Support Vector Machines

SYS 6018 | Spring 2021

svm.pdf

Contents

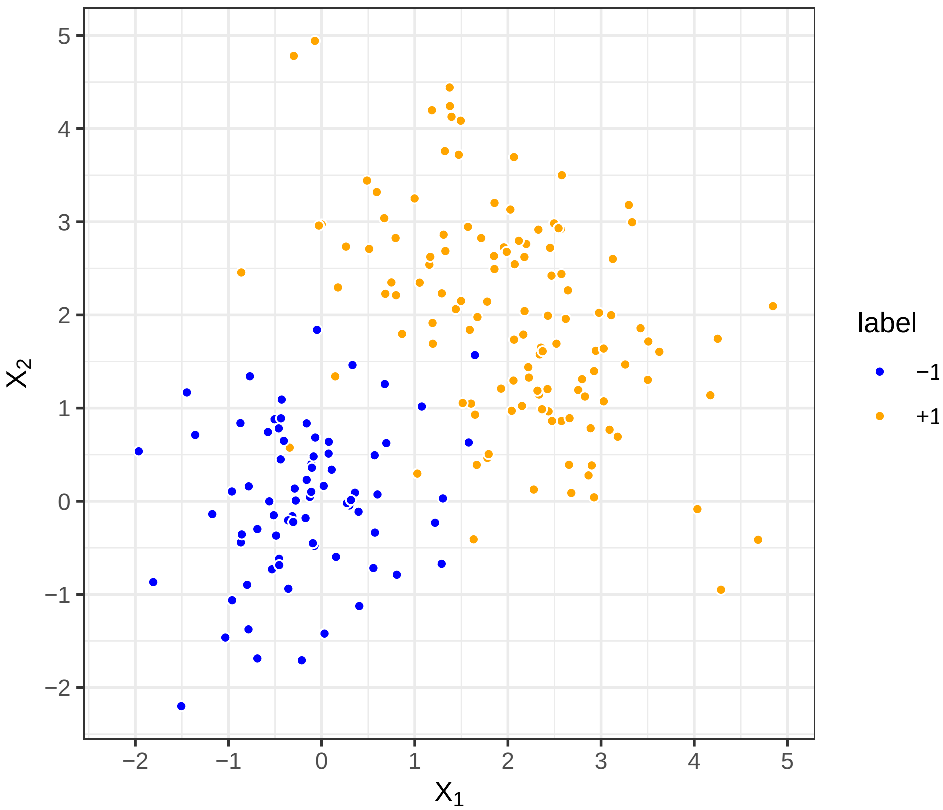
1 Support Vector Machines (SVM) Introduction

1.1 Required R Packages

We will be using the R packages of:

- `tidyverse` for data manipulation and visualization
- `e1071` for the `svm()` functions

1.2 Example



1.3 SVM as Loss + Penalty

Data: $\{(\tilde{y}_i, \mathbf{x}_i)\}_{i=1}^n$

- $\tilde{y}_i \in \{-1, +1\}$
- $\mathbf{x}_i^\mathsf{T} = [x_{i1}, x_{i2}, \ldots, x_{in}] \in \mathbb{R}^p$

