Crop Class: Damp.Grey.Soil Class: Grey.Soil
## Sensitivity                      0.9624               0.62892           0.9251
## Specificity                      0.9942               0.96891           0.9637
## Pos Pred Value                   0.9525               0.67617           0.8759
## Neg Pred Value                   0.9954               0.96197           0.9789
## Prevalence                       0.1080               0.09357           0.2167
## Detection Rate                   0.1039               0.05885           0.2005
## Detection Prevalence             0.1091               0.08703           0.2289
## Balanced Accuracy                0.9783               0.79891           0.9444
##                      Class: Red.Soil Class: Soil.With.Vegetation.Stubble
## Sensitivity                   0.9674                             0.83830
## Specificity                   0.9902                             0.98386
## Pos Pred Value                0.9692                             0.86026
## Neg Pred Value                0.9896                             0.98089
```

```
## Prevalence                  0.2417                             0.10598
## Detection Rate              0.2338                             0.08884
## Detection Prevalence        0.2413                             0.10327
## Balanced Accuracy           0.9788                             0.91108
##                        Class: Very.Damp.Grey.Soil
## Sensitivity                                0.8661
## Specificity                                0.9638
## Pos Pred Value                             0.8796
## Neg Pred Value                             0.9593
## Prevalence                                 0.2340
## Detection Rate                             0.2027
## Detection Prevalence                       0.2304
## Balanced Accuracy                          0.9149
```

```
classAcc(confusion_matrix)
```

```
##                  Cotton.Crop               Damp.Grey.Soil
##                         96.2                         62.9
##                    Grey.Soil                     Red.Soil
##                         92.5                         96.7
## Soil.With.Vegetation.Stubble          Very.Damp.Grey.Soil
##                         83.8                         86.6
```

The overall accuracy of model is 89.7% which is much better than a random guess while NIR is 0.2417. The accuracies for each class shows what we predicted - the accuracy for "Damp Grey Soil" is only 60.2% while for other classes: + "Cotton Crop" - 95.6%, + "Grey Soil" - 94.1%, + "Red Soil" - 97.5%, + "Soil With Vegetation Stubble" - 84.3%, + Very Damp Grey Soil" - 89.1%.

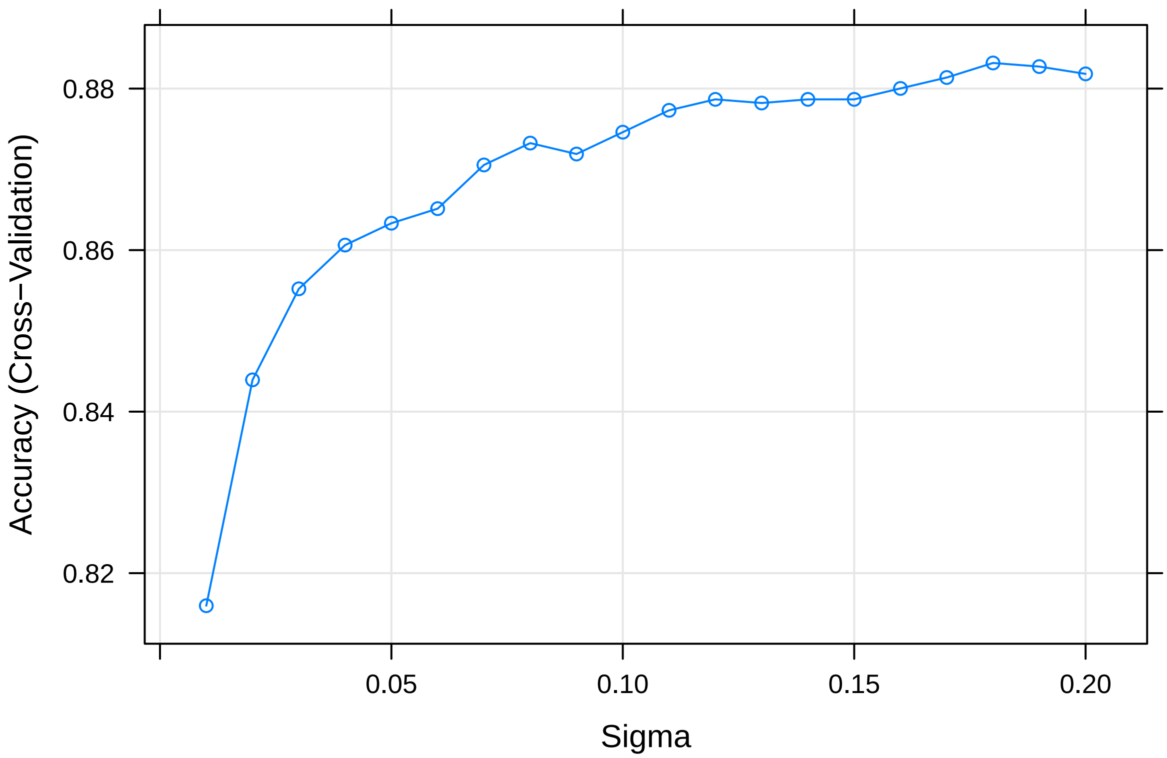
1.5.2 Gaussian Process with Radial Basis Function Kernel

```
fitControl <- trainControl(
  method = "cv",
  number = 2,
  classProbs = TRUE,
  savePredictions="all",
  verboseIter = TRUE)

gpGrid =  expand.grid(sigma = seq(0.01,0.2,0.01))


gp.fit <- train(Target ~ ., data = data %>%
                   mutate(Target = factor(Target, labels = make.names(levels(Target)))),
                 method = "gaussprRadial",
                 trControl = fitControl,
                 preProcess=c("center", "scale","pca"),
                 tuneGrid = gpGrid)
```

```
## Aggregating results
## Selecting tuning parameters
## Fitting sigma = 0.18 on full training set
```

Let's look which sigma gives the best accuracy:

```
plot(gp.fit,metric = "Accuracy")
```



Best results are obtained when sigma is 0.15.

Using sigma = 0.15 let's look at the accuracies of each class and overall accuracy:

```
resamp_rb = gp.fit$pred[gp.fit$pred$sigma == gp.fit$bestTune[1,1],]
confusion_matrix <- confusionMatrix(resamp_rb$pred, resamp_rb$obs)

confusion_matrix
```

```
## Confusion Matrix and Statistics
##
##                                 Reference
## Prediction                        Cotton.Crop Damp.Grey.Soil Grey.Soil Red.Soil
##   Cotton.Crop                             230              2         2        2
##   Damp.Grey.Soil                            2            102        14        1
##   Grey.Soil                                 0             48       451        6
##   Red.Soil                                  0              2         5      525
##   Soil.With.Vegetation.Stubble              4              5         0        2
##   Very.Damp.Grey.Soil                       3             49         8        0
##                                 Reference
## Prediction                        Soil.With.Vegetation.Stubble Very.Damp.Grey.Soil
##   Cotton.Crop                                                2                   1
```

