Chp 14 Support Vector Machines

Hands-on Machine Learning with R

Boookclub R-Ladies Utrecht and R-Ladies Den Bosch

Martine Jansen

stRt

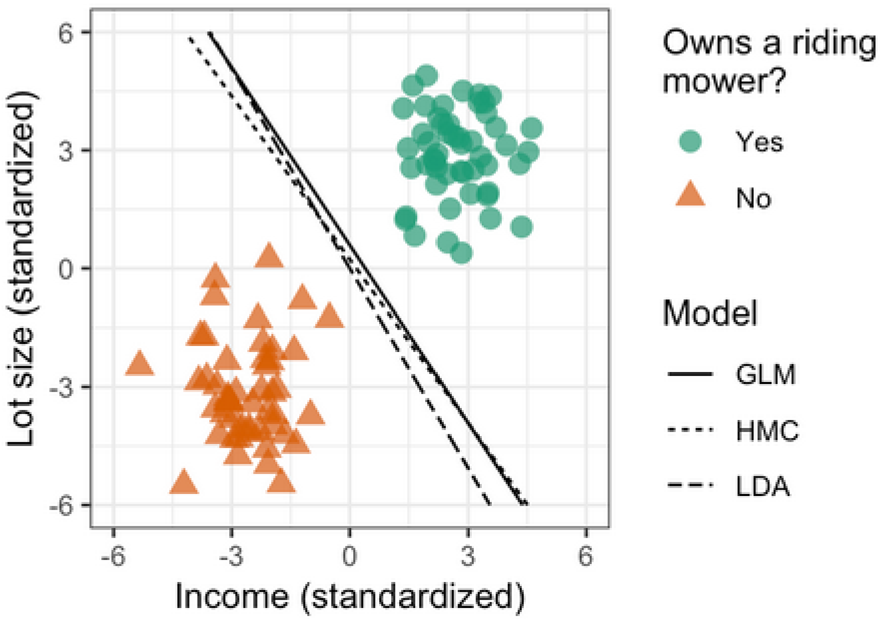
- Organized by @RLadiesUtrecht and @RLadiesDenBosch
- Meet-ups every 2 weeks on "Hands-On Machine Learning with R"
  by Bradley Boehmke and Brandon Greenwell
- No session recording!But we will publish the slides and notes!
- We use HackMD for making shared notes and for the registry:
  https://hackmd.io/rGu7xw2bRS-Im8Iq7-wvXw
- Please keep mic off during presentation. Nice to have camera on and participate to make the meeting more interactive.
- Questions? Raise hand / write question in HackMD or in chat
- Remember presenters are not necessarily also subject experts ☺
- Remember the R-Ladies code of conduct.
  In summary, please be nice to each other and help us make an **inclusive** meeting! ♡

