Diagnostics & Kernel Methods Visualized

Ondřej Bojar

April 3, 2019




Charles University
Faculty of Mathematics and Physics
Institute of Formal and Applied Linguistics

Diagnostics

Outline

- Motivation for principled analysis.
- Ideas for visualization.
- Bias vs. Variance.
- Optimizer vs. Objective function issue.
  - a.k.a. Search error vs. Modelling error.
- Error analysis, Ablative analysis.

Slides based on:

- the Stanford ML Lecture 11: `http://www.youtube.com/watch?v=sQ8T9b-uGVE`
- `http://scott.fortmann-roe.com/docs/BiasVariance.html`
- All errors are Ondřej's fault.

Motivation: Debugging ML

Some ML does not perform sufficiently well.
You can consider random improvements:

- Getting more training examples.
- Reduce the set of features.
- Enlarge the set of features.
- Use different features.
- Run the optimizer for some more iterations.
- Choose a different optimization algorithm.
- Use a different regularization term or constant value.
- Try another learning algorithm (SVM).

... some may be fixing problems you don't have.

Principled Analysis

First figure out what's going on.

- Overfitting vs. Underfitting?
- Search error vs. Modelling error?
- Complex system: Find the most problematic component.

Trivial but vital:

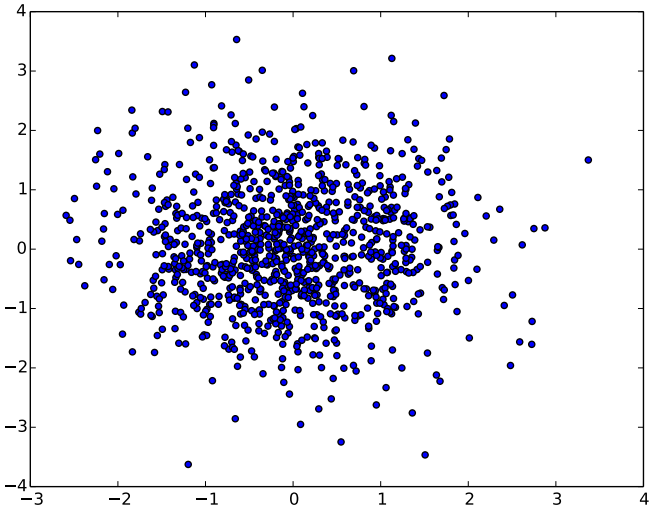
- Visualize the data. (Plot or view frequent patterns.)
- Start with simple things.

Data Visualization

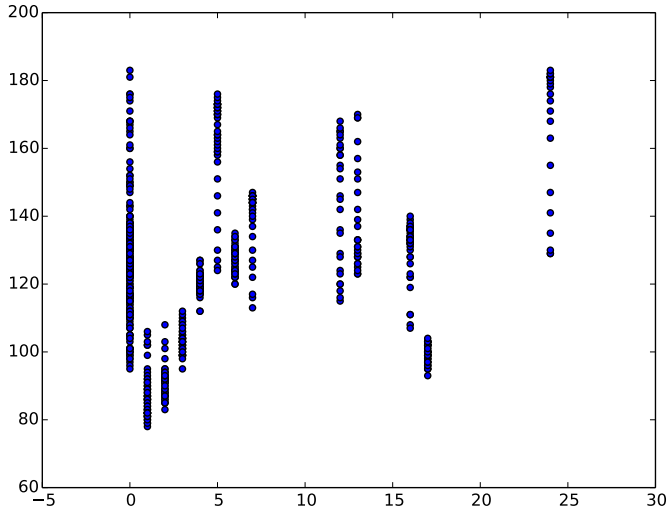
- Data visualization is extremely useful.
- Always plot the data when working with a new dataset.
- This is inherent part of `hw_my_dataset`.
  - Choose one way of visualizing the data to give a quick overview of it.
- Suggested gradual steps on the following slides.
  (Python source on the seminar web page.)