```
##   Damp.Grey.Soil                                        3                   25
##   Grey.Soil                                             1                   11
##   Red.Soil                                             15                    0
##   Soil.With.Vegetation.Stubble                        173                    5
##   Very.Damp.Grey.Soil                                  41                  477
##
## Overall Statistics
##
##                Accuracy : 0.8832
##                  95% CI : (0.8691, 0.8963)
##     No Information Rate : 0.2418
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8543
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Cotton.Crop Class: Damp.Grey.Soil Class: Grey.Soil
## Sensitivity                      0.9623               0.49038           0.9396
## Specificity                      0.9954               0.97760           0.9620
## Pos Pred Value                   0.9623               0.69388           0.8723
## Neg Pred Value                   0.9954               0.94879           0.9829
## Prevalence                       0.1078               0.09382           0.2165
## Detection Rate                   0.1037               0.04601           0.2034
## Detection Prevalence             0.1078               0.06631           0.2332
## Balanced Accuracy                0.9789               0.73399           0.9508
##                      Class: Red.Soil Class: Soil.With.Vegetation.Stubble
## Sensitivity                   0.9795                             0.73617
## Specificity                   0.9869                             0.99193
## Pos Pred Value                0.9598                             0.91534
## Neg Pred Value                0.9934                             0.96943
## Prevalence                    0.2418                             0.10600
## Detection Rate                0.2368                             0.07803
## Detection Prevalence          0.2467                             0.08525
## Balanced Accuracy             0.9832                             0.86405
##                      Class: Very.Damp.Grey.Soil
## Sensitivity                              0.9191
## Specificity                              0.9405
## Pos Pred Value                           0.8253
## Neg Pred Value                           0.9744
## Prevalence                               0.2341
## Detection Rate                           0.2152
## Detection Prevalence                     0.2607
## Balanced Accuracy                        0.9298
```

```
classAcc(confusion_matrix)
```

```
##                  Cotton.Crop               Damp.Grey.Soil
##                         96.2                         49.0
##                    Grey.Soil                     Red.Soil
##                         94.0                         97.9
## Soil.With.Vegetation.Stubble          Very.Damp.Grey.Soil
```

```
##                       73.6                          91.9
```

The overall accuracy of model is 87.9% which is much better than a random guess while NIR is 0.2419. The accuracies for each class shows what we predicted - the accuracy for "Damp Grey Soil" is only 54.1% while for other classes: + "Cotton Crop" - 97.9%, + "Grey Soil" - 92.9%, + "Red Soil" - 97.0%, + "Soil With Vegetation Stubble" - 72.3%, + Very Damp Grey Soil" - 89.6%.

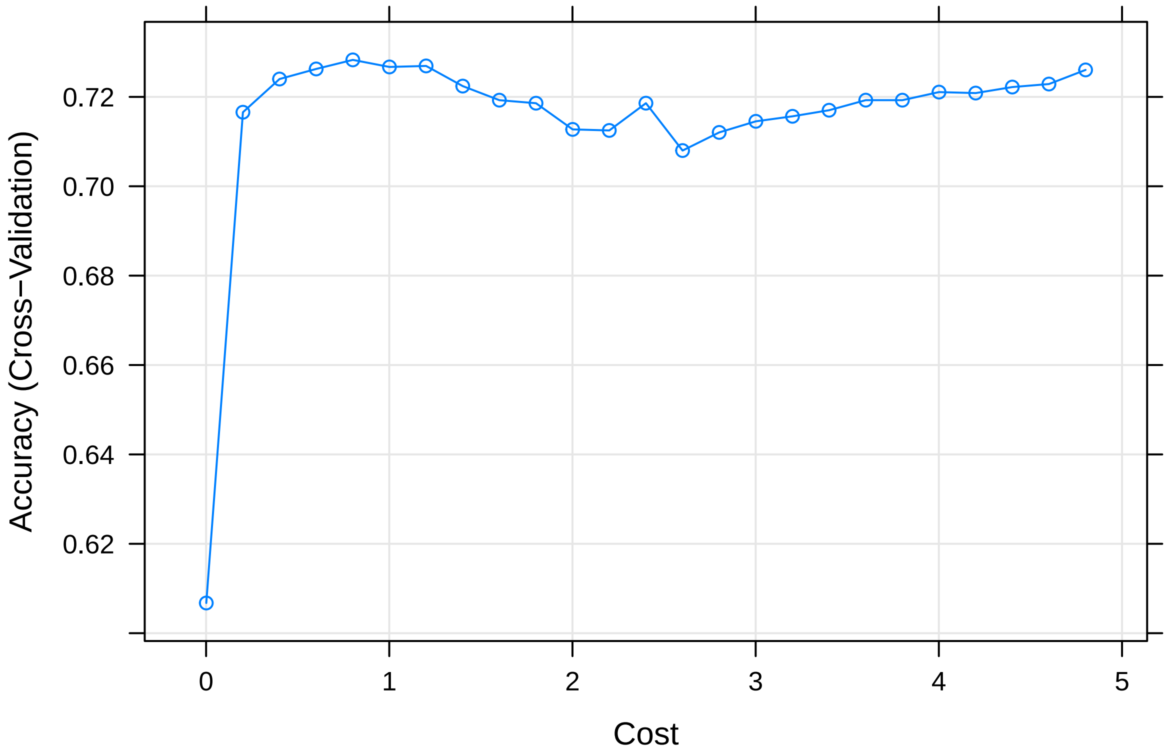
1.5.3 Support Vector Machines with Linear Kernel

```
fitControl <- trainControl(
  method = "cv",
  number = 10,
  classProbs = TRUE,
  savePredictions="all",
  verboseIter = TRUE)

svmGrid =  expand.grid(C=seq(0.001,5,0.2))

svm.fit <- train(Target ~ ., data = data %>%
                   mutate(Target = factor(Target, labels = make.names(levels(Target)))),
                 method = "svmLinear",
                 trControl = fitControl,
                 preProcess=c("center", "scale","pca"),
                 tuneGrid = svmGrid)
```

```
## Aggregating results
## Selecting tuning parameters
## Fitting C = 0.801 on full training set
```

Let's look which Cost gives us the best accuracy:

```
plot(svm.fit,metric = "Accuracy")
```



Best accuracy is obtained using Cost = 0.601.

Using Cost = 0.601 let's look at the accuracies:

```
resamp_svm = svm.fit$pred[svm.fit$pred$C==svm.fit$bestTune[1,1],]

confusion_matrix <- confusionMatrix(resamp_svm$pred, resamp_svm$obs)

confusion_matrix
```

```
## Confusion Matrix and Statistics
##
##                                 Reference
## Prediction                        Cotton.Crop Damp.Grey.Soil Grey.Soil Red.Soil
##   Cotton.Crop                             462              5         1        0
##   Damp.Grey.Soil                            0            174        27        2
##   Grey.Soil                                 0            116       903       11
##   Red.Soil                                  0              5        27      741
##   Soil.With.Vegetation.Stubble              7              2         0      318
##   Very.Damp.Grey.Soil                      10            113         3        0
##                                 Reference
## Prediction                        Soil.With.Vegetation.Stubble Very.Damp.Grey.Soil
##   Cotton.Crop                                               43                   1
##   Damp.Grey.Soil                                             9                 121
##   Grey.Soil                                                  2                  25
##   Red.Soil                                                  18                   0
```

```
##   Soil.With.Vegetation.Stubble                         60                   1
##   Very.Damp.Grey.Soil                                 338                 890
##
## Overall Statistics
##
##                Accuracy : 0.7283
##                  95% CI : (0.7149, 0.7413)
##     No Information Rate : 0.2417
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6625
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Cotton.Crop Class: Damp.Grey.Soil Class: Grey.Soil
## Sensitivity                      0.9645               0.41928           0.9396
## Specificity                      0.9874               0.96045           0.9557
## Pos Pred Value                   0.9023               0.52252           0.8543
## Neg Pred Value                   0.9957               0.94125           0.9828
## Prevalence                       0.1080               0.09357           0.2167
## Detection Rate                   0.1042               0.03923           0.2036
## Detection Prevalence             0.1154               0.07508           0.2383
## Balanced Accuracy                0.9759               0.68986           0.9477
##                      Class: Red.Soil Class: Soil.With.Vegetation.Stubble
## Sensitivity                   0.6912                             0.12766
## Specificity                   0.9851                             0.91728
## Pos Pred Value                0.9368                             0.15464
## Neg Pred Value                0.9092                             0.89869
## Prevalence                    0.2417                             0.10598
## Detection Rate                0.1671                             0.01353
## Detection Prevalence          0.1784                             0.08749
## Balanced Accuracy             0.8382                             0.52247
##                      Class: Very.Damp.Grey.Soil
## Sensitivity                              0.8574
## Specificity                              0.8634
## Pos Pred Value                           0.6573
## Neg Pred Value                           0.9520
## Prevalence                               0.2340
## Detection Rate                           0.2007
## Detection Prevalence                     0.3053
## Balanced Accuracy                        0.8604
```

```
classAcc(confusion_matrix)
```

```
##                  Cotton.Crop                Damp.Grey.Soil
##                         96.5                          41.9
##                   Grey.Soil                      Red.Soil
##                         94.0                          69.1
## Soil.With.Vegetation.Stubble           Very.Damp.Grey.Soil
##                         12.8                          85.7
```

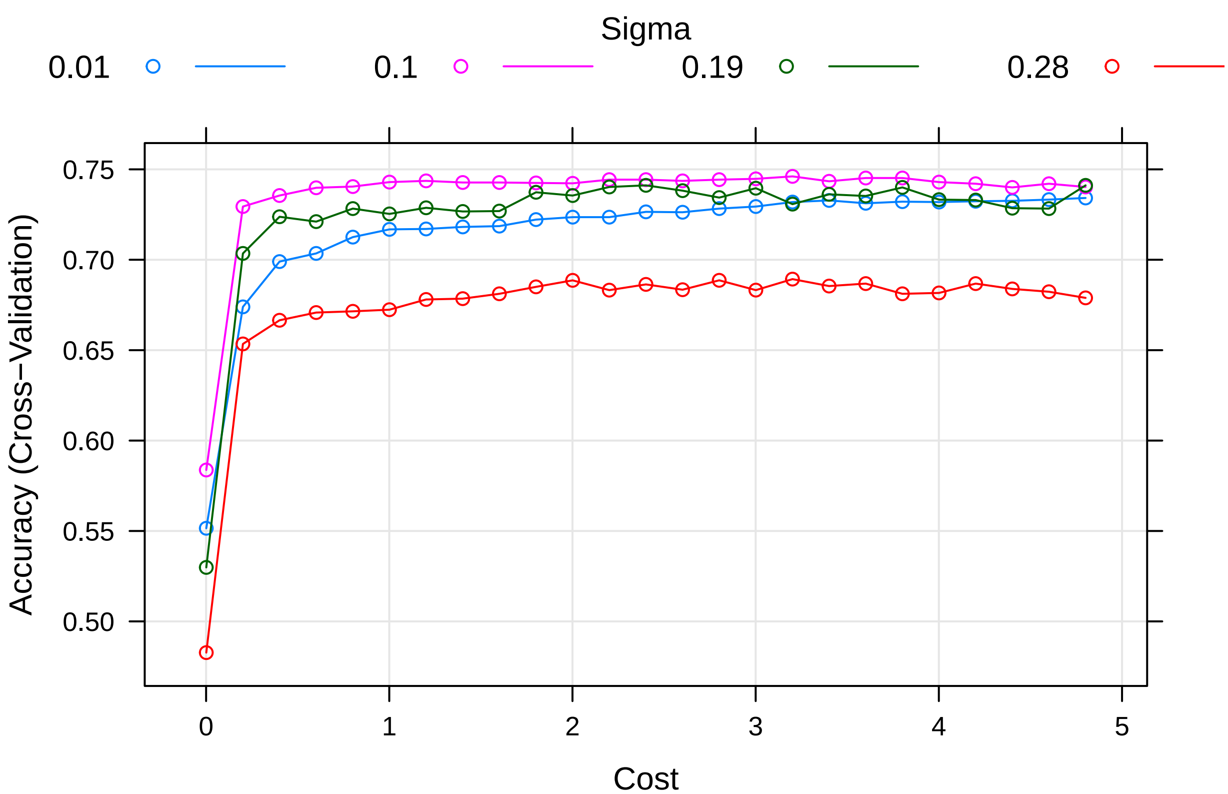
1.5.4 Support Vector Machines with Radial Basis Function Kernel