Predictor Function: $f(x)$

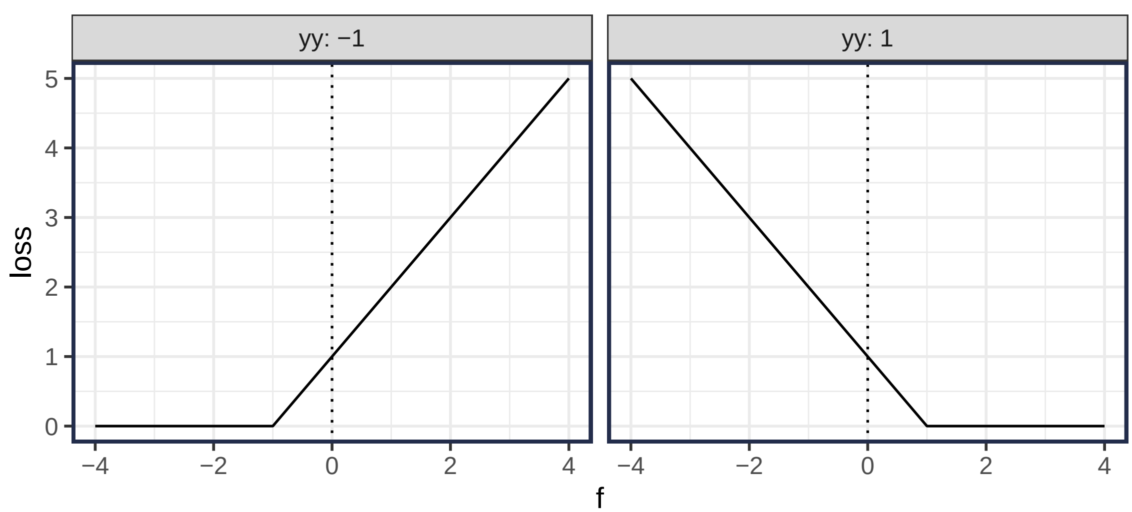
- Linear: $f(\mathbf{x}; \beta) = \beta_0 + \sum_{j=1}^p x_j \beta_j$
- Basis Expansion: $f(\mathbf{x}; \beta) = \beta_0 + \sum_{j=1}^d h_j(\mathbf{x}) \beta_j$
  - $h_j(\mathbf{x})$ transforms the raw x vector
  - Polynomial example: $h_1(\mathbf{x}) = x_1$, $h_2(\mathbf{x}) = x_1^2$, $h_3(\mathbf{x}) = x_2$, $h_4(\mathbf{x}) = x_2^2$, $h_5(\mathbf{x}) = x_1 x_2$
  - This is the connection to *kernels* that we will discuss later
- Let $f_i = f(\mathbf{x}_i)$

Classification:

- If $\hat{f}_i \geq 0$, then label as class $+1$
- If $\hat{f}_i < 0$, then label as class -1

Loss Function: Hinge Loss

$$L(\tilde{y}_i, f_i) = \max\{0, 1 - \tilde{y}_i f_i\}$$



Penalty Function: Ridge Penalty

$$\begin{aligned} P_\lambda(\beta) &= \frac{\lambda}{2} \sum_{j=1}^d \beta_j^2 \\ &= \lambda \|\beta\|^2 / 2 \end{aligned}$$

- This should have you thinking that the $\mathbf{x}$'s should be scaled!

Summary of SVM

1. Estimate *model parameters* β

$$
\begin{aligned}
\hat{\beta}_\lambda &= \arg\min_\beta \left\{ \sum_{i=1}^{n} \max\{0, 1 - \tilde{y}_i f_i(\beta)\} + \lambda \left\| \beta \right\|^2 / 2 \right\} \\
&= \arg\min_\beta \left\{ \text{Hinge Loss}(\beta) + \lambda \text{ Penalty}(\beta) \right\}
\end{aligned}
$$

2. Label Class $+1$ if $\hat{f}_i \geq 0$, else label Class -1
   - This implicitly assumes a 0-1 loss (or equal cost FP and FN)
   - More generally, use threshold t that considers the costs of FP and FN
3. **Tuning Parameters**: besides the λ, SVM will also have tuning parameters related to the kernels (more to come on this).

1.4 SVM vs. Logistic Regression

The *linear* SVM is actually very similar to Logistic Regression (with Ridge Penalty).

Let $f_i(\beta) = \beta_0 + \sum_{j=1}^{p} x_j \beta_j$ be the log-odds.