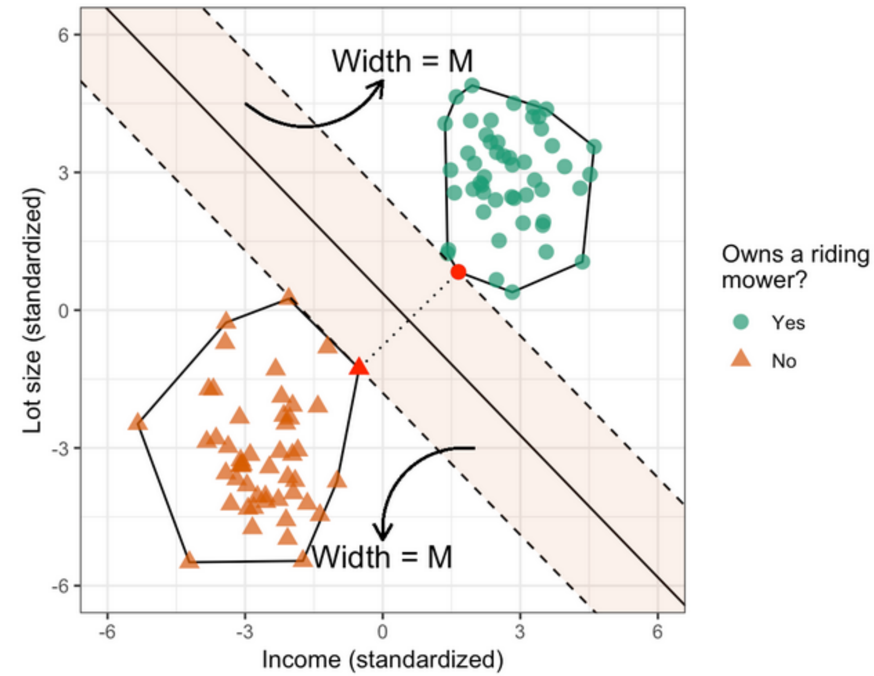
Support Vector Machines

- Supervised Learning Algorithm
- Binary classification by means of separating hyperplane
- Can be extended to more than two classes
- Can also be used for regression

