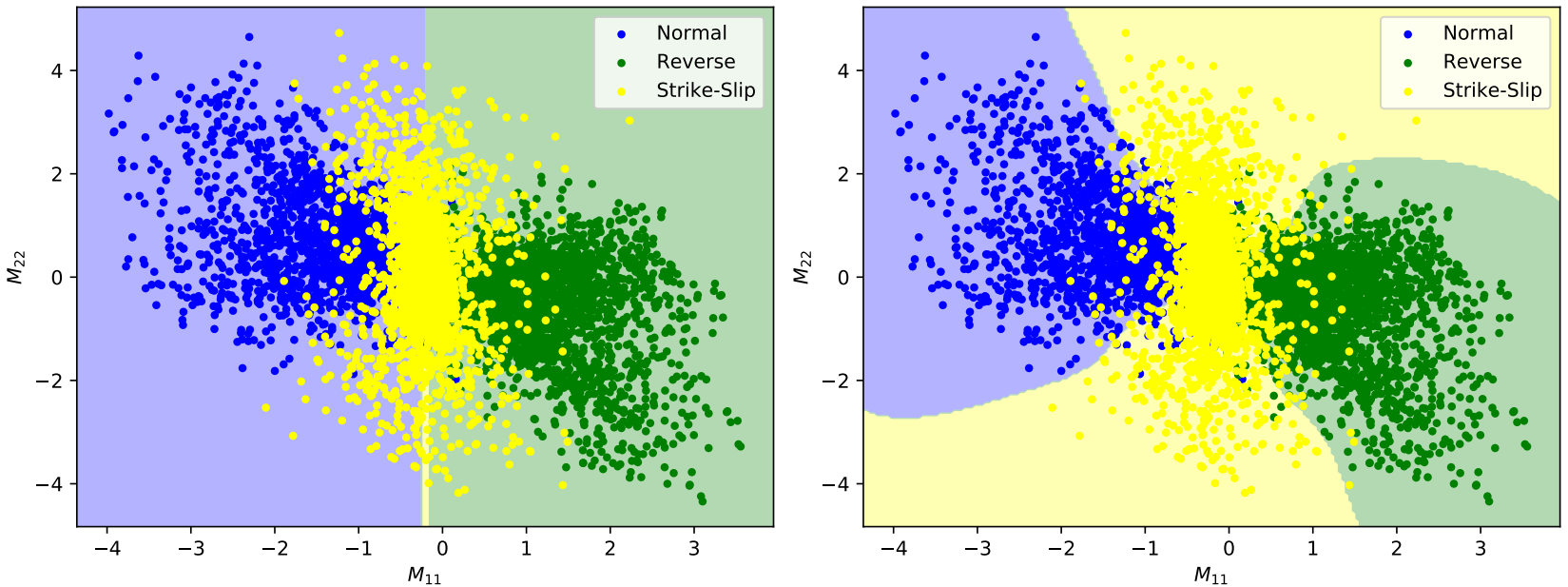
Machine Learning in Geophysics

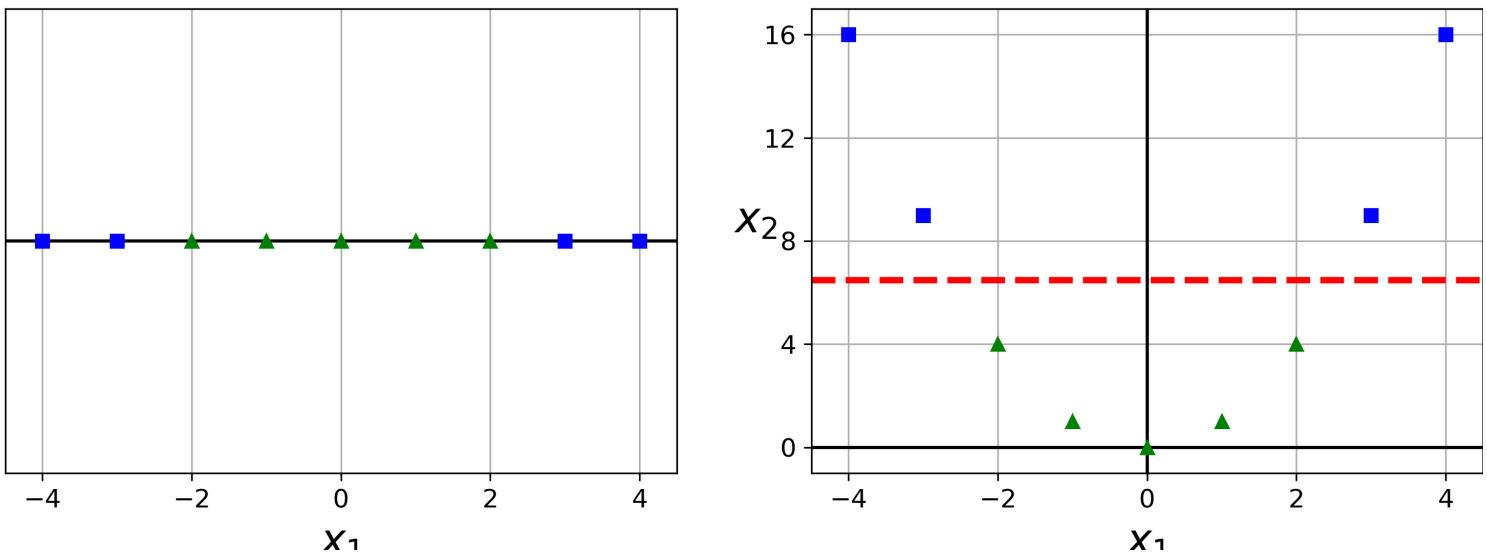
Lecture 4 – Kernel SVM, Overfitting and Underfitting

Non-linear SVM



We can extend Support Vector Machines beyond the simple linear model.

Basic idea



We add additional features that are functions of the data, e.g. here

$$\mathbf{x}_2 = \phi(\mathbf{x}_1) = \mathbf{x}_1^2.$$

Then categories can become linearly separable.

Theory