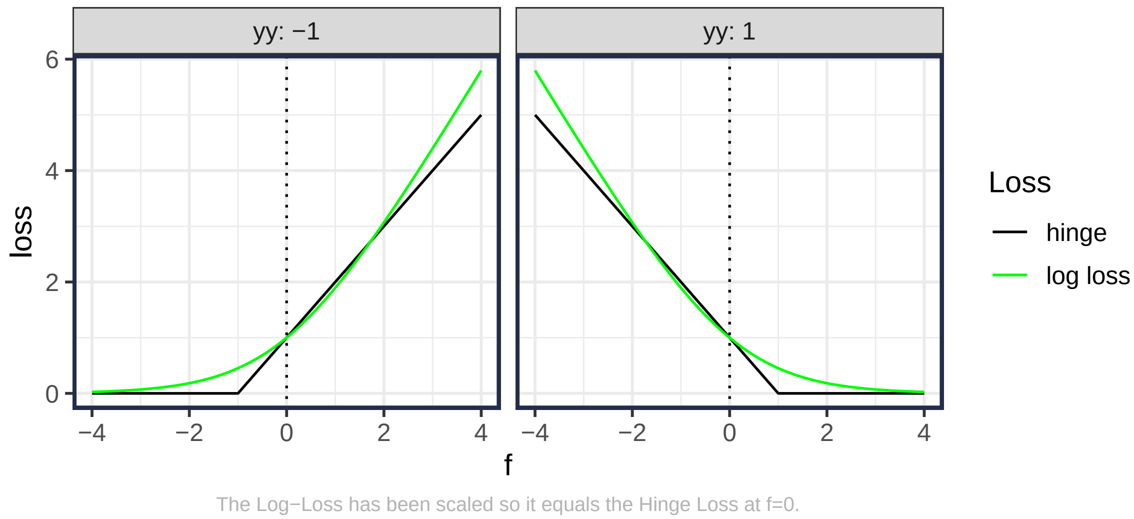
- We also referred to this as $\gamma(\mathbf{x}_i)$ in previous notes.

1. Estimate *model parameters* β
   - This assumes standardized $\mathbf{x}$'s

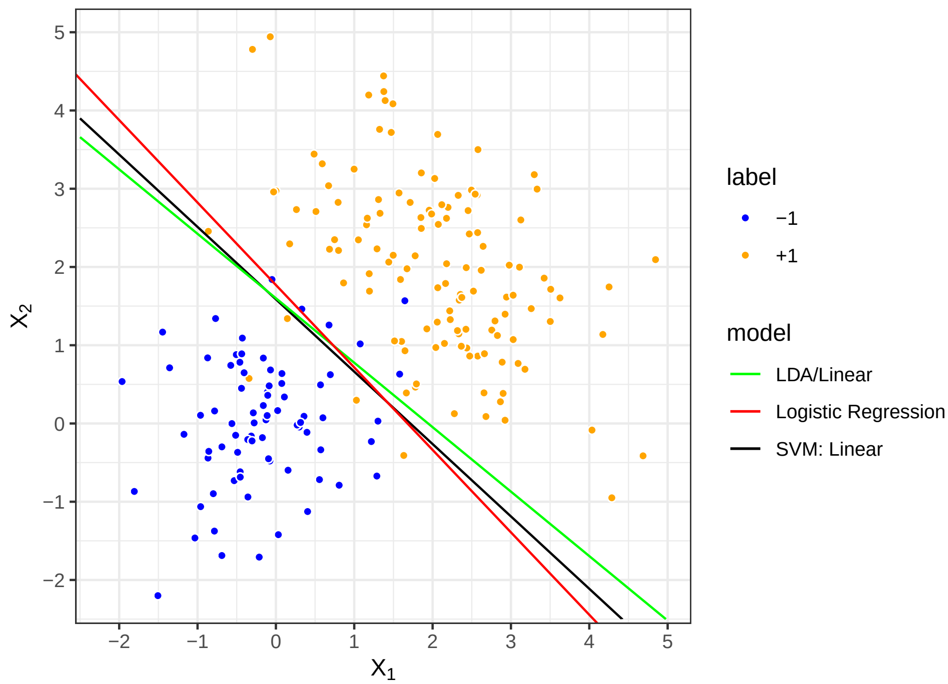
$$
\begin{aligned}
\hat{\beta}_\lambda &= \arg\min_\beta \left\{ \sum_{i=1}^{n} \log\left(1 + \exp\left(-\tilde{y}_i f_i(\beta)\right)\right) + \lambda \left\| \beta \right\|^2 / 2 \right\} \\
&= \arg\min_\beta \left\{ \text{Log Loss}(\beta) + \lambda \text{ Penalty}(\beta) \right\}
\end{aligned}
$$

