

Chp 14 Support Vector Machines

Hands-on Machine Learning with R
Boookclub R-Ladies Utrecht and R-Ladies Den Bosch

Martine Jansen

- 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/3rAbHxj5RqSaUAIE-shY2A>
- 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! ❤️

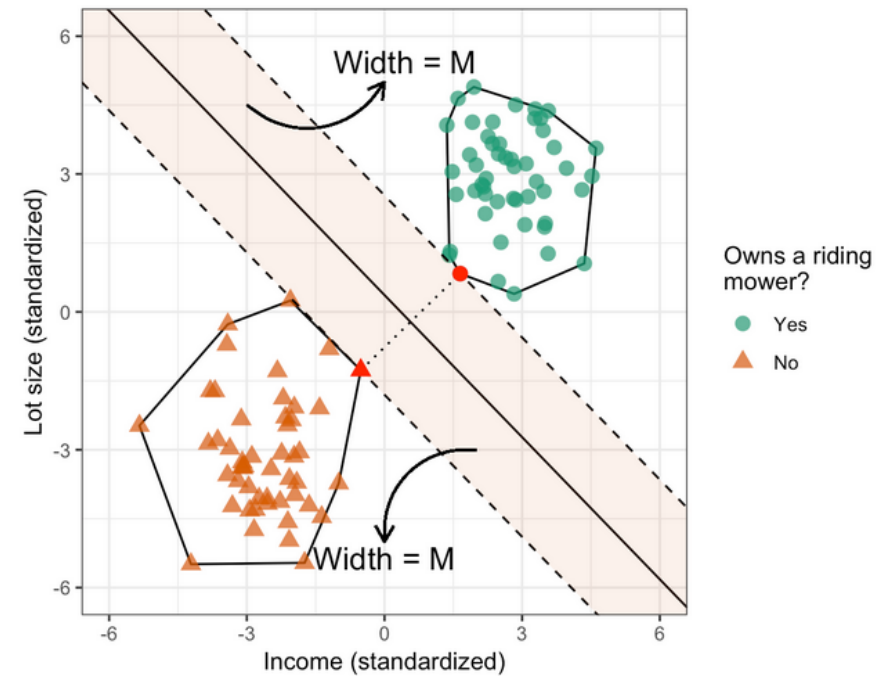
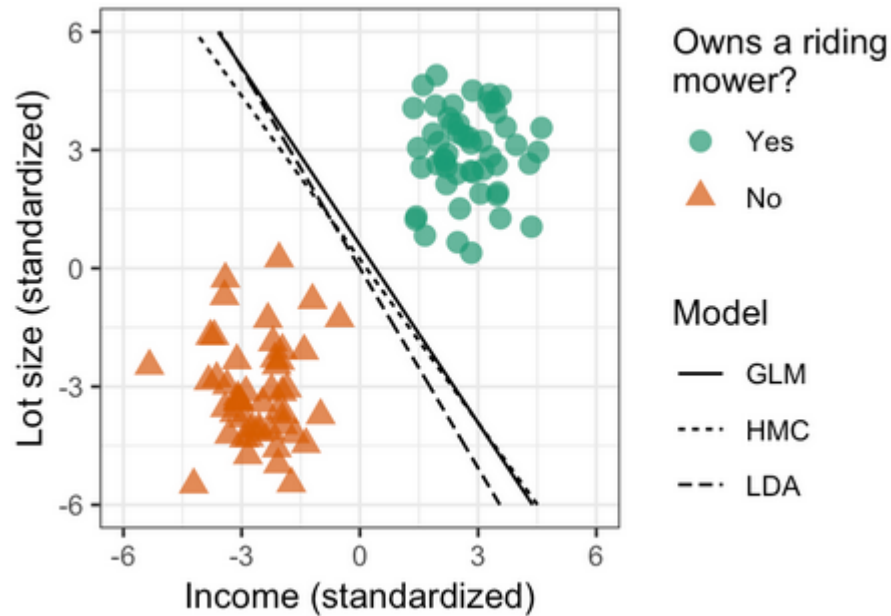
Support Vector Machines