```
fitControl <- trainControl(
  method = "cv",
  number = 10,
  classProbs = TRUE,
  savePredictions="all",
  verboseIter = TRUE)

svmGrid =  expand.grid(C=seq(0.001,5,0.2), sigma = seq(0.01, 0.3, 0.09))

svm.radial.fit <- train(Target ~ ., data = data %>%
                   mutate(Target = factor(Target, labels = make.names(levels(Target)))),
                 method = "svmRadial",
                 trControl = fitControl,
                 preProcess=c("center", "scale","pca"),
                 tuneGrid = svmGrid)
```

```
## Aggregating results
## Selecting tuning parameters
## Fitting sigma = 0.1, C = 3.2 on full training set
```

Let's look at the accuracy using different parameters:

```
plot(svm.radial.fit,metric = "Accuracy")
```



Best accuracy is obtained using sigma = 0.1, Cost = 2.8, let's use those parameters to look at the class accuracies:

```r
resamp_svm_radial = svm.radial.fit$pred[svm.radial.fit$pred$C == svm.radial.fit$bestTune[1,2] & svm.radi

confusion_matrix <- confusionMatrix(resamp_svm_radial$pred, resamp_svm_radial$obs)

confusion_matrix
```

```
## Confusion Matrix and Statistics
##
##                                Reference
## Prediction                      Cotton.Crop Damp.Grey.Soil Grey.Soil Red.Soil
##   Cotton.Crop                           455              7         3        3
##   Damp.Grey.Soil                          6            199         5        1
##   Grey.Soil                               0            110       932        9
##   Red.Soil                                0              3        17      655
##   Soil.With.Vegetation.Stubble           10              0         1      404
##   Very.Damp.Grey.Soil                     8             96         3        0
##                                Reference
## Prediction                      Soil.With.Vegetation.Stubble Very.Damp.Grey.Soil
##   Cotton.Crop                                             15                   1
##   Damp.Grey.Soil                                           3                  68
##   Grey.Soil                                                8                  50
##   Red.Soil                                                19                   0
##   Soil.With.Vegetation.Stubble                           150                   1
##   Very.Damp.Grey.Soil                                    275                 918
##
## Overall Statistics
##
##                Accuracy : 0.7461
##                  95% CI : (0.733, 0.7589)
##     No Information Rate : 0.2417
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6858
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Cotton.Crop Class: Damp.Grey.Soil Class: Grey.Soil
## Sensitivity                      0.9499               0.47952           0.9698
## Specificity                      0.9927               0.97935           0.9491
## Pos Pred Value                   0.9401               0.70567           0.8404
## Neg Pred Value                   0.9939               0.94799           0.9913
## Prevalence                       0.1080               0.09357           0.2167
## Detection Rate                   0.1026               0.04487           0.2101
## Detection Prevalence             0.1091               0.06359           0.2501
## Balanced Accuracy                0.9713               0.72944           0.9594
##                      Class: Red.Soil Class: Soil.With.Vegetation.Stubble
## Sensitivity                   0.6110                             0.31915
## Specificity                   0.9884                             0.89508
## Pos Pred Value                0.9438                             0.26502
```

```
## Neg Pred Value              0.8885                         0.91729
## Prevalence                  0.2417                         0.10598
## Detection Rate              0.1477                         0.03382
## Detection Prevalence        0.1565                         0.12762
## Balanced Accuracy           0.7997                         0.60712
##                      Class: Very.Damp.Grey.Soil
## Sensitivity                              0.8844
## Specificity                              0.8875
## Pos Pred Value                           0.7062
## Neg Pred Value                           0.9617
## Prevalence                               0.2340
## Detection Rate                           0.2070
## Detection Prevalence                     0.2931
## Balanced Accuracy                        0.8860
```

```
classAcc(confusion_matrix)
```

```
##                  Cotton.Crop              Damp.Grey.Soil
##                         95.0                        48.0
##                    Grey.Soil                    Red.Soil
##                         97.0                        61.1
## Soil.With.Vegetation.Stubble          Very.Damp.Grey.Soil
##                         31.9                        88.4
```

Overall accuracy is 74.4% while the accuracies for each class are: + Cotton Crop - 95.8%, + Damp Grey Soil - 46.3%, + Grey Soil - 96.5%, + Red Soil - 60.5%, + Soil With Vegetation Stubble - 32.3%, + Very Damp Grey Soil - 88.6%

1.6 Comparison of models

```
accuracies <- data.frame() %>%
   bind_rows(
      classAcc(confusionMatrix(resamp_vbmp$pred, resamp_vbmp$obs)),
      classAcc(confusionMatrix(resamp_rb$pred, resamp_rb$obs)),
      classAcc(confusionMatrix(resamp_svm$pred, resamp_svm$obs)),
      classAcc(confusionMatrix(resamp_svm_radial$pred, resamp_svm_radial$obs))
             ) %>%
   bind_cols(
      model = c("GP - Variational Bayesian Multinomial Probit Reg.", "GP - Radial Basis", "SVM - Linear
      overall_accuracy = round(c(
         sum(diag(as.matrix(confusionMatrix(resamp_vbmp$pred, resamp_vbmp$obs)$table))) / sum(colSums(c
         sum(diag(as.matrix(confusionMatrix(resamp_rb$pred, resamp_rb$obs)$table))) / sum(colSums(confu
         sum(diag(as.matrix(confusionMatrix(resamp_svm$pred, resamp_svm$obs)$table))) / sum(colSums(con
         sum(diag(as.matrix(confusionMatrix(resamp_svm_radial$pred, resamp_svm_radial$obs)$table))) / s
      ), 2)
      ) %>%
   tibble::column_to_rownames(var = "model")

accuracies
```

```
##                                                     Cotton.Crop Damp.Grey.Soil
```

```
## GP - Variational Bayesian Multinomial Probit Reg.        96.2           62.9
## GP - Radial Basis                                        96.2           49.0
## SVM - Linear Kernel                                      96.5           41.9
## SVM - Radial Kernel                                      95.0           48.0
##                                                   Grey.Soil Red.Soil
## GP - Variational Bayesian Multinomial Probit Reg.      92.5     96.7
## GP - Radial Basis                                      94.0     97.9
## SVM - Linear Kernel                                    94.0     69.1
## SVM - Radial Kernel                                    97.0     61.1
##                                                   Soil.With.Vegetation.Stubble
## GP - Variational Bayesian Multinomial Probit Reg.                         83.8
## GP - Radial Basis                                                         73.6
## SVM - Linear Kernel                                                       12.8
## SVM - Radial Kernel                                                       31.9
##                                                   Very.Damp.Grey.Soil
## GP - Variational Bayesian Multinomial Probit Reg.                86.6
## GP - Radial Basis                                                91.9
## SVM - Linear Kernel                                              85.7
## SVM - Radial Kernel                                              88.4
##                                                   overall_accuracy
## GP - Variational Bayesian Multinomial Probit Reg.            88.86
## GP - Radial Basis                                            88.32
## SVM - Linear Kernel                                          72.83
## SVM - Radial Kernel                                          74.61
```

Variational Bayesian Multinomial Probit Regression gave us the best overall accuracy of 89.7% while Gaussian Process with Radial Basis has 87.7% accuracy. Simple SVM with linear kernel gave us the worst overall accuracy of 72.5%.

Both Gaussian Process methods struggle with predicting "Damp Grey Soil" class, but that's what we predicted from scatter plots. While SVM methods showed similar results predicting "Damp Grey Soil" class, but the SVM models had a very hard time predicting "Soil With Vegetation Stubble" while GP methods didnt had a very hard time predicting this class.

2 SVM dataset

2.1 Required packages