2. Label Class $+1$ if $\hat{f}_i \geq t$, else label Class -1
   - For some threshold t that considers the costs of FP and FN
   - If Cost(FP) = Cost(FN), the set $t = 0$, which is equivalent to $p(x) = 1/2$

Details of the Logistic Regression Loss Function



1.5 Example: Compare Linear Classifiers



1.6 Three Representations of the Optimization Problem

1. **Penalized optimization**

$$
\begin{aligned}
\hat{\beta} &= \arg\min_{\beta} \{\text{Loss}(\beta) + \lambda \, \text{Penalty}(\beta)\} \\
&= \arg\min_{\beta} \{C \, \text{Loss}(\beta) + \text{Penalty}(\beta)\}
\end{aligned}
$$

   - where $C = \frac{1}{\lambda} > 0$ is an alternative strength of penalty
   - In SVM, C is referred to as the *cost*

2. **Constraint on Penalty**

$$
\begin{aligned}
\hat{\beta} &= \arg\min_{\beta} \; \text{Loss}(\beta) \qquad \text{subject to } \; \text{Penalty}(\beta) \leq t \\
&= \mathop{\arg\min}_{\beta:\, \text{Penalty}(\beta) \leq t} \; \text{Loss}(\beta)
\end{aligned}
$$

3. **Constraint on Loss**

$$
\begin{aligned}
\hat{\beta} &= \arg\min_{\beta} \; \text{Penalty}(\beta) \qquad \text{subject to } \; \text{Loss}(\beta) \leq M \\
&= \mathop{\arg\min}_{\beta:\, \text{Loss}(\beta) \leq M} \; \text{Penalty}(\beta)
\end{aligned}
$$

There is a one-to-one relationship between all tuning parameters: λ, C, t, M.

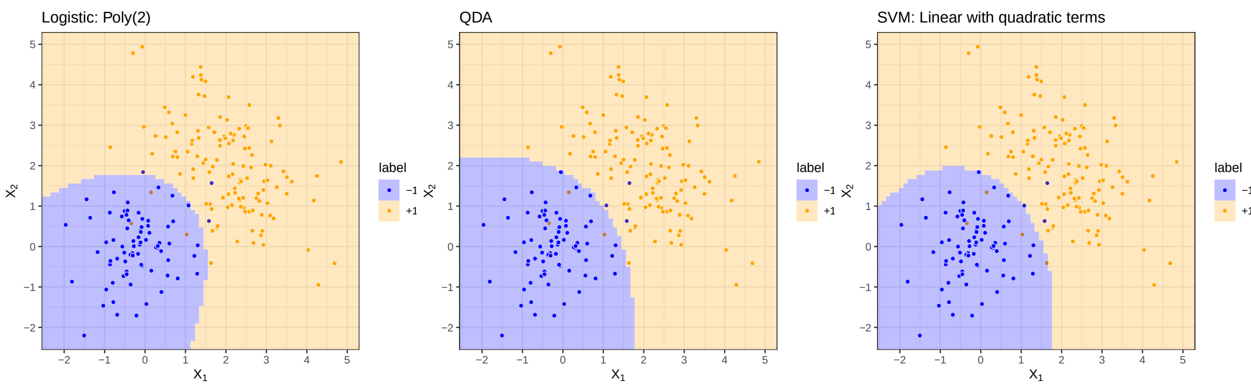
2 Kernels and Non-linear SVM

2.1 Basis Expansion

- Linear: $f(\mathbf{x}; \beta) = \beta_0 + \sum_{j=1}^{p} x_j \beta_j$
- Basis Expansion: $f(\mathbf{x}; \beta) = \beta_0 + \sum_{j=1}^{d} h_j(\mathbf{x}) \beta_j$
  - $h_j(\mathbf{x})$ transforms the raw $\mathbf{x}$ vector