An excellent resource: `https://matplotlib.org/gallery.html`

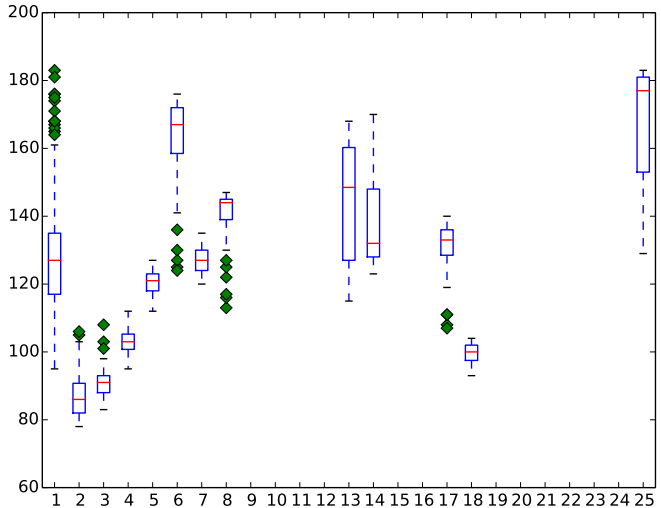
Scatter Plot of Random Values



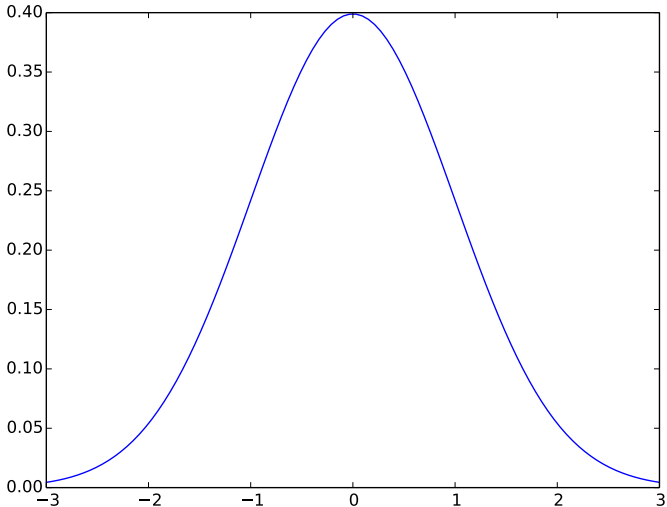
Scatter Plot of Activity – Heart Rate



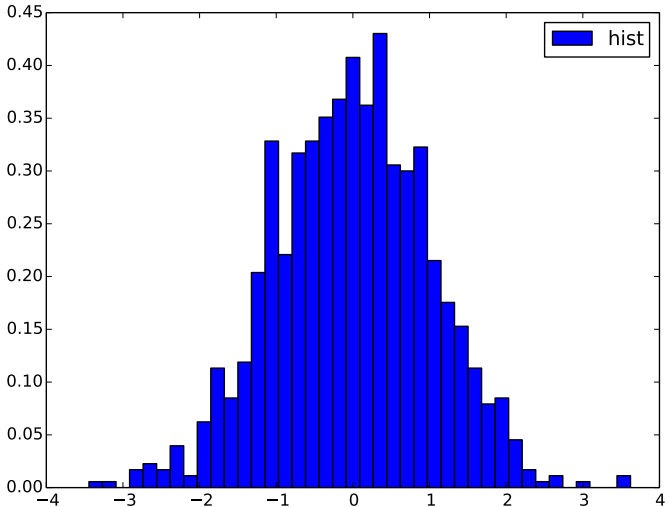