```
library(SmartEDA)
library(dplyr)
library(ggplot2)
library(caret)
library(R.matlab)
library(kernlab)

# Function for 6 class accuracy from confusion matrix
classAcc <- function(confusionMatrix) {
  class1 <- round(confusionMatrix$table[1, 1] / sum(confusionMatrix$table[, 1]) * 100, 1)
  class2 <- round(confusionMatrix$table[2, 2] / sum(confusionMatrix$table[, 2]) * 100, 1)
  acc <- c(class1, class2 )
  names(acc) <- colnames(confusionMatrix$table)
```

```
  return(acc)
}
```

2.2 Reading Data

```
set.seed(123)

annthyroid <- readMat("annthyroid.mat") %>% as.data.frame()
```

2.3 EDA, first look at the dataset

```
ExpData(annthyroid,type=1)
```

```
##                                              Descriptions    Value
## 1                                      Sample size (nrow)     7200
## 2                                 No. of variables (ncol)        7
## 3                       No. of numeric/interger variables        7
## 4                                 No. of factor variables        0
## 5                                   No. of text variables        0
## 6                                No. of logical variables        0
## 7                             No. of identifier variables        0
## 8                                   No. of date variables        0
## 9                No. of zero variance variables (uniform)        0
## 10                  %. of variables having complete cases 100% (7)
## 11   %. of variables having >0% and <50% missing cases   0% (0)
## 12 %. of variables having >=50% and <90% missing cases   0% (0)
## 13          %. of variables having >=90% missing cases   0% (0)
```

We have a dataset of 7200 observations, all 7 variables are of numeric. None of the columns have missing values.

Let's look at the target variable frequencies

```
annthyroid %>%
   group_by(y) %>%
   summarise(n = n()) %>%
   mutate(n_prop = round(n / sum(n) * 100, 2))
```

```
## # A tibble: 2 x 3
##       y     n n_prop
##   <dbl> <int>  <dbl>
## 1     0  6666   92.6
## 2     1   534   7.42
```

It's a two-class problem, there's a huge class imbalance of 6666 (92.5%) / 534 (7.5%). In this lab, we are not going to try to address this problem.

Let's look at descriptive statistics of each variable:

```r
ExpNumStat(annthyroid,by ="A",round= 2, gp = "y") %>%
  select(Vname, min, max, mean, median, SD)
```

```
##   Vname  min  max mean median   SD
## 1   X.1 0.01 0.97 0.52   0.55 0.19
## 2   X.2 0.00 0.53 0.00   0.00 0.02
## 3   X.3 0.00 0.18 0.02   0.02 0.01
## 4   X.4 0.00 0.60 0.11   0.11 0.04
## 5   X.5 0.02 0.23 0.10   0.10 0.02
## 6   X.6 0.00 0.64 0.11   0.11 0.04
```

Nothing seems unordinary.

We should look at the correlation between variables

```r
correlation_matrix = cor(annthyroid %>% select(-y))

length(findCorrelation(correlation_matrix, cutoff = 0.99))
```

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

```r
length(findCorrelation(correlation_matrix, cutoff = 0.95))
```

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