- 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



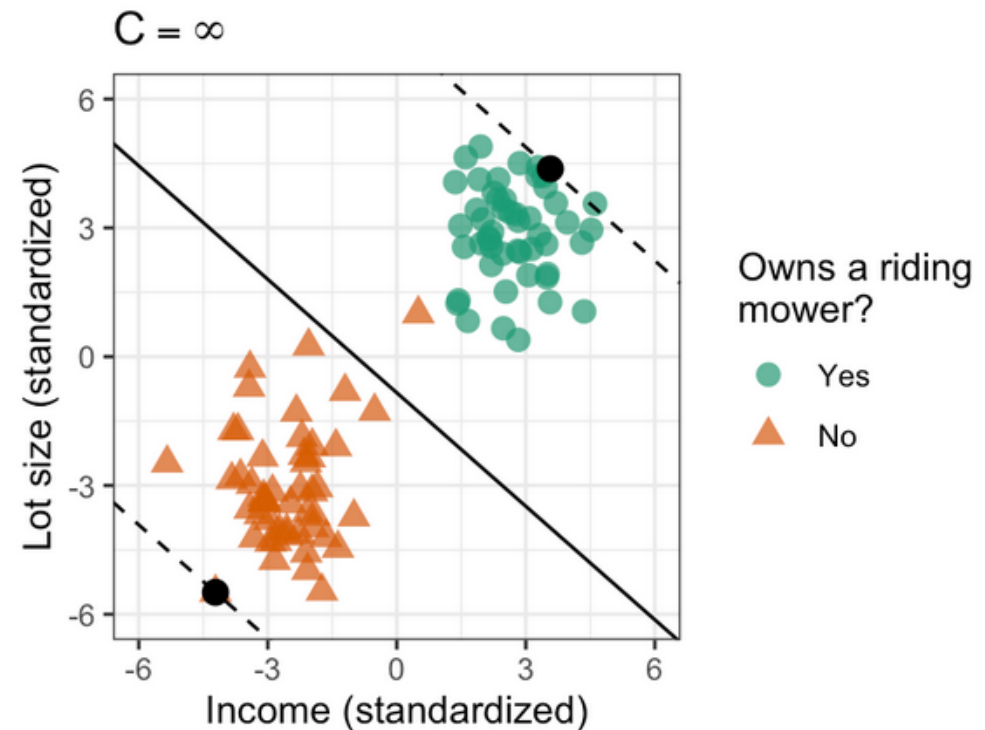
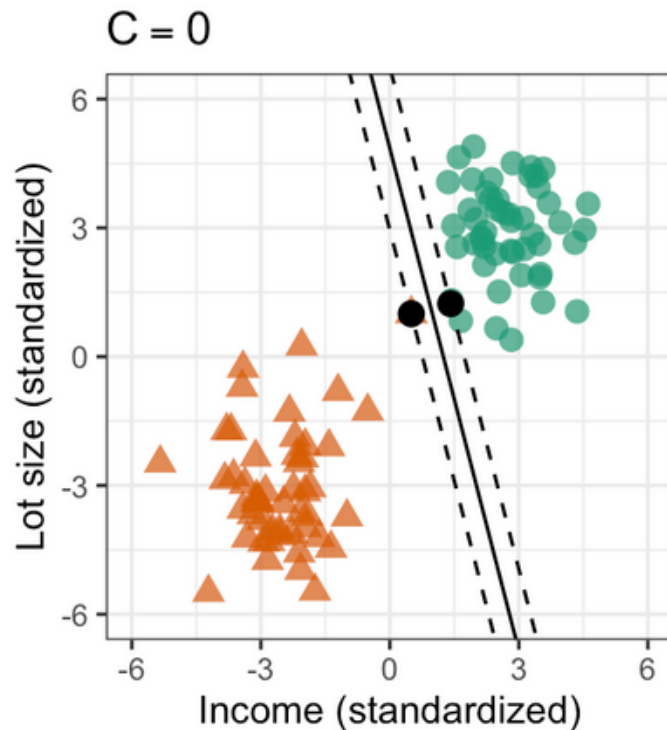
Hard Margin Classifier