Hyperplane

Hard Margin Classifier

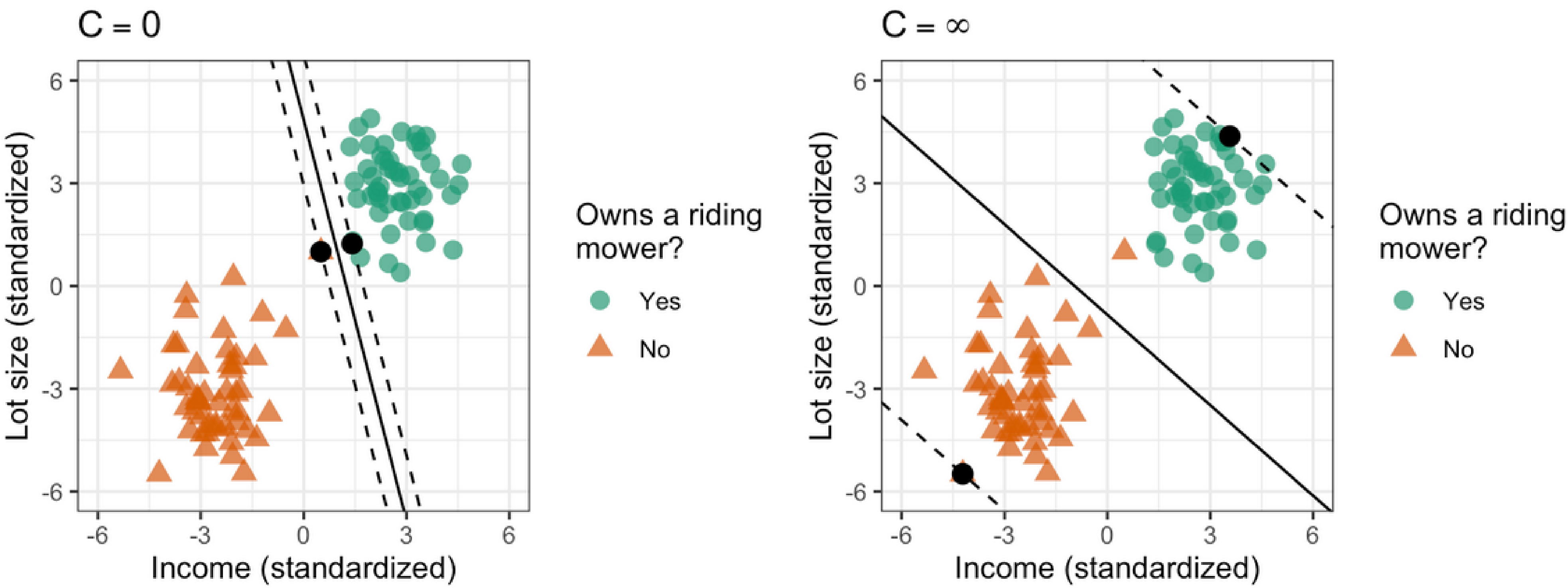




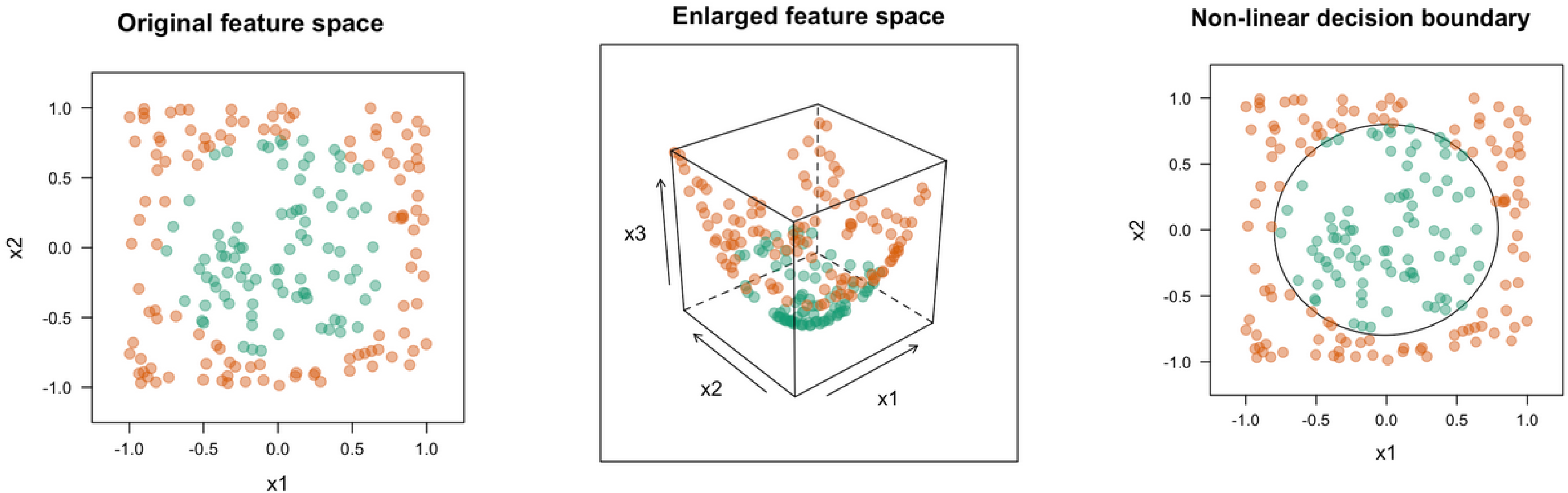
- Boundary with maximum separation (2M) between classes
- Maximizing distance to closest points from either class
- Points on the margin are Support Vectors

Soft Margin Classifier

- Sometimes data are separable by hyperplane
- HMC not robust for noisy data
- Solution:
    - allow points in margin or on wrong side hyperplane
    - budget for wrong points C (hyperparameter $\rightarrow$ tuning)



But then this:



- Enlarge the feature space by adding more features
- Here with $X_3 = X_1^2 + X_2^2$, a polynomial function degree 2
- With new feature space hyperplane still linear, not so in original feature space

Support Vector Machines

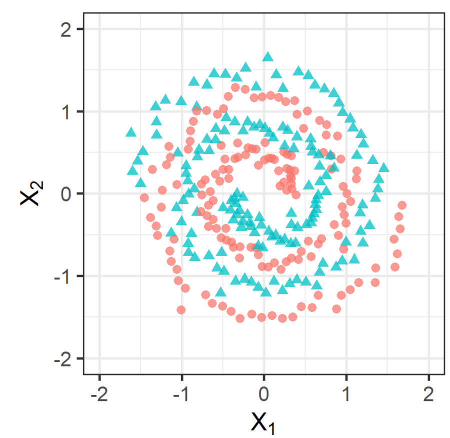
- Enlarge feature space in structured way, with *kernels*
- Polynomial kernel degree d and scale γ:
  - $K(x_i, x_{i'}) = \gamma(1 + \sum_{j=1}^{p} x_{ij}x_{i'j})^d$
- Radial kernel:
  - $K(x_i, x_{i'}) = exp(\gamma \sum_{j=1}^{p}(x_{ij} - x_{i'j})^2)$, with $\gamma = \frac{1}{2\sigma^2}$
- ...
- Hyperparameters found with tuning
- SVMs are extremely flexible and capable of estimating complex nonlinear decision boundaries
- Advice authors: start with radial kernel