```r
length(findCorrelation(correlation_matrix, cutoff = 0.9))
```

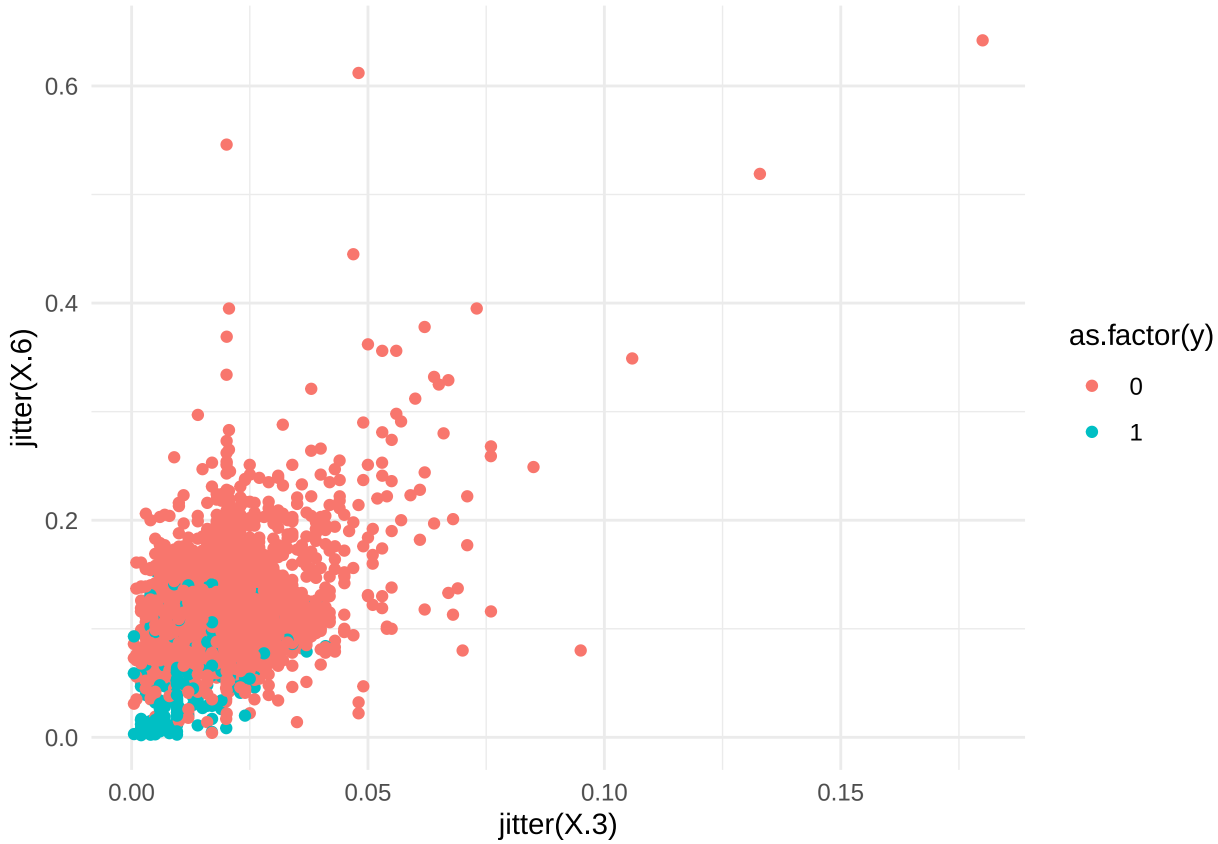
```
## [1] 0
```

We don't have any variables with higher correlation than 0.9.

We'll look through random scatter plots and see how our target variable is seperated across. We'll save some interesting combinations for later. P.S. we'll use jitter() function to overcome observation overlap.

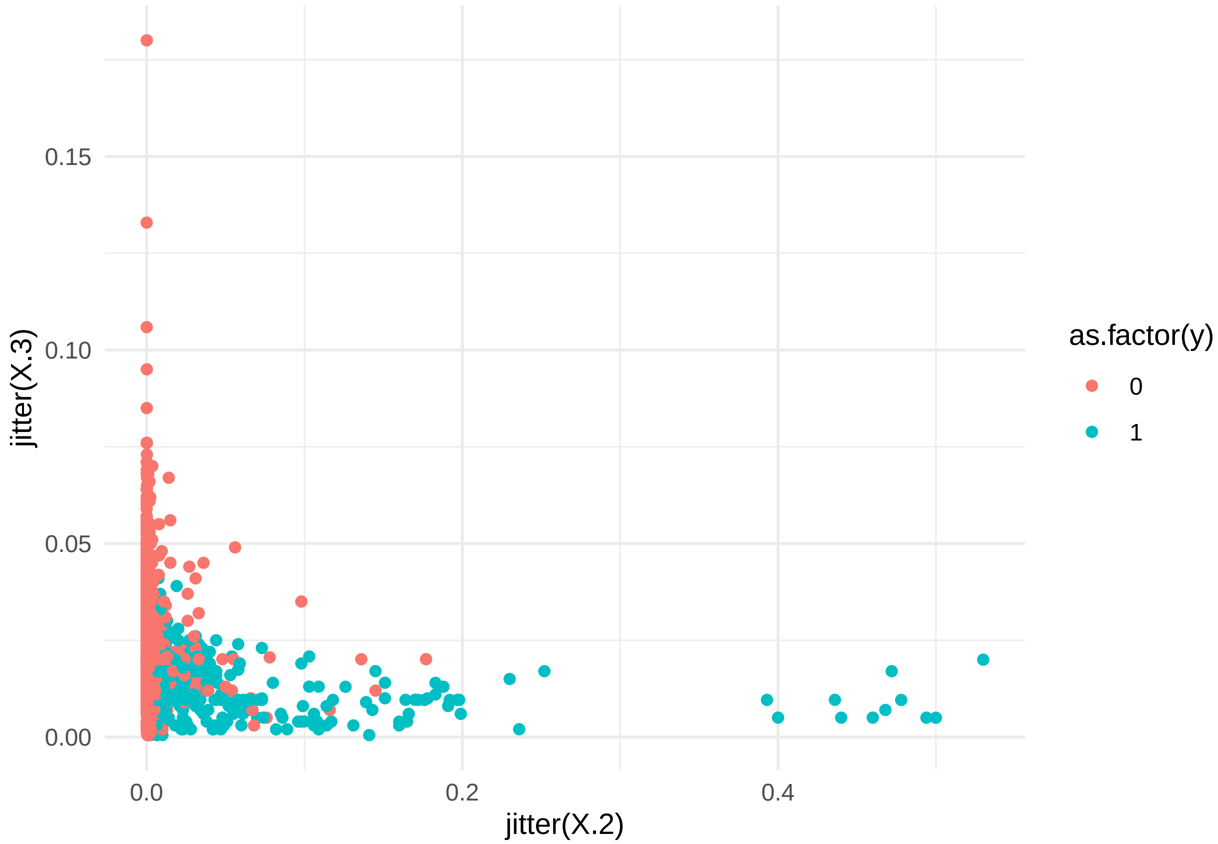
```r
# data_plot <- annthyroid %>%
#    select(sample(0:6, 1), sample(0:6, 1), y) %>%
#   mutate(y = as.factor(y))
# data_plot_colnames <- colnames(data_plot)
# colnames(data_plot) <- c("V_1", "V_2", "y")
# data_plot %>%
#    ggplot(aes(x = jitter(V_1), y = jitter(V_2), color = y)) +
#    geom_point() +
#    xlab(data_plot_colnames[1]) +
#    ylab(data_plot_colnames[2])