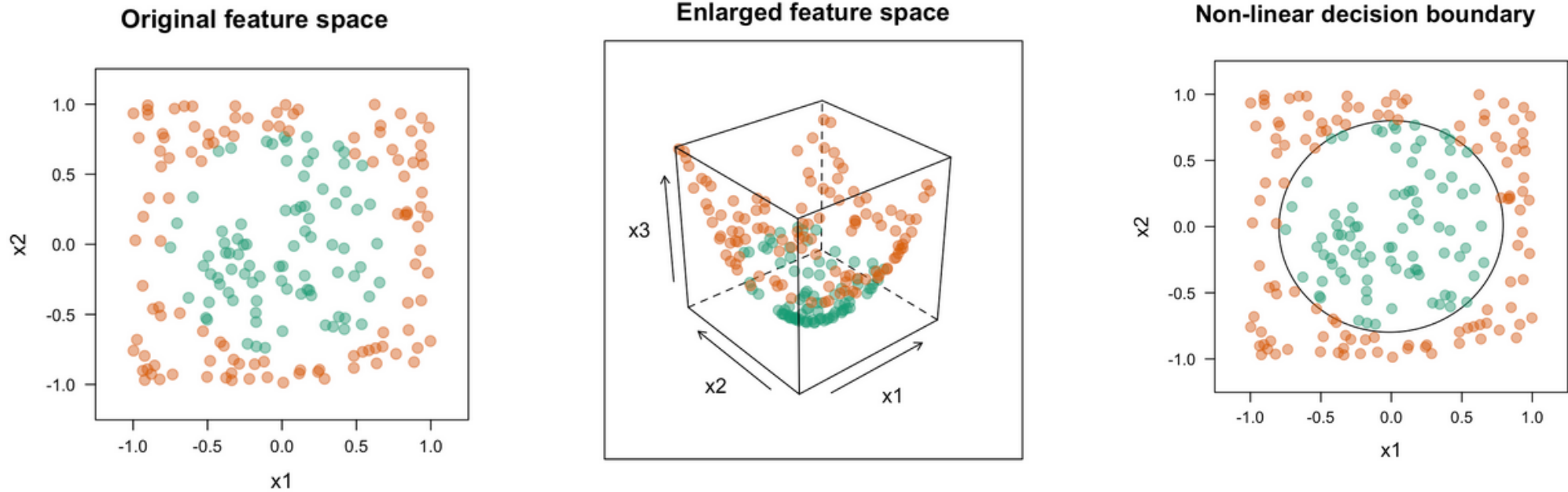
- Boundary with maximum separation ($2M$) between classes
- Maximizing distance to closest points from either class
- Points on the margin are Support Vectors

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

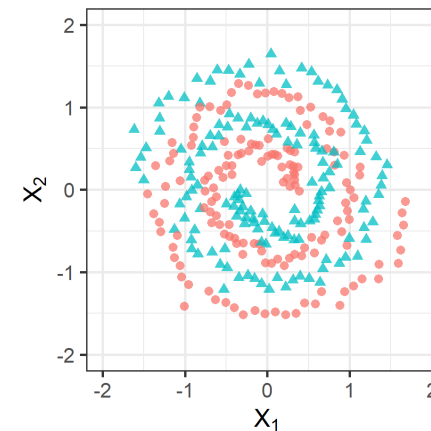
- Enlarge feature space in structured way, with *kernels*
- Polynomial kernel degree d and scale γ :
 - $K(x_i, x_{i'}) = \gamma(1 + \sum_{j=1}^p x_{ij}x_{i'j})^d$
- Radial kernel:
 - $K(x_i, x_{i'}) = \exp(\gamma \sum_{j=1}^p (x_{ij} - x_{i'j})^2)$, with $\gamma = \frac{1}{2\sigma^2}$
- ...
- Hyperparameters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

Example 1/n

```
1 # Libraries needed
2 library(tidyverse)
3 library(kernlab) # fitting SVMs
4 library(mlbench) # ML benchmark data set
5
6 # Simulate data
7 set.seed(0841)
8 spirals <- as.data.frame(
9   mlbench.spirals(300,
10     cycles = 2,
11     sd = 0.09))
12 names(spirals) <- c("x1", "x2", "classes")
13 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

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



Example 2/n

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