Example 1/n

```r
# Libraries needed
library(tidyverse)
library(kernlab)  # fitting SVMs
library(mlbench)  # ML benchmark data set

# Simulate data
set.seed(0841)
spirals <- as.data.frame(
  mlbench.spirals(300,
                  cycles = 2,
                  sd = 0.09))
names(spirals) <- c("x1", "x2", "classes"
head(spirals)
```

```
         x1          x2 classes
1 0.3894633 -0.01786672       1
2 0.2731469  0.04359156       1
3 0.3394452  0.05940869       1
4 0.1959808  0.09491952       1
5 0.2001946 -0.58237355       2
6 0.3331377  0.12047611       1
```

```r
# make a plot
ggplot(spirals, aes(x = x1, y = x2))
  geom_point(aes(shape = classes,
                 color = classes),
             size = 3, alpha = 0.75)
  xlab(expression(X[1])) +
  ylab(expression(X[2])) +
  xlim(-2, 2) +
  ylim(-2, 2) +
  coord_fixed() +
  theme_bw(base_size = 20) +
  theme(legend.position = "none")
```



Example 2/n

```r
# Fit an SVM using a radial basis function kernel
spirals_svm <- ksvm(classes ~ x1 + x2,
                    data = spirals,
                    #Radial Basis kernel "Gaussian":
                    kernel = "rbfdot",
                    C = 500,
                    prob.model = TRUE)
```

```r
spirals_svm
```

```
Support Vector Machine object of class "ksvm"

SV type: C-svc  (classification)
 parameter : cost C = 500

Gaussian Radial Basis kernel function.
 Hyperparameter : sigma =  1.56383735596073

Number of Support Vectors : 82

Objective Function Value : -15568.76
Training error : 0.023333
Probability model included.
```