#combinations <- c("X.3 - X.6", "X.2 - X.3", "X.5 - X.3")
```

```r
annthyroid %>%
   ggplot(aes(x = jitter(X.3), y = jitter(X.6), color = as.factor(y))) +
   geom_point() +
   theme_minimal()
```



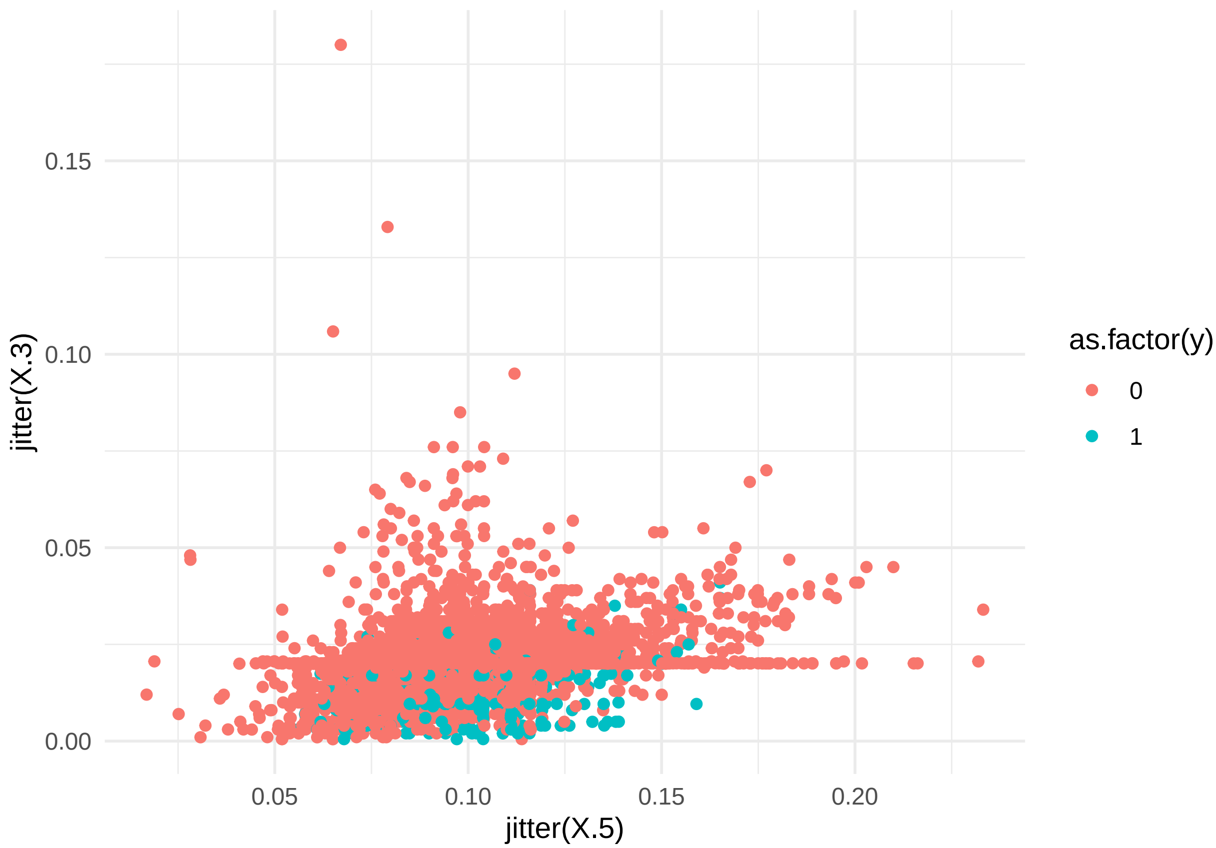
Combination of X.3 and X.6 features shows a very small separation between the 2 features. Seems like class 1 lies mostly in the range of X.6 [0 - 0.08] and X.3 [0 - 0.025].

```
annthyroid %>%
  ggplot(aes(x = jitter(X.2), y = jitter(X.3), color = as.factor(y))) +
  geom_point() +
  theme_minimal()
```



This time, the class separation is much better, when the value of X.2 is greater than 0, there's a high chance the target class will be "1".

```
annthyroid %>%
  ggplot(aes(x = jitter(X.5), y = jitter(X.3), color = as.factor(y))) +
  geom_point() +
  theme_minimal()
```



A combination of X.3 and X.5 almost shows us no class separation at all.

2.4 Fitting models

2.4.1 SVM Novelty Detection - Linear Kernel

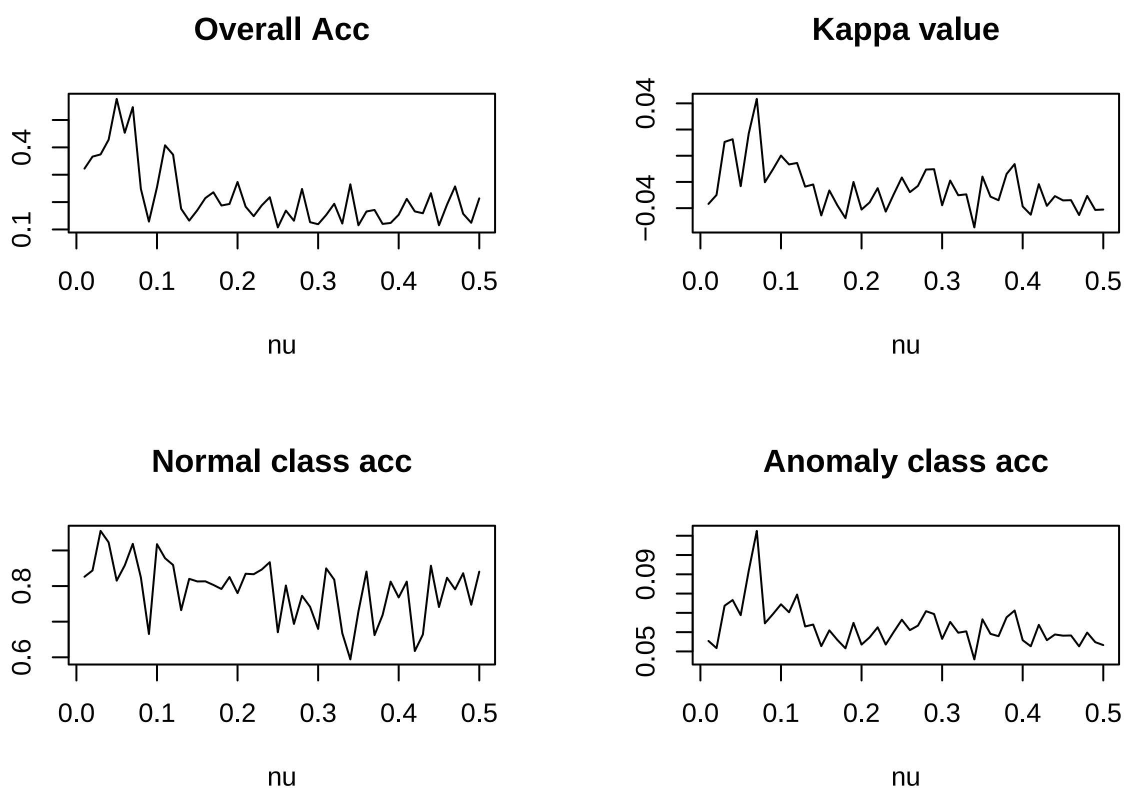
```
fold_ids = createFolds(annthyroid$y, k = 5, list = TRUE, returnTrain = FALSE)

nu_list = seq(0.01,0.5,0.01)

collect_CV = array(0, dim = c(length(nu_list),5,5))

for(k in 1:5){
  for( i in 1:length(nu_list)){
    anomaly = ksvm(y ~ .,
                   annthyroid[-fold_ids[[k]], ],
                   kernel = "vanilladot",
                   type = 'one-svc',
                   nu = nu_list[i], kpar = list())
    y_true = annthyroid$y[fold_ids[[k]]]
    y_pred = 1 - 1*(predict(anomaly, annthyroid[fold_ids[[k]], ]))
    confMat = confusionMatrix(table(y_true = y_true, y_pred = y_pred))
    collect_CV[i,1,k] = confMat$overall[1] #overall accuracy
    collect_CV[i,2,k] = confMat$overall[2] #Kappa
```