```
1 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

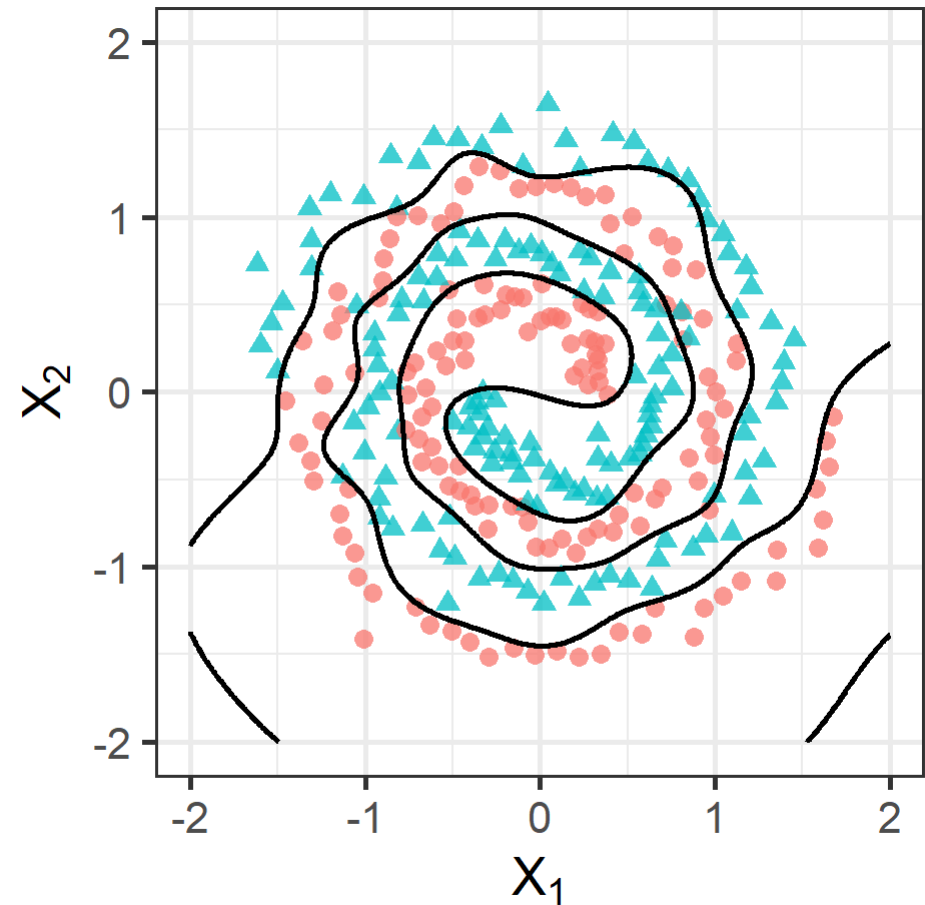
```
1 # Grid over which to evaluate decision boundaries
2 npts <- 500
3 xgrid <- expand.grid(
4   x1 = seq(from = -2, 2, length = npts),
5   x2 = seq(from = -2, 2, length = npts)
6 )
7
8 # Predicted probabilities (as a two-column matrix)
9 prob_svm <- predict(spirals_svm,
10                    newdata = xgrid,
11                    type = "probabilities")
12
13 xgrid2 <- bind_cols(xgrid, prob = prob_svm[,1])
```

```
1 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