Box Plot of Activity – Heart Rate



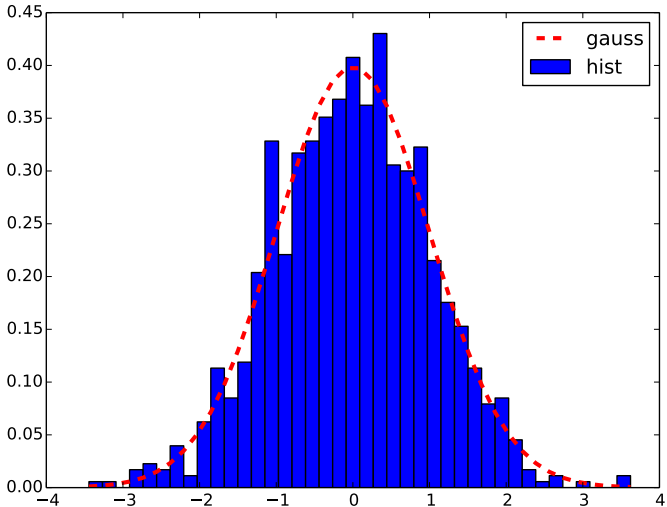
Gaussian Function



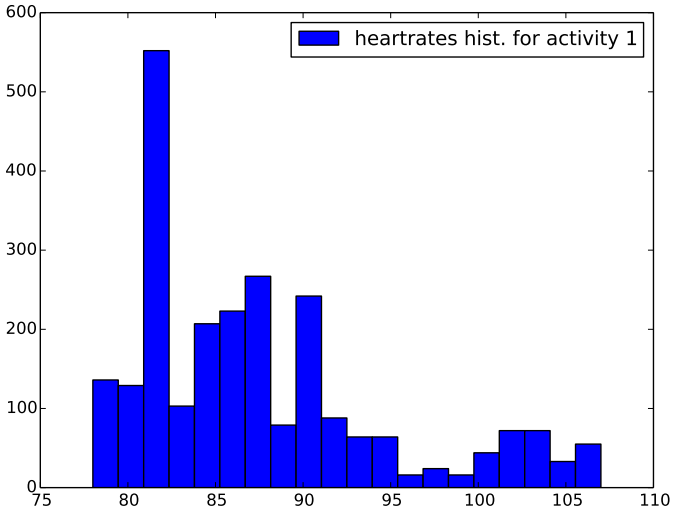
Histogram of Random Values



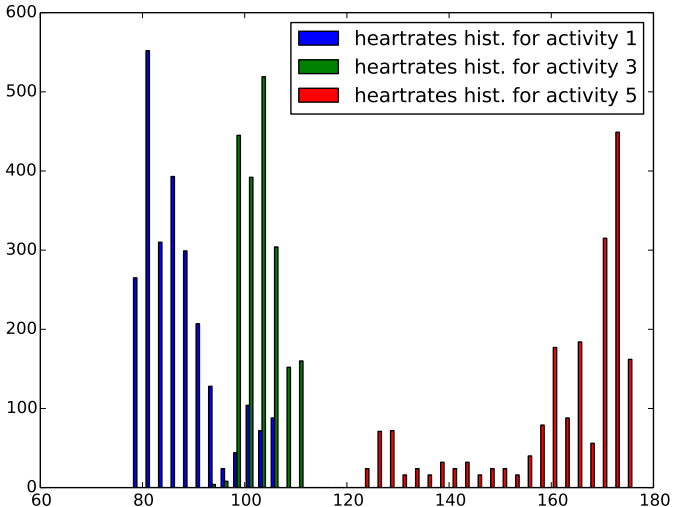
Histogram and Gaussian Function (not Fit!)