2.1.1 Polynomial Expansion: Quadratic Terms

- Polynomial example: $h_1(\mathbf{x}) = x_1$, $h_2(\mathbf{x}) = x_1^2$, $h_3(\mathbf{x}) = x_2$, $h_4(\mathbf{x}) = x_2^2$, $h_5(\mathbf{x}) = x_1 x_2$
- $f(\mathbf{x}; \beta) = \beta_0 + \sum_{j=1}^{5} h_j(\mathbf{x}) \beta_j$



2.2 Alternative (Dual) Formulation

It turns out that the SVM solution for the model coefficients β can be written as

$$\hat{\beta}_j = \sum_{i=1}^{n} \hat{\alpha}_i \, y_i \, h_j(\mathbf{x}_i)$$

- See ESL Eq. 12.17 if you are interested in the details.
- In this reformulation, the $\{\hat{a}_i\}_{i=1}^{n}$ become the model parameters.
  - There is one model parameter for each observation!
  - $0 \leq \alpha_i \leq C$ (or $0 \leq \alpha_i \leq \frac{1}{\lambda}$)
  - But SVM will force most values to 0.
  - The observations with $\alpha_i > 0$ are called the *support vectors*
    * So the entire SVM model is a function of the support vectors only!
    * The support vectors are the observations on the wrong side of the margin.
- The decision function $\hat{f}(\mathbf{x})$ can be re-written

$$\begin{aligned}
\hat{f}(\mathbf{x}) &= f(\mathbf{x}; \hat{\beta}) \\
&= \hat{\beta}_0 + \sum_{j=1}^{d} h_j(\mathbf{x})\hat{\beta}_j \\
&= \hat{\beta}_0 + \sum_{j=1}^{d} h_j(\mathbf{x}) \left[\sum_{i=1}^{n} \hat{\alpha}_i \, y_i \, h_j(\mathbf{x}_i) \right] \\
&= \hat{\beta}_0 + \sum_{i=1}^{n} \hat{\alpha}_i \, y_i \left[\sum_{j=1}^{d} h_j(\mathbf{x}) h_j(\mathbf{x}_i) \right] \\
&= \hat{\beta}_0 + \sum_{i=1}^{n} \hat{\alpha}_i \, y_i \, K(\mathbf{x}, \mathbf{x}_i)
\end{aligned}$$

  where

$$K(\mathbf{x}, \mathbf{x}_i) = \sum_{j=1}^{d} h_j(\mathbf{x}) h_j(\mathbf{x}_i) = \langle h(\mathbf{x}), h(\mathbf{x}_i) \rangle$$

  is called a **kernel** and measures the inner product, or *similarity* between x and x_i (observation i).