```
1 # Scatterplots with decision boundaries
2 ggplot(spirals, aes(x = x1, y = x2)) +
3   geom_point(aes(shape = classes,
4                 color = classes),
5             size = 3, alpha = 0.75) +
6   xlab(expression(X[1])) +
7   ylab(expression(X[2])) +
8   xlim(-2, 2) +
9   ylim(-2, 2) +
10  coord_fixed() +
11  theme_bw(base_size = 20) +
12  theme(legend.position = "none") +
13  stat_contour(data = xgrid2,
14             aes(x = x1,
15               y = x2,
16               z = prob),
17             linewidth = 1,
18             breaks = 0.5,
19             color = "black")
```



- **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

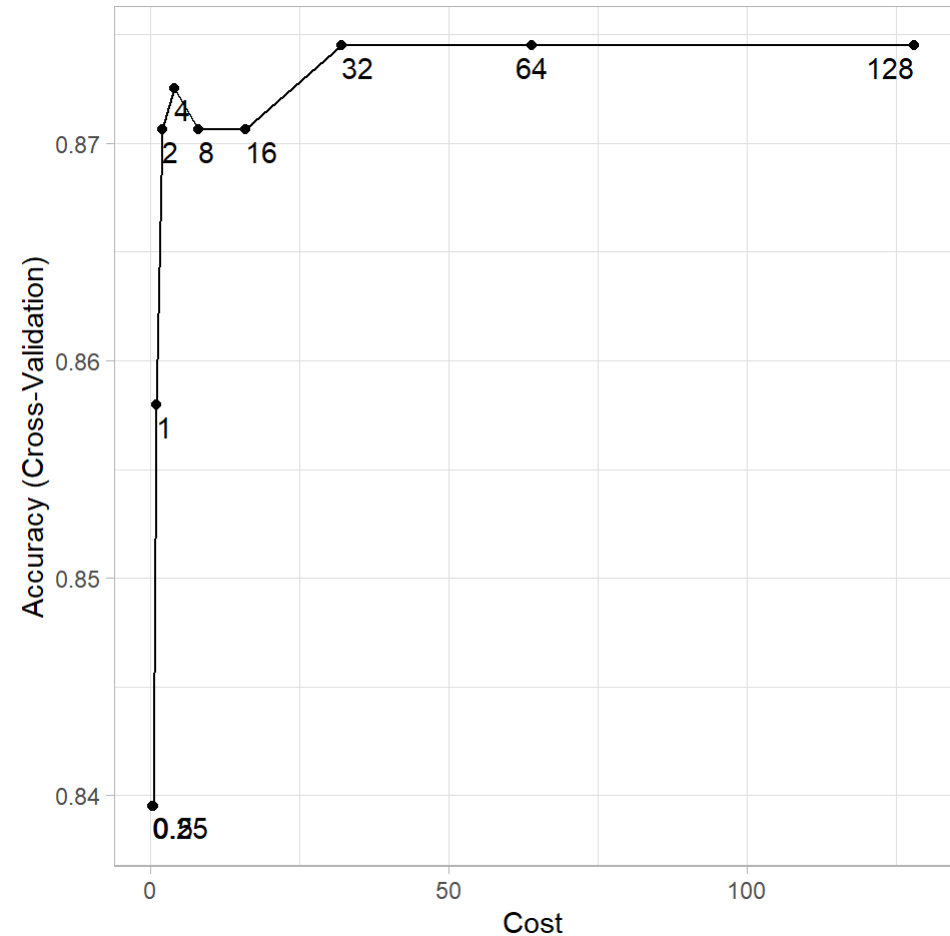
Example Job attrition 1/n

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

```
1 # Load attrition data
2 df <- modeldata::attrition %>%
3   #change all factors to unordered factors
4   mutate_if(is.ordered, factor, ordered = FALSE)
5
6 # Create training (70%) and test (30%) sets
7 set.seed(123) # for reproducibility
8 library(rsample)
9 churn_split <- initial_split(df, prop = 0.7, strata = "Attrition")
10 churn_train <- training(churn_split)
11 churn_test  <- testing(churn_split)
```

Example Job attrition 2/n

```
1 # Tune an SVM with radial basis kernel
2 library(caret)
3 set.seed(1854) # for reproducibility
4 churn_svm <- caret::train(
5   Attrition ~ .,
6   data = churn_train,
7   method = "svmRadial",
8   preProcess = c("center", "scale"),
9   trControl = trainControl(method = "cv",
10                             number = 10)
11   tuneLength = 10
12 )
13
14 # different results than in book?
15 ggplot(churn_svm) +
16   geom_text(aes(label = C),
17             hjust = "inward",
18             nudge_y = -0.001) +
19   theme_light()
```



Example Job attrition 3/n

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

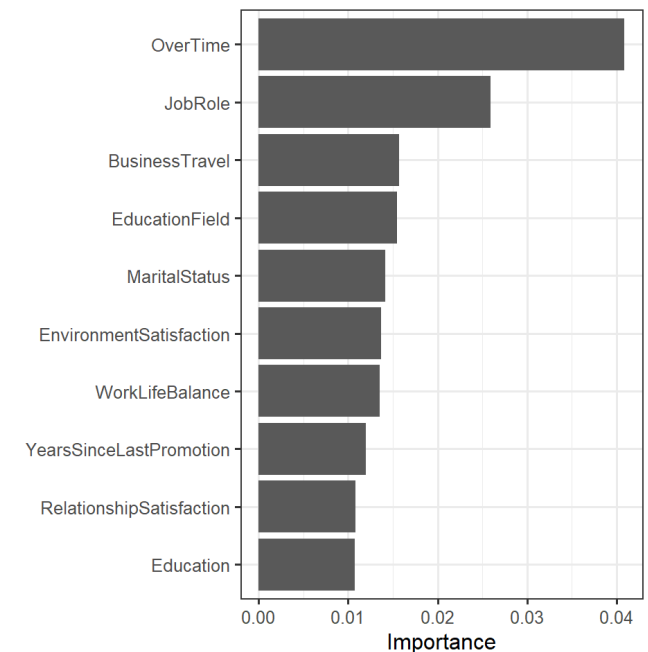
```
1 # Control params for SVM
2 ctrl <- trainControl(method = "cv", number = 10, classProbs = TRUE,
3                       summaryFunction = twoClassSummary ) # needed for AUC/ROC
4 # Tune an SVM
5 set.seed(5628) # for reproducibility
6 churn_svm_auc <- train(Attrition ~ .,
7                        data = churn_train, method = "svmRadial",
8                        preProcess = c("center", "scale"),
9                        metric = "ROC", # area under ROC curve (AUC)
10                       trControl = ctrl, tuneLength = 10)
11
12 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