Example 3/n

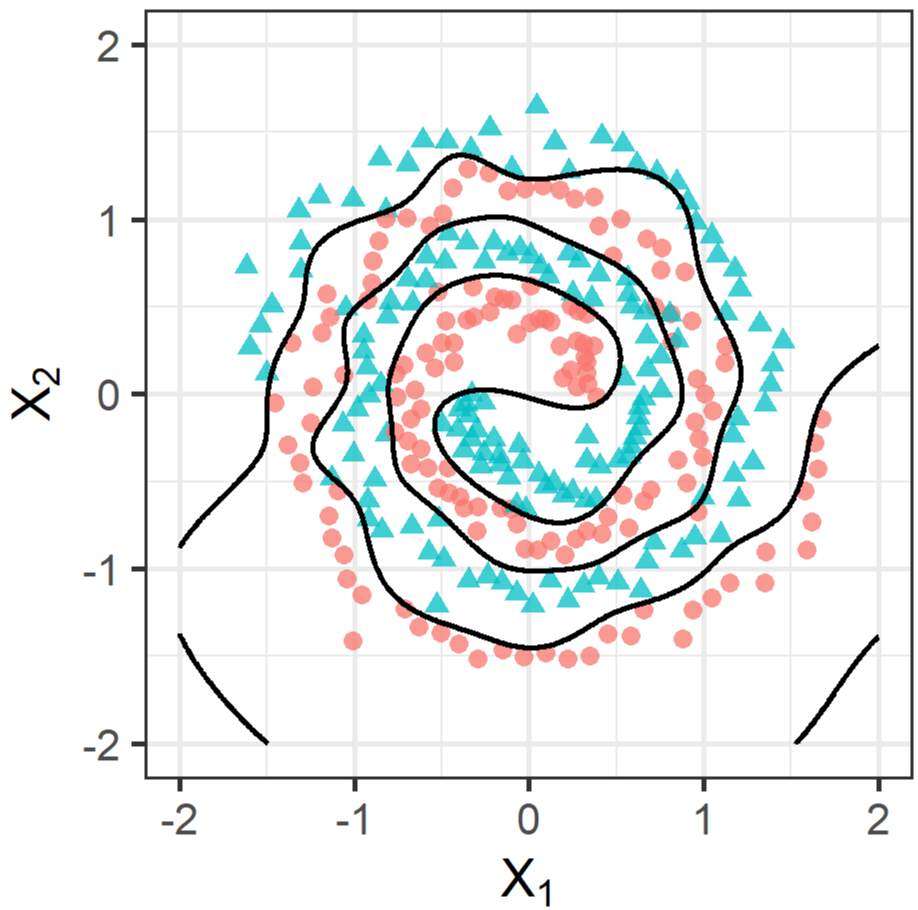
```r
# Grid over which to evaluate decision boundaries
npts <- 500
xgrid <- expand.grid(
  x1 = seq(from = -2, 2, length = npts),
  x2 = seq(from = -2, 2, length = npts)
)

# Predicted probabilities (as a two-column matrix)
prob_svm <- predict(spirals_svm,
                    newdata = xgrid,
                    type = "probabilities")

xgrid2 <- bind_cols(xgrid, prob = prob_svm[,1])
```

```r
head(xgrid2)
```

```
         x1 x2      prob
1 -2.000000 -2 0.2344511
2 -1.991984 -2 0.2353928
3 -1.983968 -2 0.2363789
4 -1.975952 -2 0.2374113
5 -1.967936 -2 0.2384918
6 -1.959920 -2 0.2396225
```

Example 4/n

```r
# Scatterplots with decision boundaries
ggplot(spirals, aes(x = x1, y = x2)) +
  geom_point(aes(shape = classes,
                 color = classes),
             size = 3, alpha = 0.75) +
  xlab(expression(X[1])) +
  ylab(expression(X[2])) +
  xlim(-2, 2) +
  ylim(-2, 2) +
  coord_fixed() +
  theme_bw(base_size = 20) +
  theme(legend.position = "none") +
  stat_contour(data = xgrid2,
               aes(x = x1,
                   y = x2,
                   z = prob),
               linewidth = 1,
               breaks = 0.5,
               color = "black")
```



Extensions

- **More than two classes**
  - One vs all: fit SVM for each class (one class vs the rest), classify to class with largest margin
  - One vs one: fit all pairwise svms (1 vs 2, 1 vs 3, …) and most voted class wins
- **Support vector regression**
  - find a good fitting hyperplane in a kernel-induced feature space that will have good generalization performance using the original features