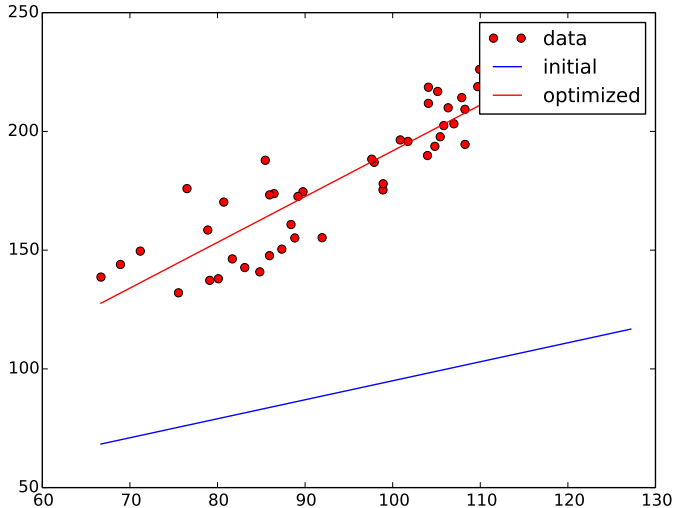
Histogram of Heart Rate when Lying



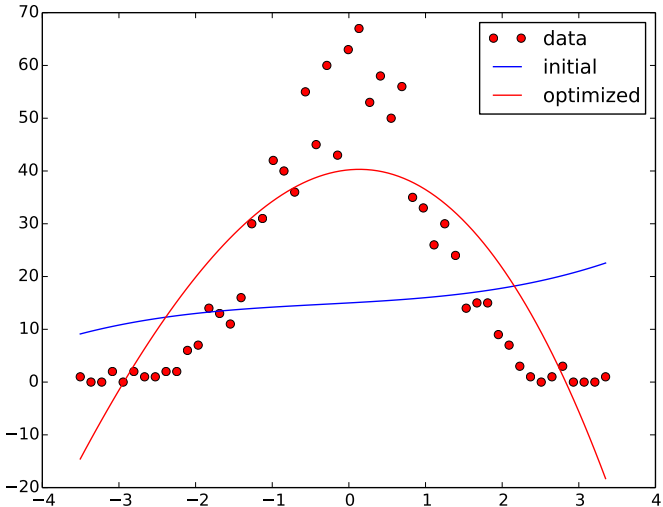
Hists of HR when Lying, Walking, Running



Least Squares for Linear Fit



Cubic Fit of a Histogram



Bias vs. Variance

High Variance = Overfitting:

- the model has too many parameters.

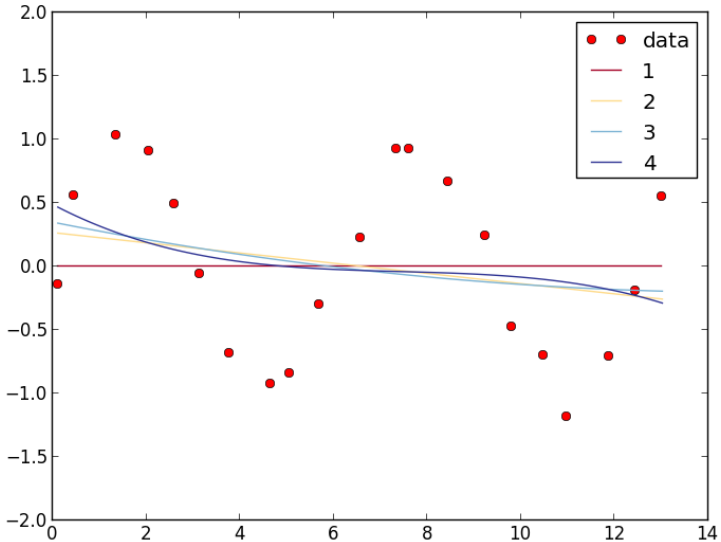
High Bias = Underfitting:

- the model is too rigid.

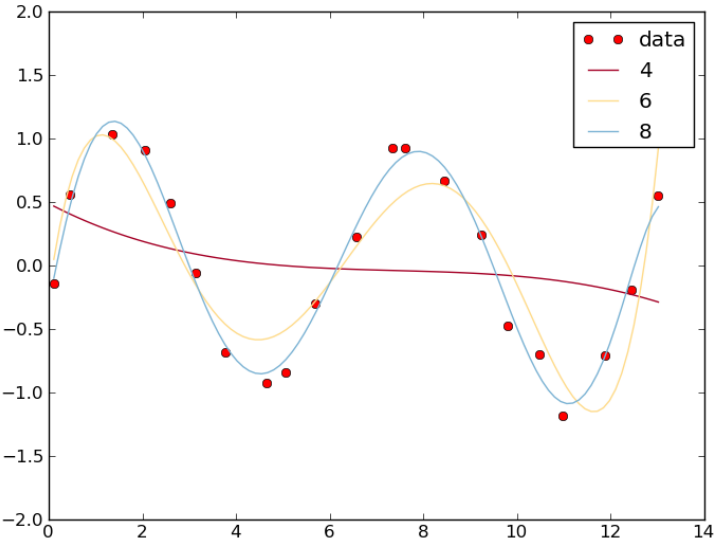
Consider:

- What is the effect of each of those on training error?
- Will more training data help?
- Sketch the shape of learning curves for each of those:
  - for the test error.
  - for the training error.