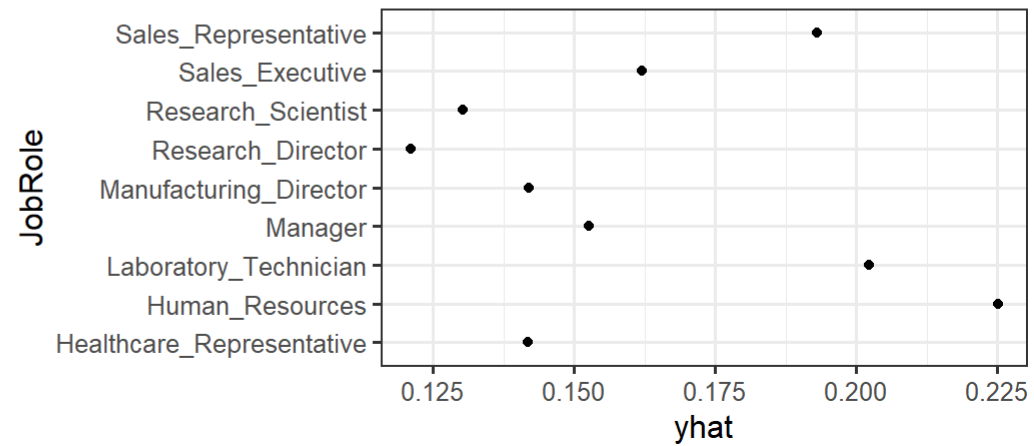
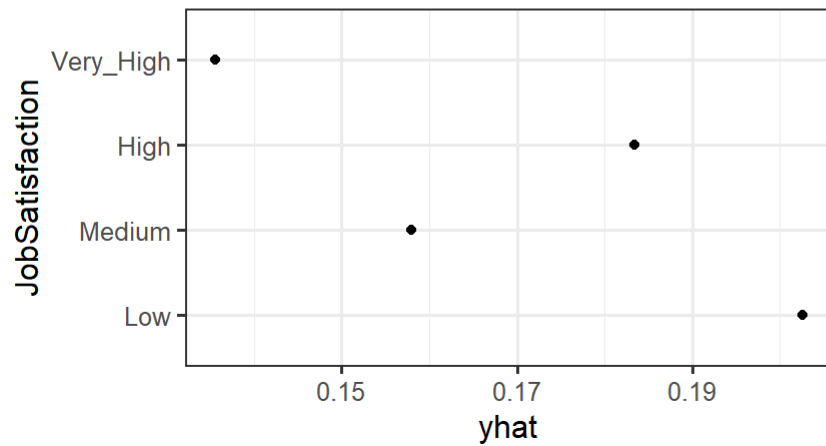
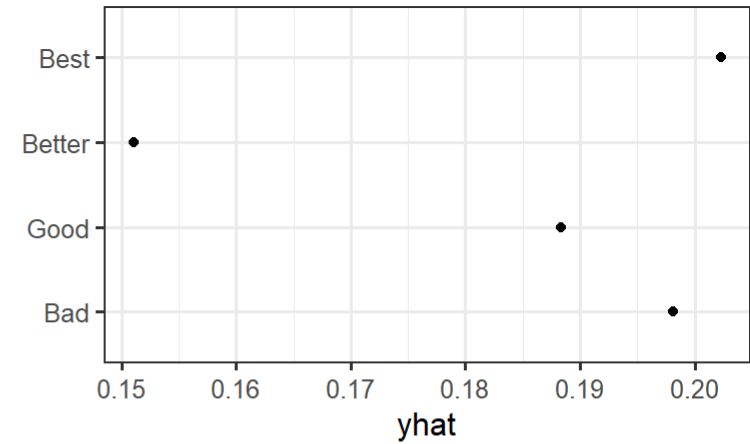
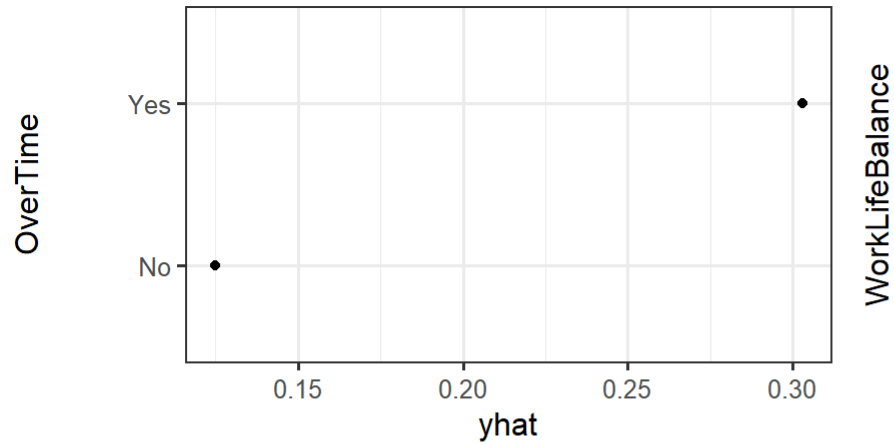
Feature Interpretation