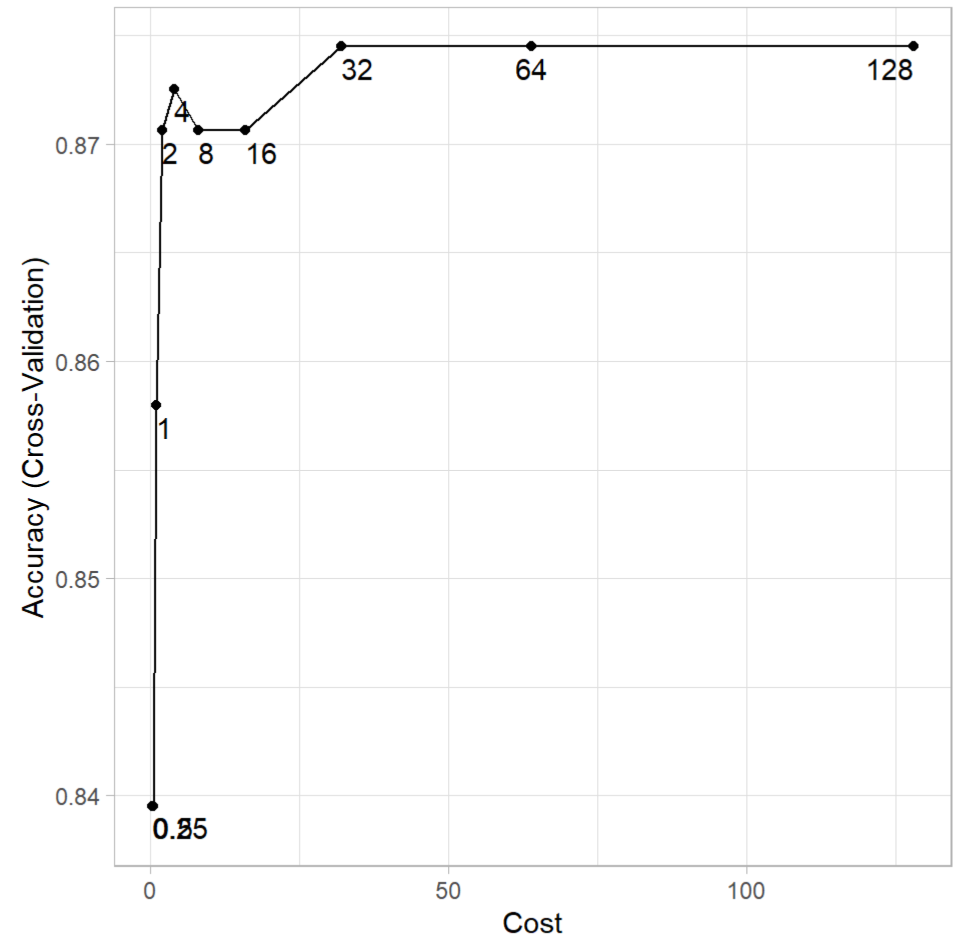
Example Job attrition 1/n

- Intent to predict *Attrition*
- Tune and fit an SVM with a radial kernel (hyperparameters and C)
- K-fold cv, can be time consuming

```
# Load attrition data
df <- modeldata::attrition %>%
  #change all factors to unordered factors
  mutate_if(is.ordered, factor, ordered = FALSE)

# Create training (70%) and test (30%) sets
set.seed(123)  # for reproducibility
library(rsample)
churn_split <- initial_split(df, prop = 0.7, strata = "Attrition")
churn_train <- training(churn_split)
churn_test  <- testing(churn_split)
```

Example Job attrition 2/n

```r
# Tune an SVM with radial basis kernel
library(caret)
set.seed(1854)  # for reproducibility
churn_svm <- caret::train(
  Attrition ~ .,
  data = churn_train,
  method = "svmRadial",
  preProcess = c("center", "scale"),
  trControl = trainControl(method = "cv
                           number = 10)
  tuneLength = 10
)

# different results than in book?
ggplot(churn_svm) +
  geom_text(aes(label = C),
            hjust = "inward",
            nudge_y = -0.001) +
  theme_light()
```



Example Job attrition 3/n

- Probabilities is not naturally for SVM, but can be “estimated”

