classification2 -

Chen Si (chensi3), Yuchen Cao (yuchenc5)

2023-04-30

```
library(caret)
```

```
##      ggplot2
```

```
##      lattice
```

```
library(kernlab)
```

```
##
##    'kernlab'
```

```
## The following object is masked from 'package:ggplot2':
##
##     alpha
```

```
library(ggplot2)
library(lattice)
library(tibble)
library(glmnet)
```

```
## Warning:   'glmnet' R 4.2.3
```

```
##      Matrix
```

```
## Loaded glmnet 4.1-7
```

```
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.
```

```
##
##    'pROC'
```

```
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

Read Data

```r
fifa = read.csv("../dataset/players_20_edited.csv")
```

```r
best.pos.c = c()
for (i in 1:dim(fifa)[1]){
  best.pos.c = c(best.pos.c, which.max(fifa[i,c(67,71,72,76)]))
}
fifa$wb_or_b = best.pos.c
```

```r
# colnames(fifa)
fifa.filtered = fifa[,setdiff(c(3:50,78), c(6:12,14,15))]
colnames(fifa.filtered)
```

```
##  [1] "age"                        "height_cm"
##  [3] "weight_kg"                  "preferred_foot"
##  [5] "pace"                       "shooting"
##  [7] "passing"                    "dribbling"
##  [9] "defending"                  "physic"
## [11] "attacking_crossing"         "attacking_finishing"
## [13] "attacking_heading_accuracy" "attacking_short_passing"
## [15] "attacking_volleys"          "skill_dribbling"
## [17] "skill_curve"                "skill_fk_accuracy"
## [19] "skill_long_passing"         "skill_ball_control"
## [21] "movement_acceleration"      "movement_sprint_speed"
## [23] "movement_agility"           "movement_reactions"
## [25] "movement_balance"           "power_shot_power"
## [27] "power_jumping"              "power_stamina"
## [29] "power_strength"             "power_long_shots"
## [31] "mentality_aggression"       "mentality_interceptions"
## [33] "mentality_positioning"      "mentality_vision"
## [35] "mentality_penalties"        "mentality_composure"
## [37] "defending_marking"          "defending_standing_tackle"
## [39] "defending_sliding_tackle"   "wb_or_b"
```

Set Preferred Foot to be Factor

```r
fifa.filtered$preferred_foot = as.factor(fifa.filtered$preferred_foot)
summary(fifa.filtered$preferred_foot)
```

```
##  Left Right
##  4098 12144
```

```r
fifa.filtered$wb_or_b = as.factor(fifa.filtered$wb_or_b)
levels(fifa.filtered$wb_or_b) = c("wb", "b")
summary(fifa.filtered$wb_or_b)
```

```
##    wb     b
## 12492  3750
```

Train-Test Split 8-2

```
set.seed(432)

N=dim(fifa)[1]
cat("There are ", N, "players in the dataset\n")
```

```
## There are  16242 players in the dataset
```

```
all.idx=1:N

trn.idx = sample(all.idx, size=round(0.8*N))
tst.idx = all.idx[is.na(pmatch(all.idx,trn.idx))]

fifa.trn = fifa.filtered[trn.idx,]
fifa.tst = fifa.filtered[tst.idx,]
cat("Train Set Size:", dim(fifa.trn)[1], "\n")
```

```
## Train Set Size: 12994
```

```
cat("Test Set Size:", dim(fifa.tst)[1], "\n")
```

```
## Test Set Size: 3248
```

Simple SVM

Train

```
train_control <- trainControl(method="cv", number=10)

simple.svm <- train(wb_or_b ~ .,
              data = fifa.trn,
              method = "svmLinear",
              trControl = train_control,
              preProcess = c("center","scale"))

simple.svm
```

```
## Support Vector Machines with Linear Kernel
##
## 12994 samples
##    39 predictor
##     2 classes: 'wb', 'b'
##
## Pre-processing: centered (39), scaled (39)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 11695, 11694, 11694, 11695, 11695, 11695, ...
## Resampling results:
```