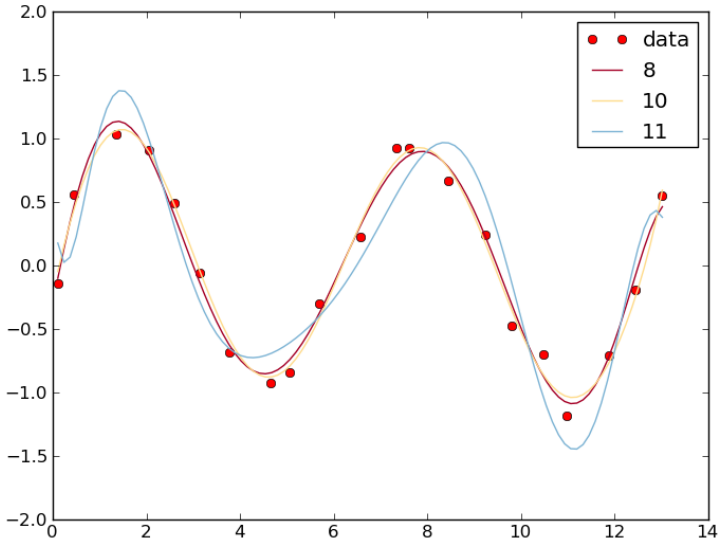
Fit sin(x) with poly, orders: 0,1,2,3



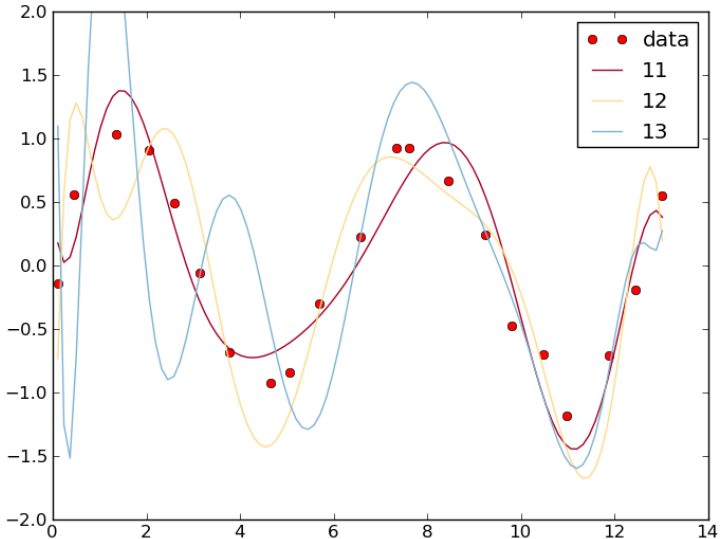
Fit sin(x) with poly, orders: 3,5,7



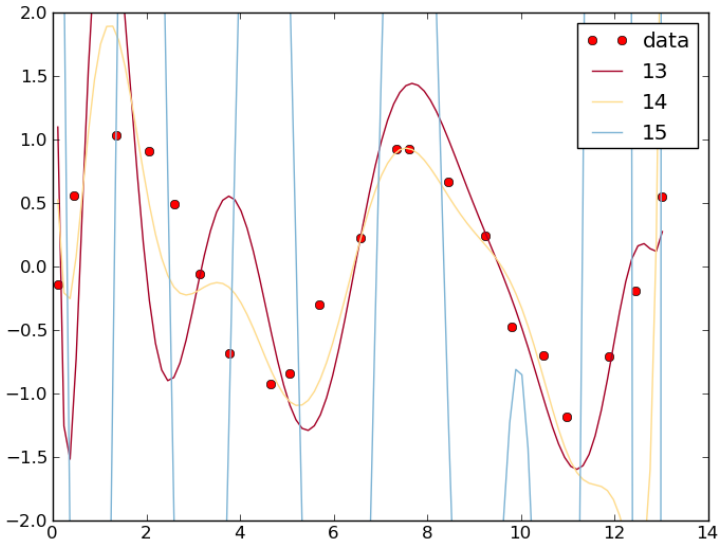
Fit sin(x) with poly, orders: 7,9,10



Fit sin(x) with poly, orders: 10,11,12



Fit sin(x) with poly, orders: 12,13,14



Bias-Variance Trade-off

$$Err(x) = E[(Y - \hat{f}_D(x))^2]$$

Expected error $Err(x)$ of learning $\hat{f}_D$ over various datasets D on a fixed test set x with observed values $Y = f(x) + \epsilon$ can be decomposed as:

$$\begin{array}{lccc} Err(x) = & (E\hat{f}_D(x) - f(x))^2 & +E(\hat{f}_D(x) - E\hat{f}_D(x))^2 & +\sigma_e^2 \\ Err(x) = & \text{Bias}^2 & +\text{Variance} & +\text{Noise} \end{array}$$

- Bias: how much the average predicted value $E\hat{f}_D(x)$ differs from the ideal value $f(x)$.
- Variance: how much a particular prediction $\hat{f}_D(x)$ differs from the average prediction $E\hat{f}_D(x)$, on average over datasets D.