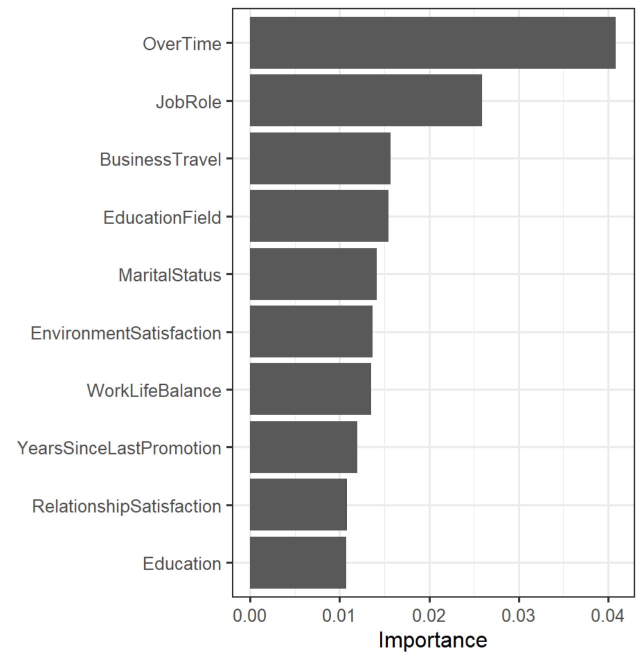
```r
# Control params for SVM
ctrl <- trainControl(method = "cv",  number = 10, classProbs = TRUE,
                     summaryFunction = twoClassSummary ) # needed for AUC/ROC
# Tune an SVM
set.seed(5628)  # for reproducibility
churn_svm_auc <- train(Attrition ~ .,
                       data = churn_train, method = "svmRadial",
                       preProcess = c("center", "scale"),
                       metric = "ROC",  # area under ROC curve (AUC)
                       trControl = ctrl, tuneLength = 10)

churn_svm_auc$results %>% round(4)
```

```
   sigma     C    ROC   Sens   Spec  ROCSD SensSD SpecSD
1 0.0094  0.25 0.8238 0.9641 0.3688 0.0589 0.0149 0.0838
2 0.0094  0.50 0.8240 0.9652 0.3816 0.0588 0.0173 0.0689
3 0.0094  1.00 0.8243 0.9652 0.3757 0.0586 0.0197 0.0946
4 0.0094  2.00 0.8271 0.9791 0.3504 0.0584 0.0092 0.1022
5 0.0094  4.00 0.8234 0.9826 0.3022 0.0583 0.0113 0.0826
6 0.0094  8.00 0.8122 0.9837 0.3129 0.0543 0.0098 0.1370
7 0.0094 16.00 0.7957 0.9837 0.2827 0.0532 0.0098 0.1288
8 0.0094 32.00 0.7864 0.9826 0.3018 0.0537 0.0147 0.1380
```