```
##
##   Accuracy   Kappa
##   0.9609821  0.8895397
##
## Tuning parameter 'C' was held constant at a value of 1
```

Test - Test Set

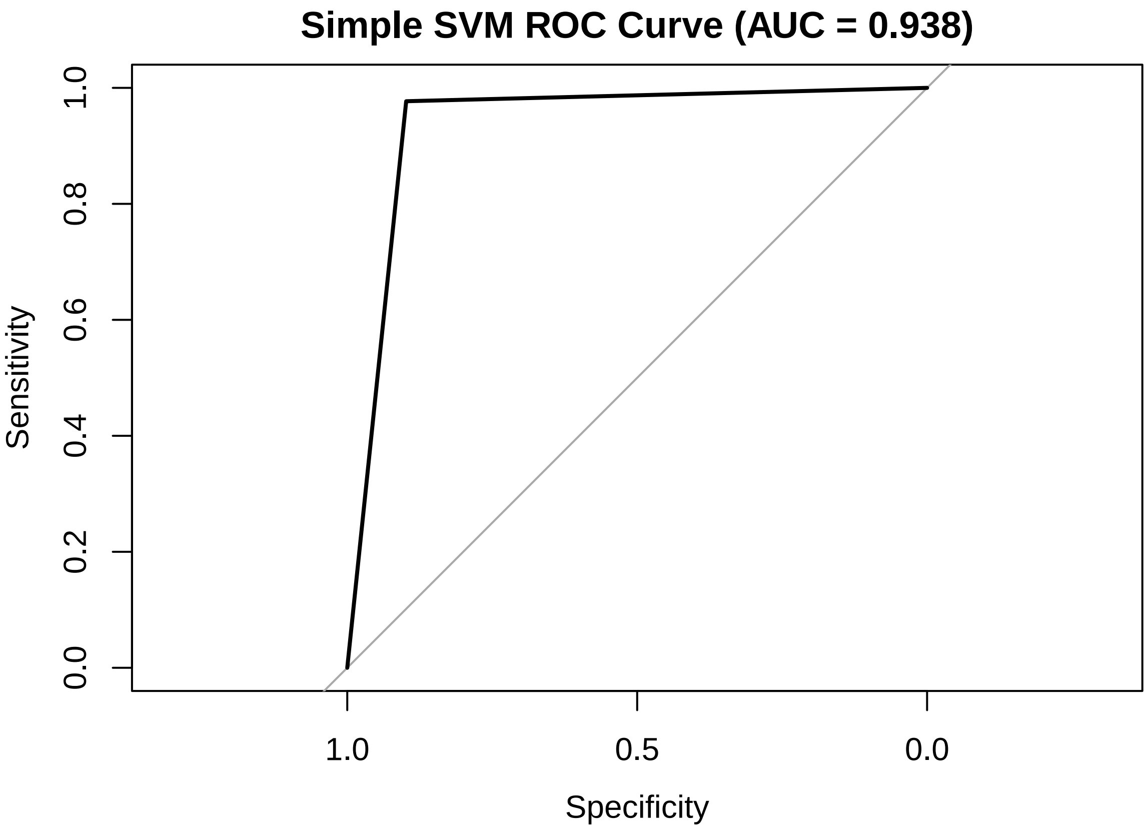
```
predict.res = predict(simple.svm, newdata=fifa.tst)
confusionMatrix(predict.res, fifa.tst$wb_or_b)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   wb    b
##         wb 2462   74
##         b    58  654
##
##                Accuracy : 0.9594
##                  95% CI : (0.952, 0.9659)
##     No Information Rate : 0.7759
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8822
##
##  Mcnemar's Test P-Value : 0.1917
##
##             Sensitivity : 0.9770
##             Specificity : 0.8984
##          Pos Pred Value : 0.9708
##          Neg Pred Value : 0.9185
##              Prevalence : 0.7759
##          Detection Rate : 0.7580
##    Detection Prevalence : 0.7808
##       Balanced Accuracy : 0.9377
##
##        'Positive' Class : wb
##
```

```
real.wb_or_b_num = ifelse(fifa.tst$wb_or_b == "wb", 1, 0)
pred.wb_or_b_num = ifelse(predict.res == "wb", 1, 0)

roc_obj <- roc(real.wb_or_b_num, pred.wb_or_b_num)
```

```
## Setting levels: control = 0, case = 1

## Setting direction: controls < cases
```

```
auc <- auc(roc_obj)