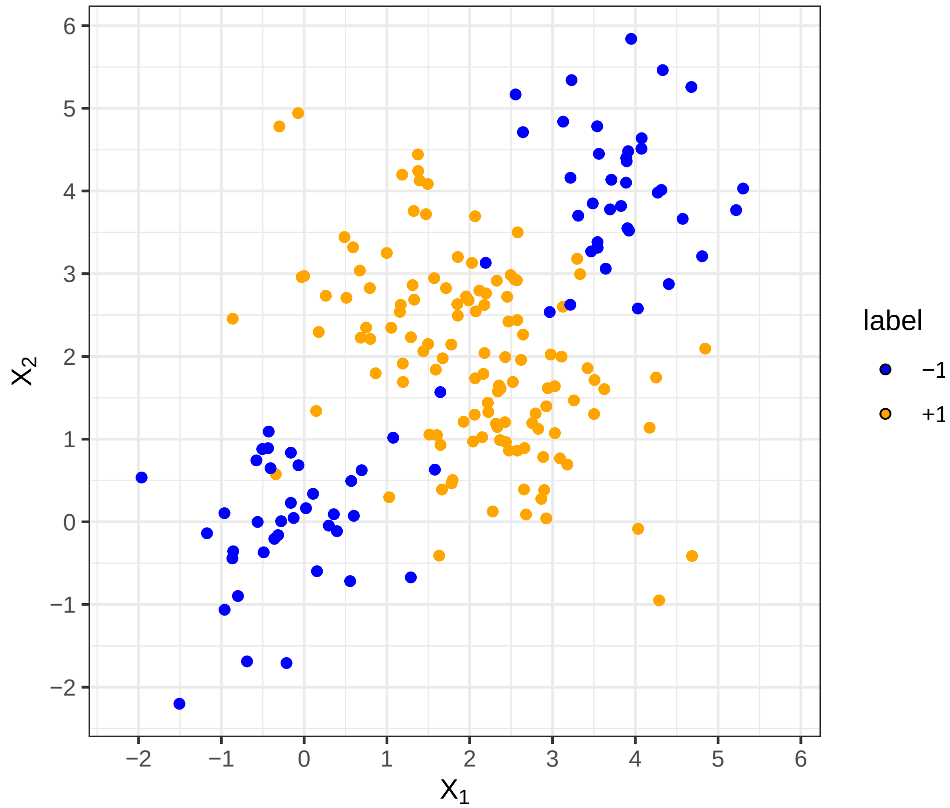
2.3 SVM in R

- The `svm()` function from the `e1071` package can implement SVM.
  - There is also a helpful `tune.svm()` to help you select the tuning parameters.

-The ISLR Lab in Section 9.6 has example R code.

2.4 Kernels

To help illustrate the difference between different kernels, let's look at slightly different data that won't be easy to classify using linear classifiers.

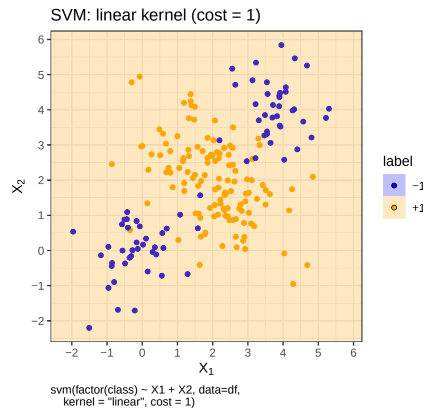


Three popular kernels (but there are many more) are the *linear*, *polynomial*, and *radial basis*

2.4.1 Linear Kernel

The *linear kernel* is

$$K(\mathbf{x}, \mathbf{u}) = \sum_{j=1}^{p} \mathbf{x}_j \mathbf{u}_j = \langle \mathbf{x}, \mathbf{u} \rangle$$