The Bias-Variance Tradeoff

Given:

- the true function we want to approximate

$$f = f(\mathbf{x})$$

- the data set for training

$$D = \{(\mathbf{x}_1, t_1), (\mathbf{x}_2, t_2), \ldots, (\mathbf{x}_N, t_N)\} \text{ where } t = f + \varepsilon \text{ and } E\{\varepsilon\} = 0$$

- given D, we train an arbitrary neural network to approximate the function f by

$$y = g(\mathbf{x}, \mathbf{w})$$

The mean-squared error of this networks is:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(t_i - y_i)^2$$

To assess the effectiveness of the network, we want to know the expectation of

Diagnosing Bias vs. Variance from Learning Curves

See the slides by Andrew Ng, plots on slide 7 and 8.

Search vs. Modelling Error

Search Error:

- the optimizer fails to find the best parameters
- ... a problem with the optimizer.

Modelling Error:

- the best parameters do not lead to the best performance.
- ... a problem with the objective function.

Consider:

- Will more iterations help?
- When can two learners help to diagnose the problem?

Diagnosing for Search vs. Modelling Error

See the slides by Andrew Ng, slide 14.

Complex Systems

Error Analysis:

- Compares the best possible vs. current accuracy.
- Provide more and more golden truth data as part of the input.
- Find the component where the jump in accuracy is the highest.

Ablative Analysis:

- Compares some baseline vs. current accuracy.
- Switch off more and more components.
- Find the component where the loss in accuracy is the highest.

Kernels Illustrations

Outline for Kernel Illustrations

- Regularization parameter C in SVM.
- Linear Kernel: $k(\mathbf{x}, \mathbf{y}) = \mathbf{x} \cdot \mathbf{y}$
- The gain from higher dimensionality.
- Polynomial Kernel: $k(\mathbf{x}, \mathbf{y}) = (\gamma * \mathbf{x} \cdot \mathbf{y} + \text{coeff0})^{\text{degree}}$
- RBF Kernel: $k(\mathbf{x}, \mathbf{y}) = exp(-\gamma \|\mathbf{x} - \mathbf{y}\|^2); \gamma > 0$
  ... including their parameters
- Cross-validation Heatmap
- Multi-class SVM
  - For the PAMAP-easy dataset.
  - Regularization parameters.
  - Inseparable classes.

Based on `http://scikit-learn.org/stable/modules/svm.html` and other scikit-demos.

Regularization (C) in linear SVM

$$k(\mathbf{x}, \mathbf{y}) = \mathbf{x} \cdot \mathbf{y}$$

(Linear kernel = no kernel)

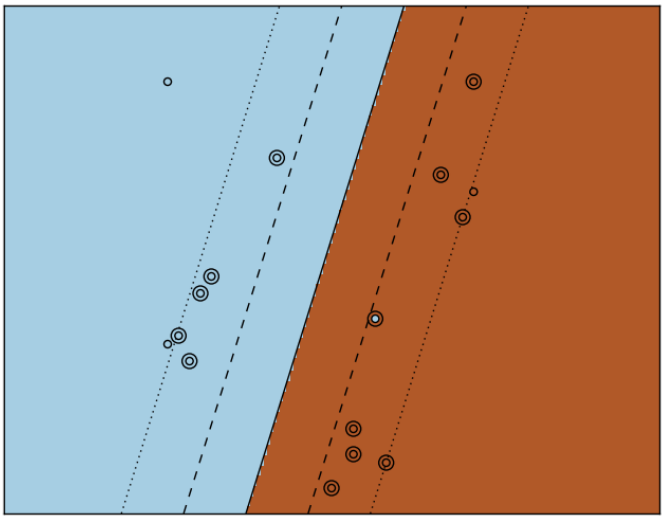
The parameter C in (linear) SVM:

- sets the weight of the sum of slack variables.
- serves as a regularization parameter.
- controls the number of support vectors.