Feature Interpretation

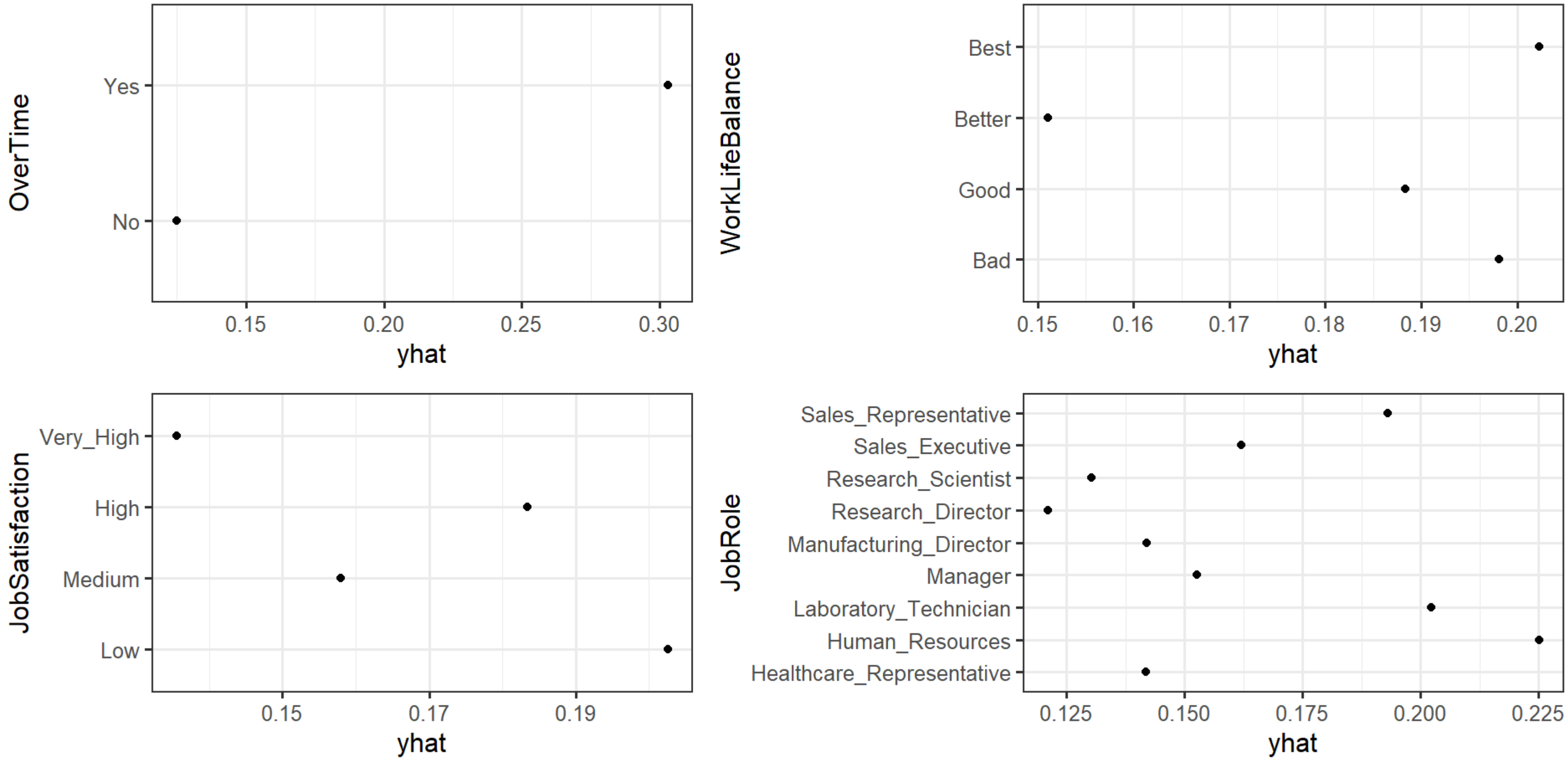
- SVMs do not emit any natural measures of feature importance
- We can use `vip()`

```r
# We want reference class "Yes"
# Make function returning prob of "Yes"
prob_yes <- function(object, newdata) {
  predict(object, newdata = newdata, type = "prob")[,
}

set.seed(2827)  # for reproducibility
vip::vip(churn_svm_auc,
         method = "permute",
         nsim = 5,
         train = churn_train,
         target = "Attrition",
         metric = "auc",
         reference_class = "Yes",
         pred_wrapper = prob_yes) +
  theme_bw(base_size = 12)
```



PDP

```r
ppdp <- function(x){pdp::partial(churn_svm_auc, pred.var = x, which.class = 2,
                                  prob = TRUE, plot = TRUE, plot.engine = "ggplot2") +
  coord_flip() + theme_bw(base_size = 12)
}
#library(patchwork)
ppdp("OverTime") +  ppdp("WorkLifeBalance") + ppdp("JobSatisfaction") +
  ppdp("JobRole") + plot_layout(ncol = 2)
```


OverTime
Yes
No
0.15
0.20
0.25
0.30
yhat
WorkLifeBalance
Best
Better
Good
Bad
0.15
0.16
0.17
0.18
0.19
0.20
yhat
JobSatisfaction
Very_High
High
Medium
Low
0.15
0.17
0.19
yhat
JobRole
Sales_Representative
Sales_Executive
Research_Scientist
Research_Director
Manufacturing_Director
Manager
Laboratory_Technician
Human_Resources
Healthcare_Representative
0.125
0.150
0.175
0.200
0.225
yhat

The end