plot(roc_obj, main = paste0("Simple SVM ROC Curve (AUC = ", round(auc, 3), ")"))
```



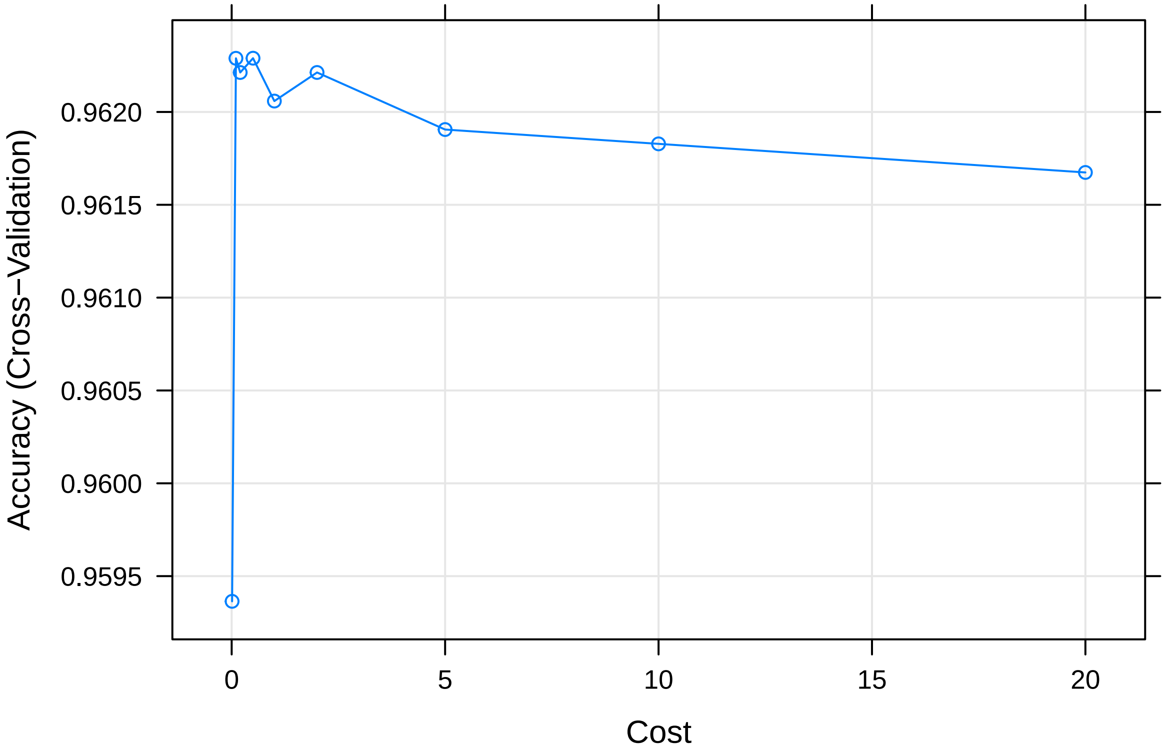
Tuned Linear SVM

Train

```
train_control <- trainControl(method="cv", number=10)

linear.svm <- train(wb_or_b~.,
              data = fifa.trn,
              method = "svmLinear",
              trControl = train_control,
              preProcess = c("center","scale"),
              tuneGrid = expand.grid(C = c(0.01,0.1,0.2,0.5,1,2,5,10,20)))

linear.svm
```

```
## Support Vector Machines with Linear Kernel
##
## 12994 samples
##    39 predictor
##     2 classes: 'wb', 'b'
##
## Pre-processing: centered (39), scaled (39)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 11695, 11695, 11695, 11693, 11695, 11695, ...
## Resampling results across tuning parameters:
##
##   C      Accuracy   Kappa
##    0.01  0.9593644  0.8833803
##    0.10  0.9622891  0.8927183
##    0.20  0.9622126  0.8928026
##    0.50  0.9622896  0.8930972
##    1.00  0.9620587  0.8924748
##    2.00  0.9622127  0.8929852
##    5.00  0.9619052  0.8921429
##   10.00  0.9618281  0.8919106
##   20.00  0.9616742  0.8914686
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was C = 0.5.
```

```
plot(linear.svm)
```



```
linear.best.C = linear.svm$bestTune$C

linear.best.res<-as_tibble(linear.svm$results[which.max(linear.svm$results[,2]),])
linear.best.res
```

```
## # A tibble: 1 x 5
##       C Accuracy Kappa AccuracySD KappaSD
##   <dbl>    <dbl> <dbl>      <dbl>   <dbl>
## 1   0.5    0.962 0.893    0.00650  0.0193
```

```
best.linear.svm = linear.svm$finalModel
best.linear.svm
```

```
## Support Vector Machine object of class "ksvm" 
## 
## SV type: C-svc  (classification) 
##  parameter : cost C = 0.5 
## 
## Linear (vanilla) kernel function. 
## 
## Number of Support Vectors : 1124 
## 
## Objective Function Value : -543.0111 
## Training error : 0.036632
```

Test - Test Set