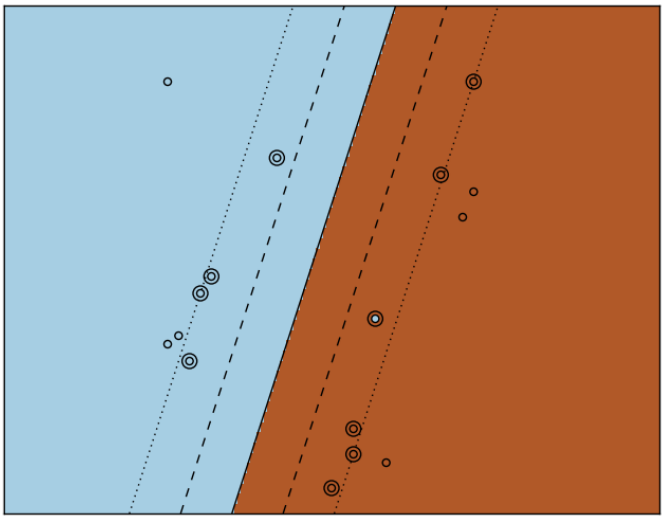
C	Penalty for Errors	Number of points considered	Margin	Bias	Variance
Low	Low	Many	Wide	High	Low
High	High	Few	Narrow	Low	High

Think C for Varian$C$e.