Remember the minimization criterion for linear soft-margin SVM

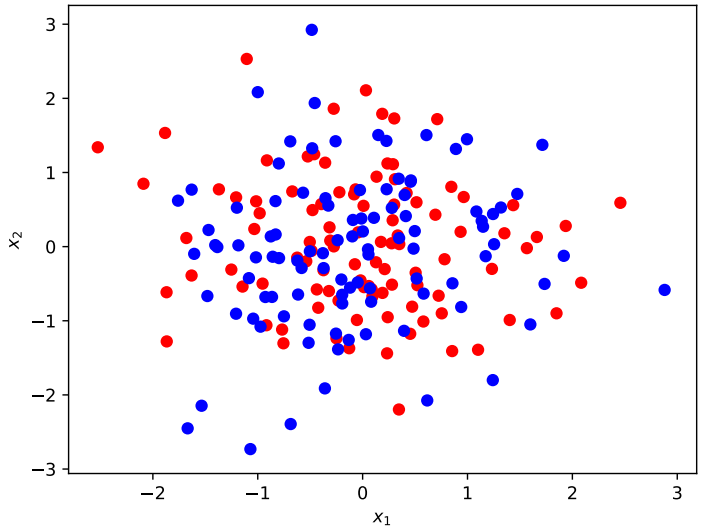
$$\|\mathbf{w}\| + C\sum_{i=1}^{M}\zeta_i \text{ subject to } y_i\left(\mathbf{w}\cdot\mathbf{x}_i + b\right) \geq 1 - \zeta_i.$$

With $\phi : \mathbb{R}^M \to \mathbb{R}^D$ can modify this

$$\|\mathbf{w}\| + C\sum_{i=1}^{D}\zeta_i \text{ subject to } y_i\left(\mathbf{w}\cdot\phi(\mathbf{x}_i) + b\right) \geq 1 - \zeta_i$$

to incorporate those extra features.