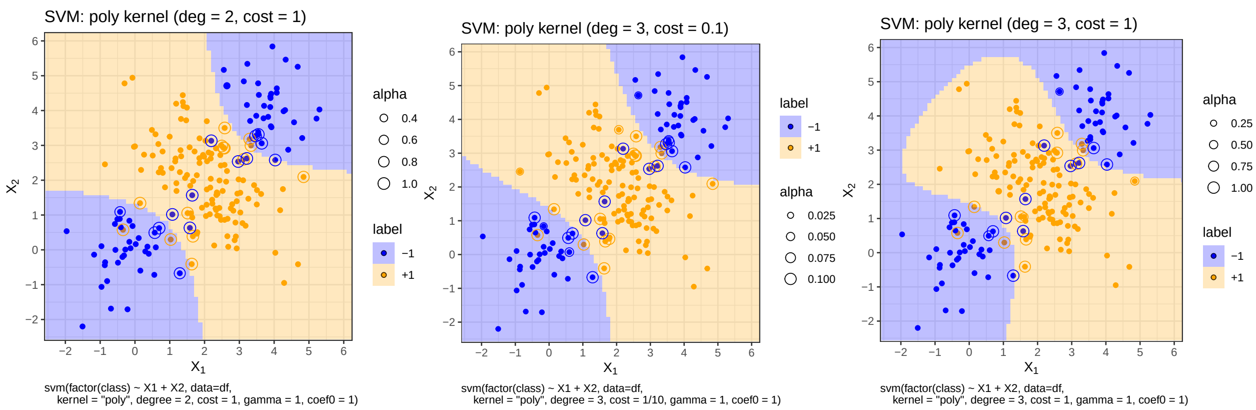
$$
\begin{aligned}
\hat{f}_{\text{linear}}(\mathbf{u}) &= \hat{\beta}_0 + \sum_{i=1}^{n} \hat{\alpha}_i \, y_i \, K(\mathbf{x}_i, \mathbf{u}) \\
&= \hat{\beta}_0 + \sum_{i=1}^{n} \hat{\alpha}_i \, y_i \left(\sum_{j=1}^{p} \mathbf{x}_{ij} \mathbf{u}_j \right)
\end{aligned}
$$

2.4.2 Polynomial Kernel

The *polynomial kernel of degree deg* is

$$
K(\mathbf{x}, \mathbf{u}) = \left(1 + \sum_{j=1}^{p} \mathbf{x}_j \mathbf{u}_j \right)^{\text{deg}}
$$

Note: in R, the ‘svm()‘ function from ‘e1071‘ package includes two other tuning parameters (`gamma` and `coef0`)



$$
\begin{aligned}
\hat{f}_{\text{poly}}(\mathbf{u}) &= \hat{\beta}_0 + \sum_{i=1}^{n} \hat{\alpha}_i \, y_i \, K(\mathbf{x}_i, \mathbf{u}) \\
&= \hat{\beta}_0 + \sum_{i=1}^{n} \hat{\alpha}_i \, y_i \left(1 + \sum_{j=1}^{p} \mathbf{x}_j \mathbf{u}_j \right)^{\text{deg}}
\end{aligned}
$$

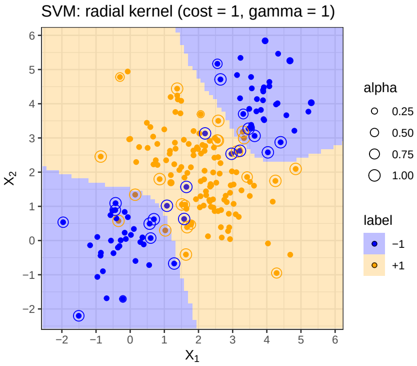
2.4.3 Radial Basis Kernel

The *Radial Basis Kernel* with parameter γ (`gamma`) is

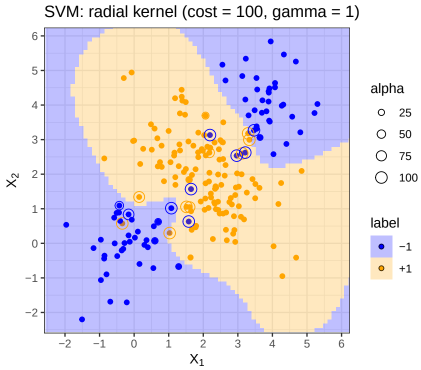
$$
\begin{aligned}
K(\mathbf{x}, \mathbf{u}) &= \exp\left(-\gamma \sum_{j=1}^{p} (\mathbf{x}_j - \mathbf{u}_j)^2 \right) \\
&= \exp\left(-\gamma \, dist^2(\mathbf{x}, \mathbf{u}) \right)
\end{aligned}
$$

where $dist^2(\mathbf{x}, \mathbf{u})$ is the *squared Euclidean distance* between $\mathbf{x}$ and $\mathbf{u}$.