```
predict.res = predict(linear.svm, newdata=fifa.tst)

confusionMatrix(predict.res, fifa.tst$wb_or_b)
```

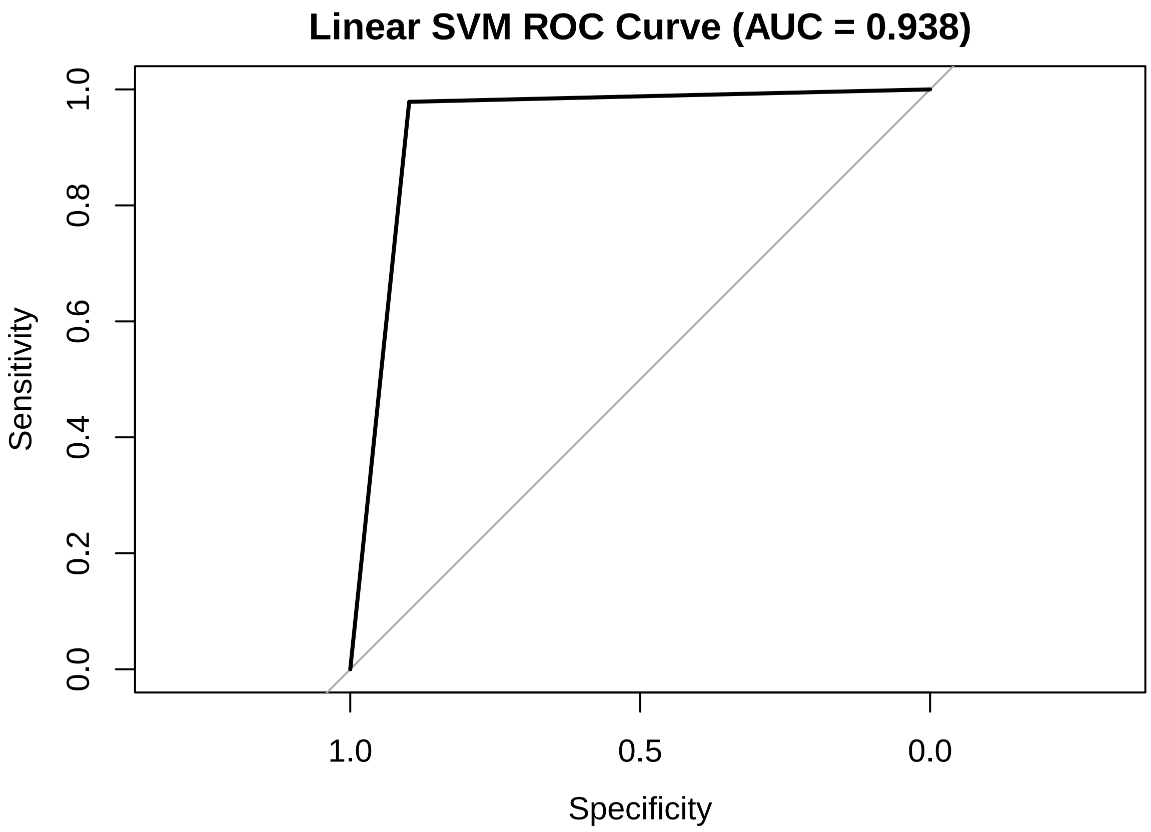
```
## Confusion Matrix and Statistics
## 
##           Reference
## Prediction   wb    b
##         wb 2466   74
##         b    54  654
##                                           
##                Accuracy : 0.9606          
##                  95% CI : (0.9533, 0.967) 
##     No Information Rate : 0.7759          
##     P-Value [Acc > NIR] : < 2e-16         
##                                           
##                   Kappa : 0.8856          
##                                           
##  Mcnemar's Test P-Value : 0.09308         
##                                           
##             Sensitivity : 0.9786          
##             Specificity : 0.8984          
##          Pos Pred Value : 0.9709          
##          Neg Pred Value : 0.9237          
##              Prevalence : 0.7759          
##          Detection Rate : 0.7592          
##    Detection Prevalence : 0.7820          
##       Balanced Accuracy : 0.9385          
##                                           
##        'Positive' Class : wb              
## 
```