Some remarks



- Adding enough features we can make any dataset linearly separable
- Computational complexity increases with additional features
- Some useful ϕ map to infinite dimensional spaces

Dual problem

Representer theorem

The solution for $\mathbf{w}$ can always be expressed as a linear combination of the inputs

$$\mathbf{w} = \sum_{j=1}^{N} \alpha_j \phi(\mathbf{x}_j).$$

Can rewrite our constraints

$$y_i \left(\mathbf{w} \cdot \phi(\mathbf{x}_i) + b\right) \geq 1 - \zeta_i$$

as

$$y_i \left(\sum_{j=1}^{N} \alpha_j \phi(\mathbf{x}_j) \cdot \phi(\mathbf{x_i}) + b \right) \geq 1 - \zeta_i$$

Dual problem

And similarly

$$\|\mathbf{w}\| = \mathbf{w} \cdot \mathbf{w} = \sum_{j,k=1}^{N} \alpha_j \alpha_k \phi(\mathbf{x}_j) \cdot \phi(\mathbf{x}_k)$$

So in both cases we only need to compute inner product terms

$$k(\mathbf{x}_i, \mathbf{x}_k) = \phi(\mathbf{x}_j) \cdot \phi(\mathbf{x}_k),$$

expressed by the kernel $k(\mathbf{x}_i, \mathbf{x}_k)$.
Can also write prediction equation in similar form.

Mercer theorem

We can construct kernels $k(\mathbf{a}, \mathbf{b})$ without having to know the associated transformation function ϕ. $k(\mathbf{a}, \mathbf{b})$ has to fulfill certain conditions (Mercer conditions).
Some useful kernels

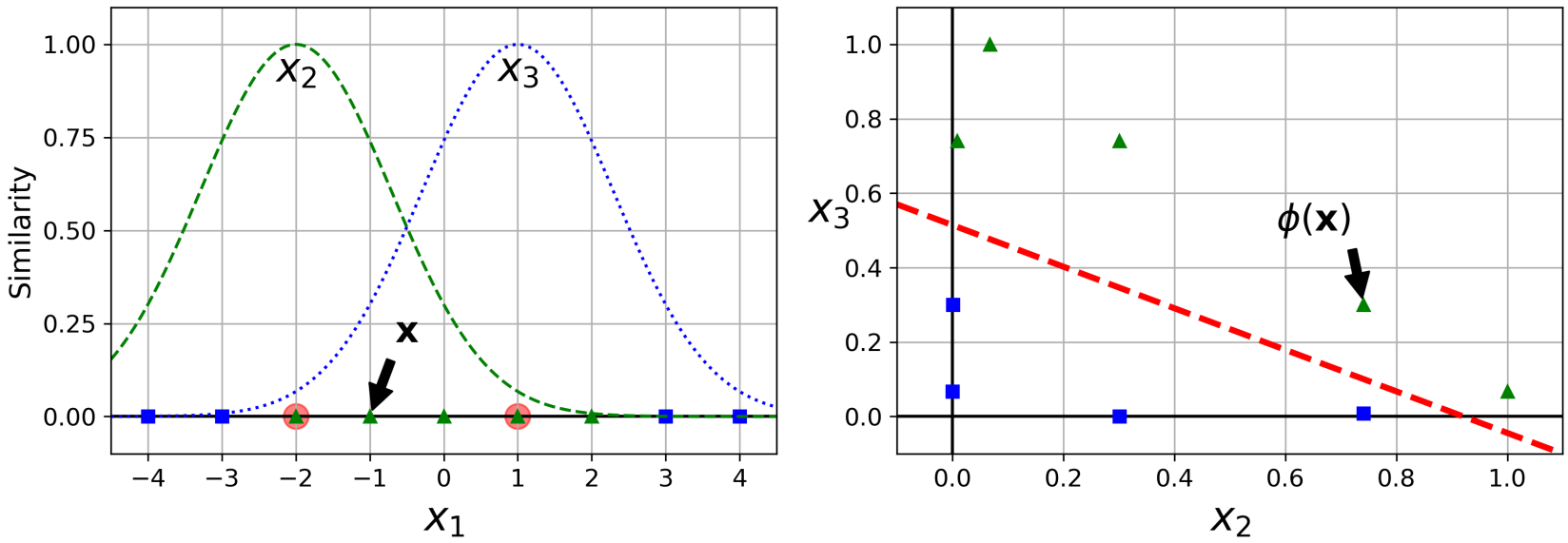
Linear: $k(\mathbf{a}, \mathbf{b}) = \mathbf{a} \cdot \mathbf{b}$