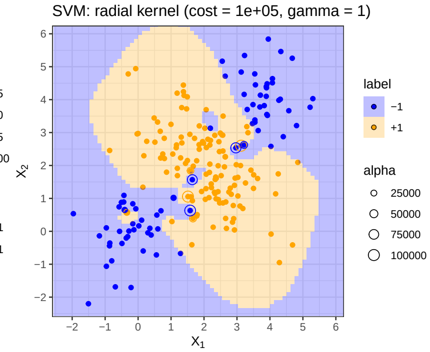
- The Radial Basis kernel is large for test observations close to training observations.
- Notice that both `cost` and `gamma` are influential tuning parameters.



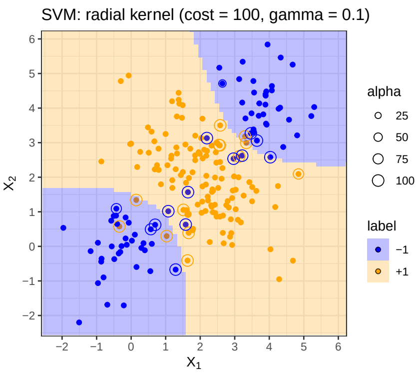
svm(factor(class) ~ X1 + X2, data=df,
kernel = "radial", cost = 1, gamma = 1)



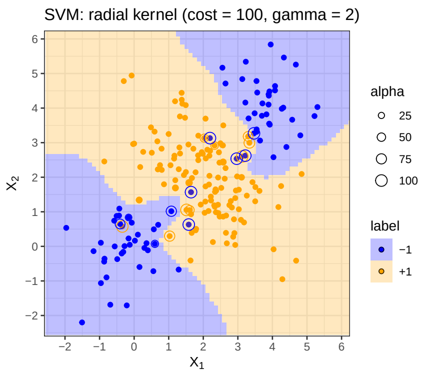
svm(factor(class) ~ X1 + X2, data=df,
kernel = "radial", cost = 100, gamma = 1)



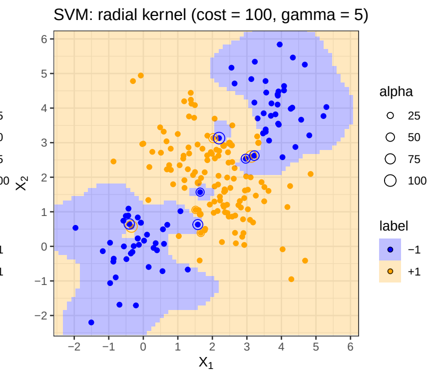
svm(factor(class) ~ X1 + X2, data=df,
kernel = "radial", cost = 100000, gamma = 1)



svm(factor(class) ~ X1 + X2, data=df,
kernel = "radial", cost = 100, gamma = 1/10)



svm(factor(class) ~ X1 + X2, data=df,
kernel = "radial", cost = 100000, gamma = 1)



svm(factor(class) ~ X1 + X2, data=df,
kernel = "radial", cost = 1, gamma = 1)

$$
\begin{aligned}
\hat{f}_{\text{radial}}(\mathbf{u}) &= \hat{\beta}_0 + \sum_{i=1}^{n} \hat{\alpha}_i \, y_i \, K(\mathbf{x}_i, \mathbf{u}) \\
&= \hat{\beta}_0 + \sum_{i=1}^{n} \hat{\alpha}_i \, y_i \, \exp\left(-\gamma \, dist^2(\mathbf{x}, \mathbf{u})\right)
\end{aligned}
$$

2.5 Unbalanced Data

Two primary approaches:

1. Use weights corresponding to prior class probabilities
   - See the `class.weights` argument in `?svm`
2. Use a threshold on the score
   - E.g., choose label = +1 if: $\hat{f}(\mathbf{x}) > t$
   - The $\hat{f}$ are the `decision.values` output of `predict.svm`
   - Its an attribute of the returned object