```
real.wb_or_b_num = ifelse(fifa.tst$wb_or_b == "wb", 1, 0)
pred.wb_or_b_num = ifelse(predict.res == "wb", 1, 0)

roc_obj <- roc(real.wb_or_b_num, pred.wb_or_b_num)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```
auc <- auc(roc_obj)

plot(roc_obj, main = paste0("Linear SVM ROC Curve (AUC = ", round(auc, 3), ")"))
```

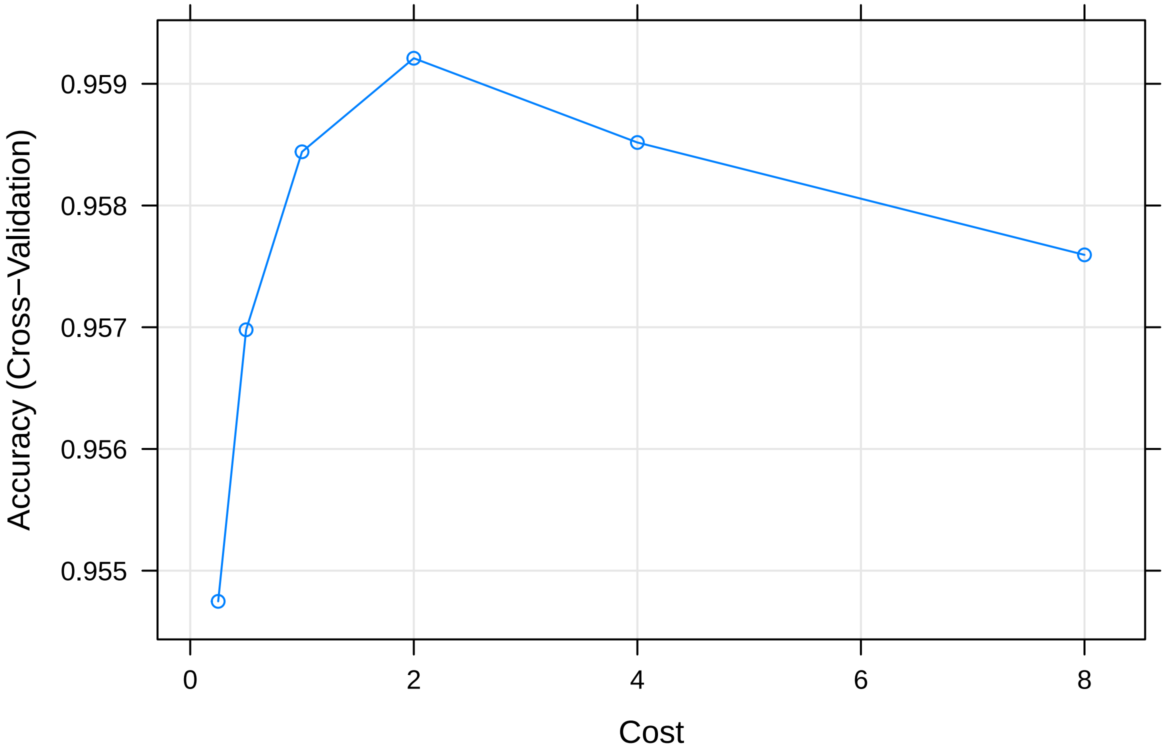


Radial SVM

###Train

```
train_control <- trainControl(method="cv", number=10)

radial.svm <- train(wb_or_b~.,
              data = fifa.trn, method = "svmRadial",
              trControl = train_control,
              preProcess = c("center","scale"),
              tuneLength = 6)
```

```
plot(radial.svm)
```



```
radial.best.C = radial.svm$bestTune$C

radial.best.res<-as_tibble(radial.svm$results[which.max(radial.svm$results[,2]),])
radial.best.res
```

```
## # A tibble: 1 x 6
##    sigma     C Accuracy Kappa AccuracySD KappaSD
##    <dbl> <dbl>    <dbl> <dbl>      <dbl>   <dbl>
## 1 0.0204     8    0.958 0.880    0.00450  0.0129
```

```
best.radial.svm = radial.svm$finalModel
best.radial.svm
```

```
## Support Vector Machine object of class "ksvm"
##
## SV type: C-svc  (classification)
##  parameter : cost C = 2
##
## Gaussian Radial Basis kernel function.
##  Hyperparameter : sigma =  0.0204296846000192
##
## Number of Support Vectors : 1472
##
## Objective Function Value : -2286.409
## Training error : 0.030553
```

Test - Test Set