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

```
1 # We want reference class "Yes"
2 # Make function returning prob of "Yes"
3 prob_yes <- function(object, newdata) {
4   predict(object, newdata = newdata, type = "prob")[,
5 ]
6
7 set.seed(2827) # for reproducibility
8 vip::vip(churn_svm_auc,
9   method = "permute",
10  nsim = 5,
11  train = churn_train,
12  target = "Attrition",
13  metric = "auc",
14  reference_class = "Yes",
15  pred_wrapper = prob_yes) +
16  theme_bw(base_size = 12)
```



```
1 ppdp <- function(x){pdp::partial(churn_svm_auc, pred.var = x, which.class = 2,  
2                                prob = TRUE, plot = TRUE, plot.engine = "ggplot2") +  
3    coord_flip() + theme_bw(base_size = 12)  
4 }  
5 #library(patchwork)  
6 ppdp("OverTime") + ppdp("WorkLifeBalance") + ppdp("JobSatisfaction") +  
7   ppdp("JobRole") + plot_layout(ncol = 2)  
8  
9 modeldata::attrition
```



	Age	Attrition	BusinessTravel	DailyRate	Department
1	41	Yes	Travel_Rarely	1102	Sales
2	49	No	Travel_Frequently	279	Research_Development
4	37	Yes	Travel_Rarely	1373	Research_Development
5	33	No	Travel_Frequently	1392	Research_Development
7	27	No	Travel_Rarely	591	Research_Development
8	32	No	Travel_Frequently	1005	Research_Development
10	59	No	Travel_Rarely	1324	Research_Development
11	30	No	Travel_Rarely	1358	Research_Development

12	38	No	Travel_Frequently	216	Research_Development
13	36	No	Travel_Rarely	1299	Research_Development
14	35	No	Travel_Rarely	809	Research_Development
15	29	No	Travel_Rarely	153	Research_Development
16	31	No	Travel_Rarely	670	Research_Development
18	34	No	Travel_Rarely	1346	Research_Development
19	28	Yes	Travel_Rarely	103	Research_Development
20	29	No	Travel_Rarely	1389	Research_Development

The end