Polynomial: $k(\mathbf{a}, \mathbf{b}) = (\gamma \mathbf{a} \cdot \mathbf{b} + r)^d$

Gaussian RBF: $k(\mathbf{a}, \mathbf{b}) = \exp(-\gamma \|\mathbf{a} - \mathbf{b}\|)$

Sigmoid: $k(\mathbf{a}, \mathbf{b}) = \tanh(\gamma \mathbf{a} \cdot \mathbf{b} + r)$, not strictly a kernel but works in practice

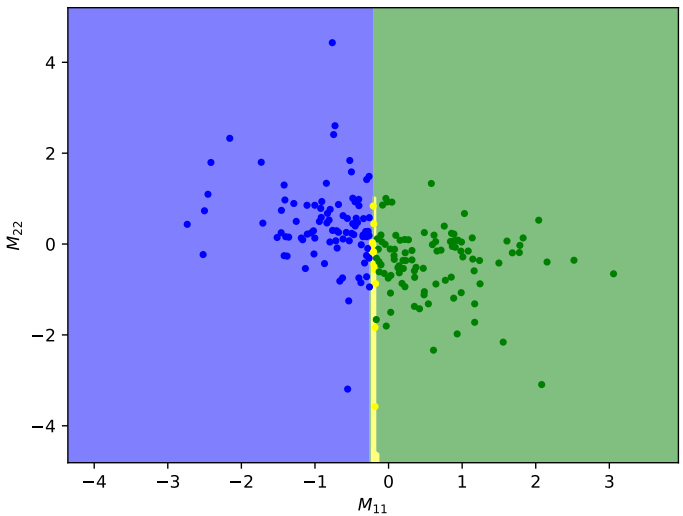
Gaussian RBF



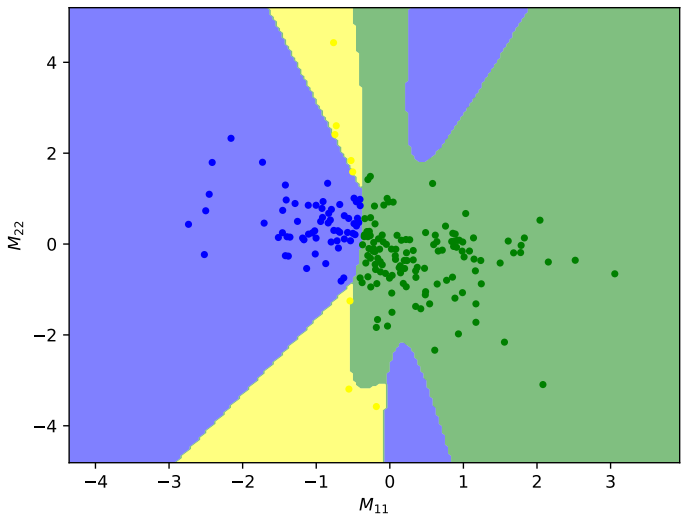
- Kernel is based on distance between points.
- Works even without identifying special references
- In practice default is to use all points as references
- Drawback large increase in features.