```
predict.res = predict(radial.svm, newdata=fifa.tst)

confusionMatrix(predict.res, fifa.tst$wb_or_b)
```

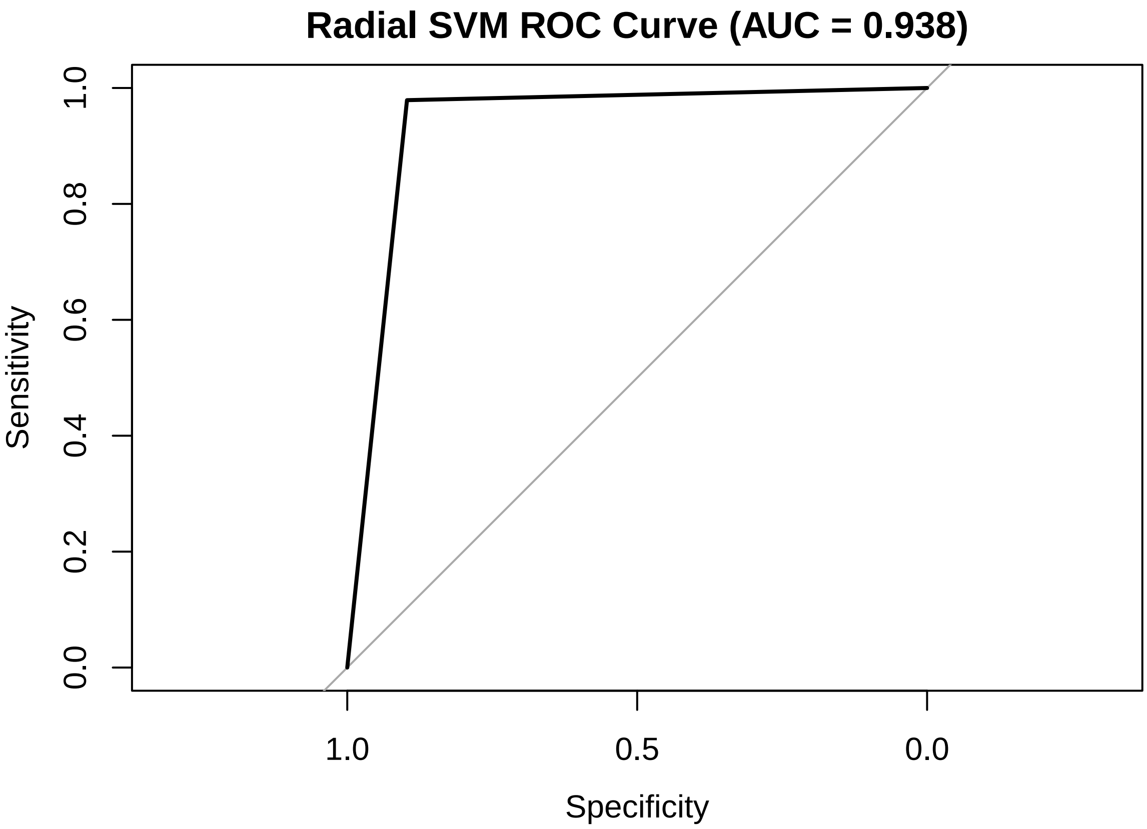
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   wb    b
##         wb 2467   75
##         b    53  653
##
##                Accuracy : 0.9606
##                  95% CI : (0.9533, 0.967)
##     No Information Rate : 0.7759
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.8855
##
##  Mcnemar's Test P-Value : 0.06343
##
##             Sensitivity : 0.9790
##             Specificity : 0.8970
##          Pos Pred Value : 0.9705
##          Neg Pred Value : 0.9249
##              Prevalence : 0.7759
##          Detection Rate : 0.7595
##    Detection Prevalence : 0.7826
##       Balanced Accuracy : 0.9380
##
##        'Positive' Class : wb
##
```

```
real.wb_or_b_num = ifelse(fifa.tst$wb_or_b == "wb", 1, 0)
pred.wb_or_b_num = ifelse(predict.res == "wb", 1, 0)

roc_obj <- roc(real.wb_or_b_num, pred.wb_or_b_num)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```r
auc <- auc(roc_obj)

plot(roc_obj, main = paste0("Radial SVM ROC Curve (AUC = ", round(auc, 3), ")"))
```