```
    collect_CV[i,3,k] = confMat$overall[5] #NIR rate
    collect_CV[i,4,k] = confMat$byClass[1] #Normal class (0) acc
    collect_CV[i,5,k] = confMat$byClass[2] #Anomaly class (1) acc

  }
  print(paste("Done with fold:", k))
}
```

```
## [1] "Done with fold: 1"
## [1] "Done with fold: 2"
## [1] "Done with fold: 3"
## [1] "Done with fold: 4"
## [1] "Done with fold: 5"
```

```
CV = apply(collect_CV, c(1,2), mean)
```

```
par(mfrow=c(2,2))
plot(nu_list,CV[,1],main="Overall Acc",xlab="nu",ylab="", type = "l")
plot(nu_list,CV[,2],main="Kappa value",xlab="nu",ylab="", type = "l")
plot(nu_list,CV[,4],main="Normal class acc",xlab="nu",ylab="", type = "l")
plot(nu_list,CV[,5],main="Anomaly class acc",xlab="nu",ylab="", type = "l")
```



Accuracy for the anomaly class is really small, no matter what nu value we are choosing.

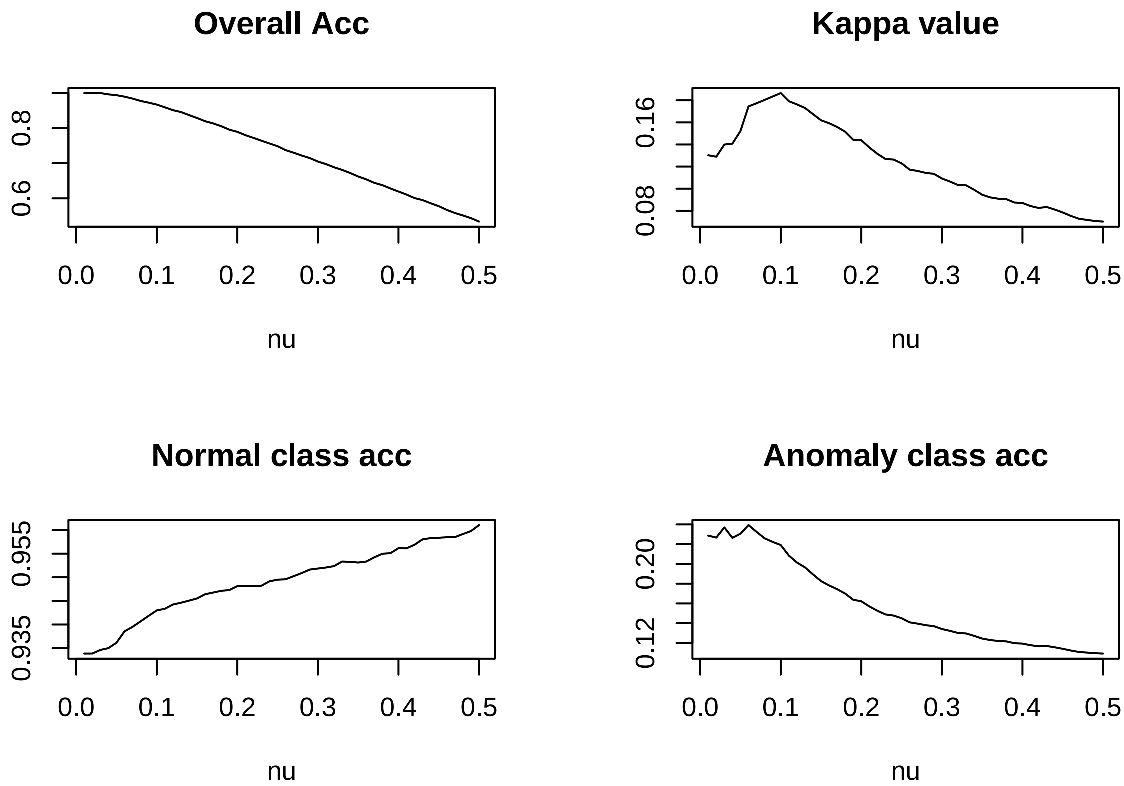
2.4.2 SVM Novelty Detection - Radial Basis Kernel

```
fold_ids = createFolds(annthyroid$y, k = 5, list = TRUE, returnTrain = FALSE)

nu_list = seq(0.01,0.5,0.01)

collect_CV = array(0, dim = c(length(nu_list),5,5))

for(k in 1:5){
  for( i in 1:length(nu_list)){
    anomaly = ksvm(y ~ .,
                   annthyroid[-fold_ids[[k]], ],
                   kernel = "rbfdot",
                   type = 'one-svc',
                   nu = nu_list[i], kpar = "automatic")
    y_true = annthyroid$y[fold_ids[[k]]]
    y_pred = 1 - 1*(predict(anomaly, annthyroid[fold_ids[[k]], ]))
    confMat = confusionMatrix(table(y_true = y_true, y_pred = y_pred))
    collect_CV[i,1,k] = confMat$overall[1] #overall accuracy
    collect_CV[i,2,k] = confMat$overall[2] #Kappa
    collect_CV[i,3,k] = confMat$overall[5] #NIR rate
    collect_CV[i,4,k] = confMat$byClass[1] #Normal class (0) acc
    collect_CV[i,5,k] = confMat$byClass[2] #Anomaly class (1) acc
  }
  print(paste("Done with fold:", k))
}
```

```
## [1] "Done with fold: 1"
## [1] "Done with fold: 2"
## [1] "Done with fold: 3"
## [1] "Done with fold: 4"
## [1] "Done with fold: 5"
```

```
CV = apply(collect_CV, c(1,2), mean)
```

```
par(mfrow=c(2,2))
plot(nu_list,CV[,1],main="Overall Acc",xlab="nu",ylab="", type = "l")
plot(nu_list,CV[,2],main="Kappa value",xlab="nu",ylab="", type = "l")
plot(nu_list,CV[,4],main="Normal class acc",xlab="nu",ylab="", type = "l")
plot(nu_list,CV[,5],main="Anomaly class acc",xlab="nu",ylab="", type = "l")
```



The higher the value of nu, the lower accuracy for the anomaly class. Highest accuracy of the anomaly class is reached using nu ~0.05 and the accuracy is ~24%

2.5 Conclusion

Seems like using Radial Basis Kernel gives much better prediction of anomaly class (~24%) while the linear kernel reached only 8% accuracy for the anomaly class