Kernel comparison

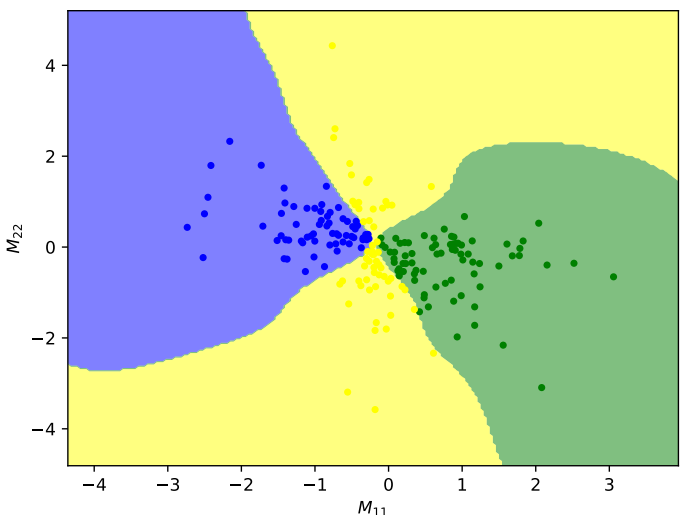
Linear



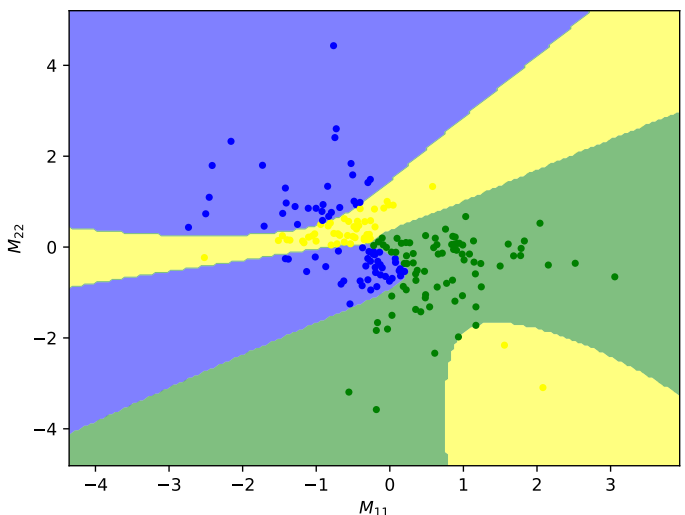
Polynomial



Gaussian RBF



Sigmoid



Confusion matrices

Linear

$$\begin{pmatrix} 57.00 & 0.00 & 1.00 \\ 1.00 & 68.00 & 0.00 \\ 31.00 & 27.00 & 11.00 \end{pmatrix}$$

Polynomial

$$\begin{pmatrix} 50.00 & 8.00 & 0.00 \\ 0.00 & 69.00 & 0.00 \\ 13.00 & 48.00 & 8.00 \end{pmatrix}$$

Gaussian RBF

$$\begin{pmatrix} 55.00 & 0.00 & 3.00 \\ 1.00 & 64.00 & 4.00 \\ 8.00 & 9.00 & 52.00 \end{pmatrix}$$

Sigmoid

$$\begin{pmatrix} 27.00 & 0.00 & 31.00 \\ 3.00 & 61.00 & 5.00 \\ 40.00 & 17.00 & 12.00 \end{pmatrix}$$

Question

Which kernel do you think performs best?

Accuracy

Can reduce confusion matrix **C** to a single number, the accuracy a

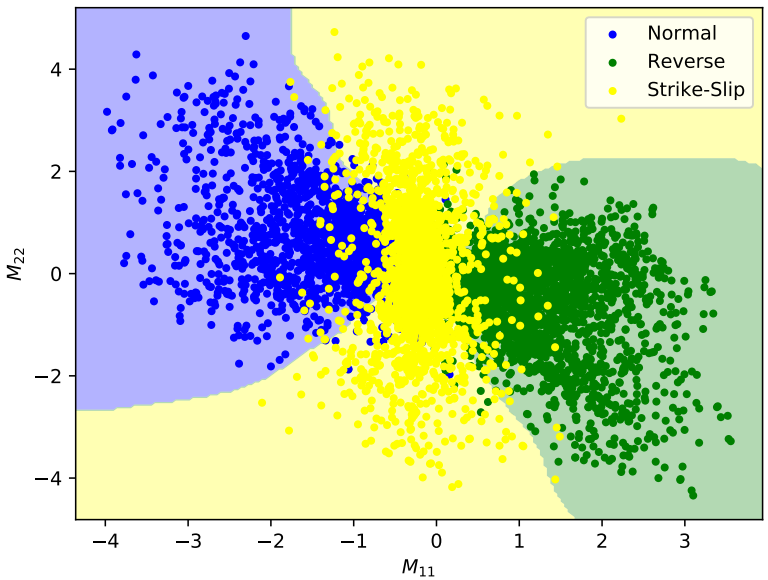
$$a = \frac{\sum_i C_{ii}}{\sum_{i,j} C_{ij}}$$

the sum of true predictions divided by total number of predictions.