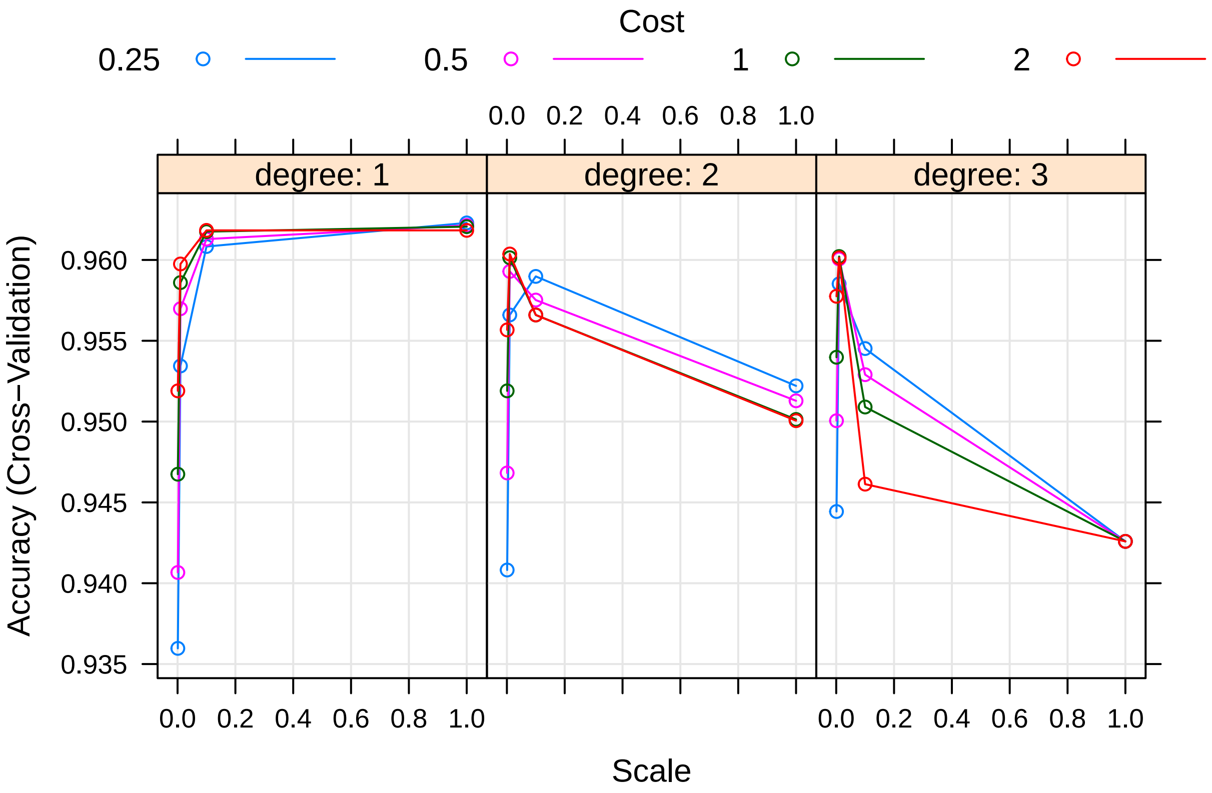


Polynomial SVM

Train

```r
poly.svm <- train(wb_or_b~.,
              data = fifa.trn, method = "svmPoly",
              trControl = train_control,
              preProcess = c("center","scale"),
              tuneLength = 4)
```

```
plot(poly.svm)
```



```
poly.best.C = poly.svm$bestTune$C

poly.best.res<-as_tibble(poly.svm$results[which.max(poly.svm$results[,2]),])
poly.best.res
```

```
## # A tibble: 1 x 7
##   degree scale     C Accuracy Kappa AccuracySD KappaSD
##    <int> <dbl> <dbl>    <dbl> <dbl>      <dbl>   <dbl>
## 1      1     1  0.25    0.962 0.893    0.00417  0.0120
```

```
best.poly.svm = poly.svm$finalModel
best.poly.svm
```

```
## Support Vector Machine object of class "ksvm"
##
## SV type: C-svc  (classification)
##  parameter : cost C = 0.25
##
## Polynomial kernel function.
##  Hyperparameters : degree =  1  scale =  1  offset =  1
##
## Number of Support Vectors : 1157
```

```
##
## Objective Function Value : -276.1925
## Training error : 0.036786
```

Test - Test Set

```r
predict.res = predict(poly.svm, newdata=fifa.tst)

confusionMatrix(predict.res, fifa.tst$wb_or_b)
```