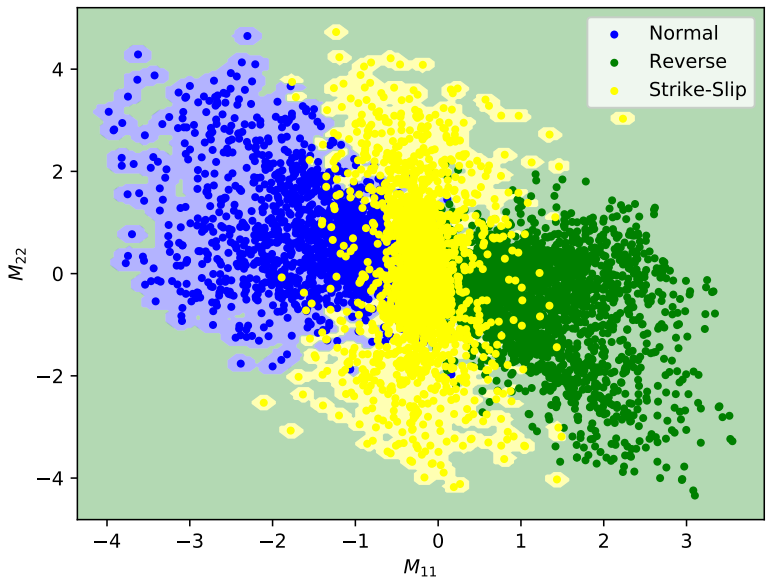
Linear	Polynomial
0.69	0.65
Gaussian RBF	Sigmoid
0.87	0.51

Training data

$\gamma = 1, c = 1$



$\gamma = 100, c = 100$



Question

Which one is better?

Training data

Only considering the training data

$\gamma = 1, c = 1$

$$\begin{pmatrix} 2325.00 & 0.00 & 268.00 \\ 7.00 & 4538.00 & 312.00 \\ 262.00 & 166.00 & 2121.00 \end{pmatrix}$$

a = 0.89

$\gamma = 100, c = 100$

$$\begin{pmatrix} 2369.00 & 0.00 & 189.00 \\ 9.00 & 4557.00 & 214.00 \\ 216.00 & 147.00 & 2298.00 \end{pmatrix}$$

a= 0.92

Training data

Only considering the training data

$\gamma = 1, c = 1$

$$\begin{pmatrix} 2325.00 & 0.00 & 268.00 \\ 7.00 & 4538.00 & 312.00 \\ 262.00 & 166.00 & 2121.00 \end{pmatrix}$$

a = 0.89

$\gamma = 100, c = 100$

$$\begin{pmatrix} 2369.00 & 0.00 & 189.00 \\ 9.00 & 4557.00 & 214.00 \\ 216.00 & 147.00 & 2298.00 \end{pmatrix}$$

a= 0.92

Looking at the validation data

$\gamma = 1, c = 1$

$$\begin{pmatrix} 55.00 & 0.00 & 3.00 \\ 1.00 & 65.00 & 3.00 \\ 8.00 & 10.00 & 51.00 \end{pmatrix}$$

a = 0.87

$\gamma = 100, c = 100$

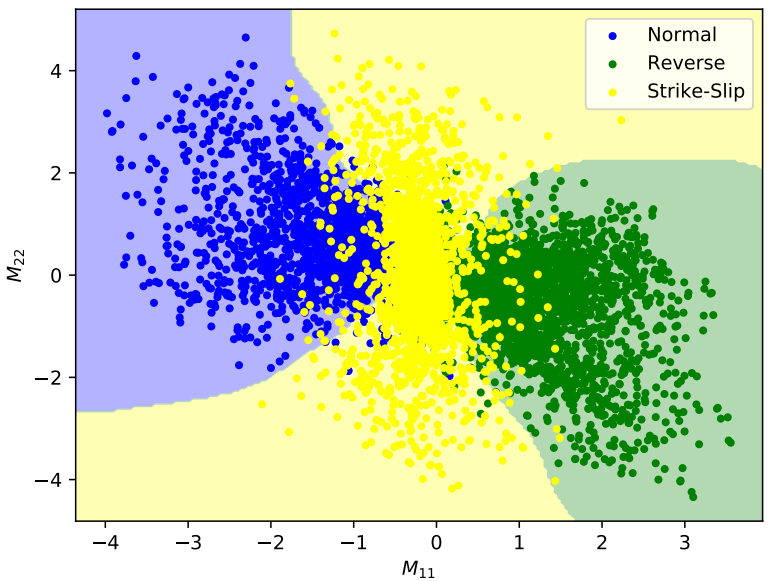
$$\begin{pmatrix} 52.00 & 0.00 & 6.00 \\ 1.00 & 62.00 & 6.00 \\ 11.00 & 8.00 & 50.00 \end{pmatrix}$$

a= 0.83

At $\gamma = 100, c = 100$ we are overfitting the data.

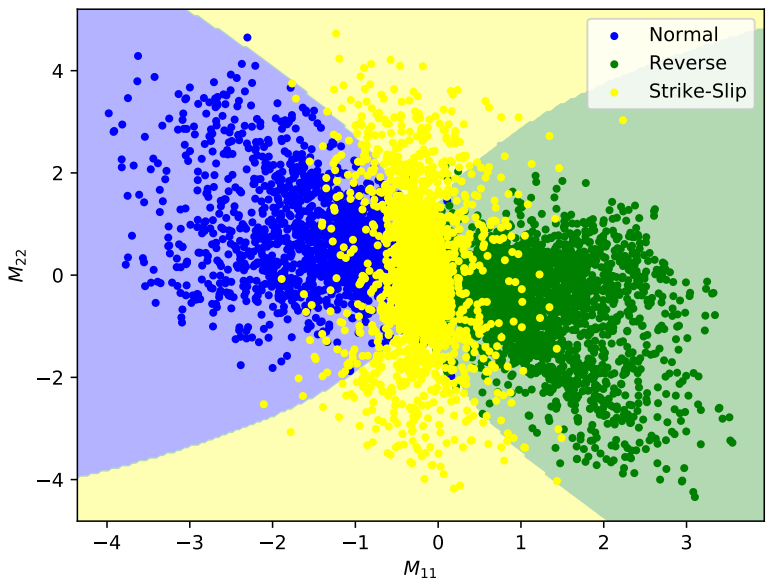
Training data

$\gamma = 1, c = 1$



a = 0.87

$\gamma = 0.01, c = 1$



a =0.71

At $\gamma = 0.01, c = 1$ we start to underfit the data.

Fitting

Overfitting

- Happens when we try to maximize fit to training data at all costs
- High accuracy (precision, recall) suggest very good classification
- Model completely adjusted to idiosyncrasies/noise of training data
- Performance drops sharply for validation data

Underfitting

- Model cannot capture complexity of data
- Either model to simple or too strongly regularized
- Performance low on test and training data

Strategies

For polynomial kernel

$$c = \begin{pmatrix} 50.00 & 8.00 & 0.00 \\ 0.00 & 69.00 & 0.00 \\ 13.00 & 48.00 & 8.00 \end{pmatrix}$$

Select important quality criterion to maximize

- **Precision:** Reduce false positives (pregnancy test)
- **Recall:** Reduce false negatives (nuclear missile detection)
- **Accuracy:** Average number of correct classifications

Systematically test hyper-parameters and necessary features.

Summary

- Kernel trick expands SVM (and other techniques) to complex classification problems
- Different types of kernel exist, polynomial, gaussian RBF, sigmoid
- Need to carefully evaluate fit to avoid overfitting or underfitting
- Need to explore hyper-parameters to find optimal balance