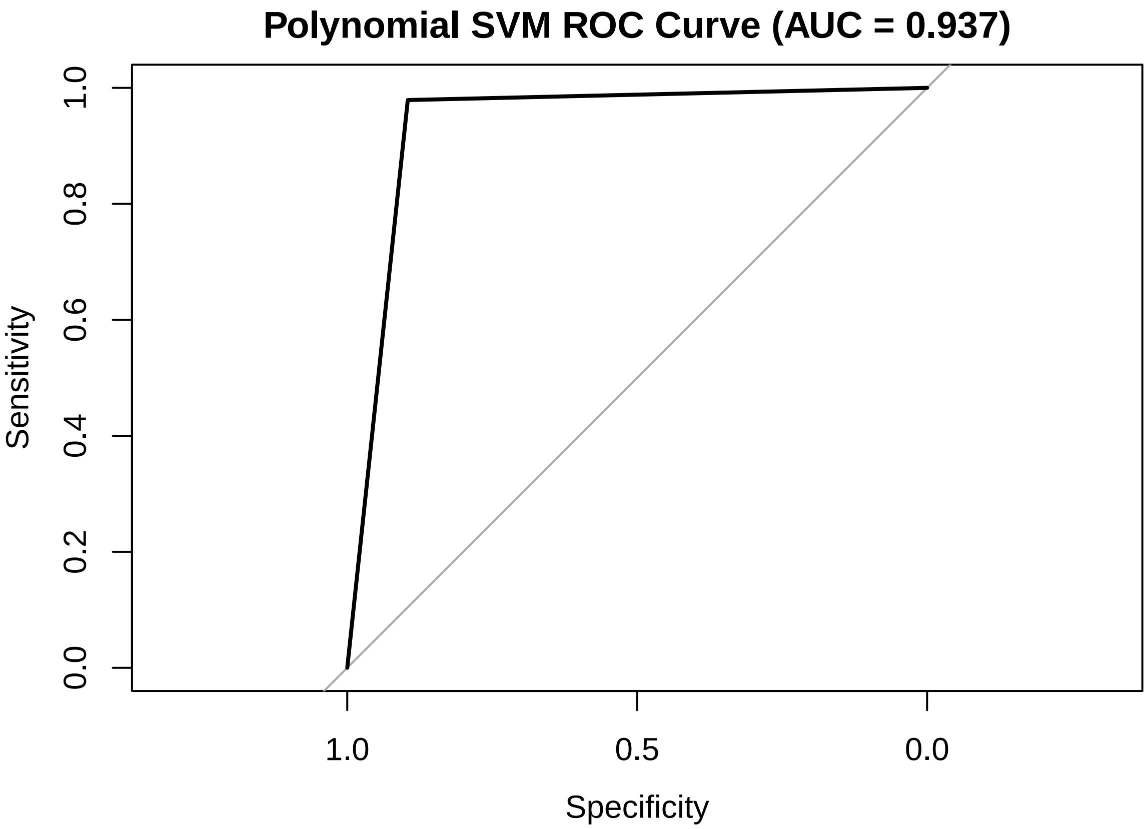
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   wb    b
##         wb 2467   76
##         b    53  652
##
##                Accuracy : 0.9603
##                  95% CI : (0.953, 0.9667)
##     No Information Rate : 0.7759
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.8845
##
##  Mcnemar's Test P-Value : 0.05275
##
##             Sensitivity : 0.9790
##             Specificity : 0.8956
##          Pos Pred Value : 0.9701
##          Neg Pred Value : 0.9248
##              Prevalence : 0.7759
##          Detection Rate : 0.7595
##    Detection Prevalence : 0.7829
##       Balanced Accuracy : 0.9373
##
##        'Positive' Class : wb
##
```

```r
real.wb_or_b_num = ifelse(fifa.tst$wb_or_b == "wb", 1, 0)
pred.wb_or_b_num = ifelse(predict.res == "wb", 1, 0)

roc_obj <- roc(real.wb_or_b_num, pred.wb_or_b_num)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```r
auc <- auc(roc_obj)

plot(roc_obj, main = paste0("Polynomial SVM ROC Curve (AUC = ", round(auc, 3), ")"))
```


Polynomial SVM ROC Curve (AUC = 0.937)
Sensitivity
0.0
0.2
0.4
0.6
0.8
1.0
Specificity
1.0
0.5
0.0