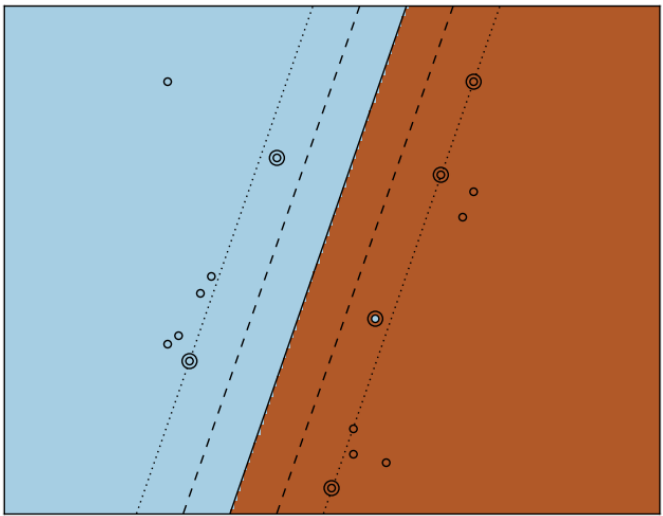
SVM Linear C=0.1



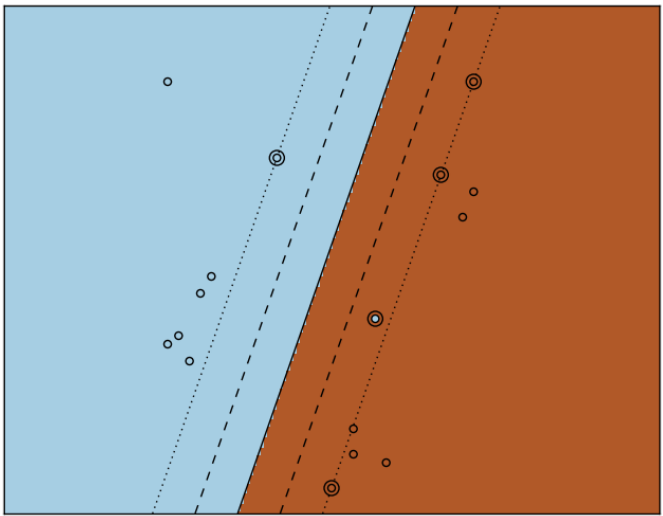
SVM Linear C=0.2



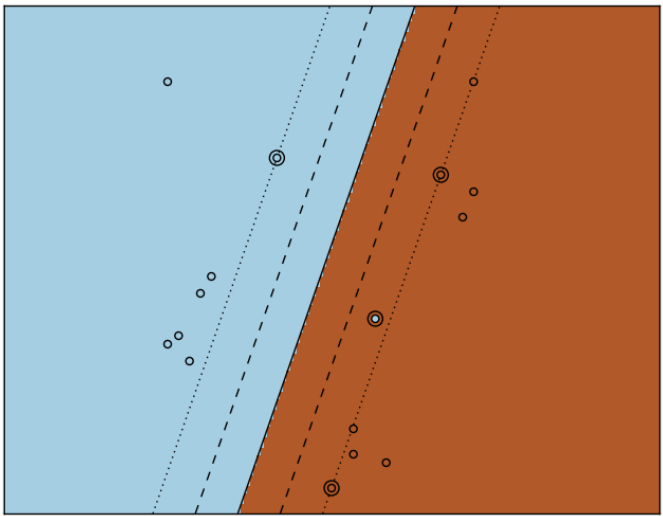
SVM Linear C=0.5



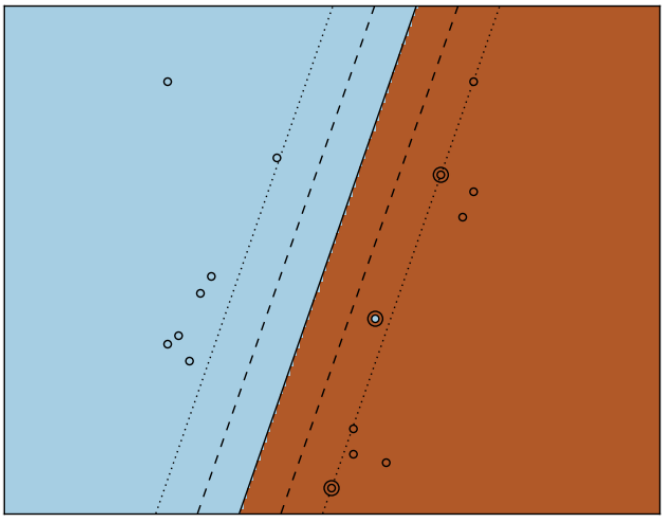
SVM Linear C=1



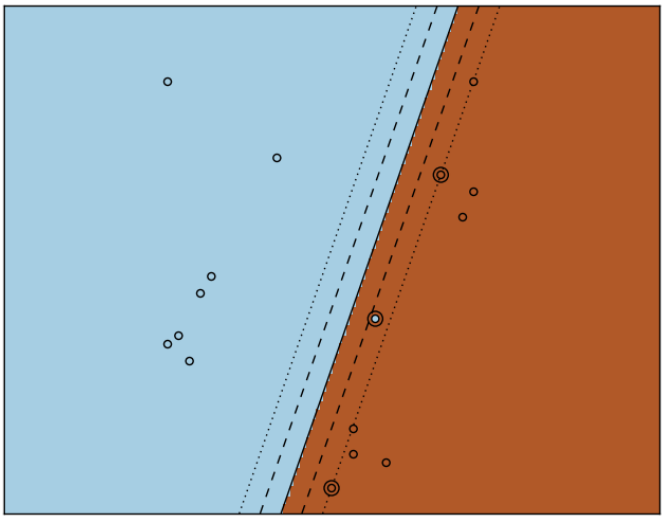
SVM Linear C=5



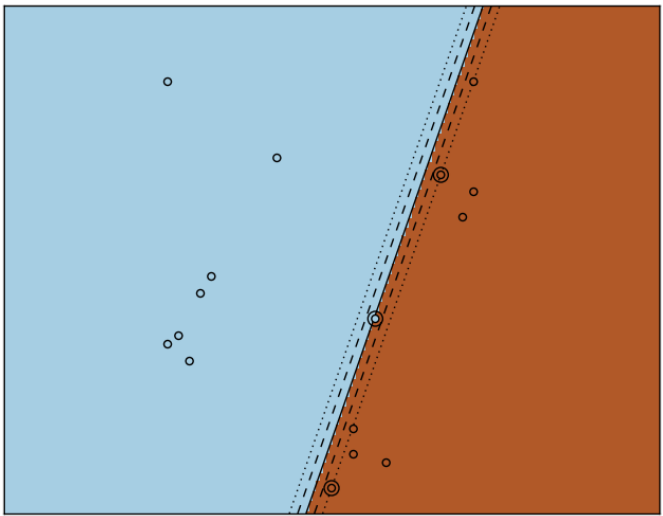
SVM Linear C=10



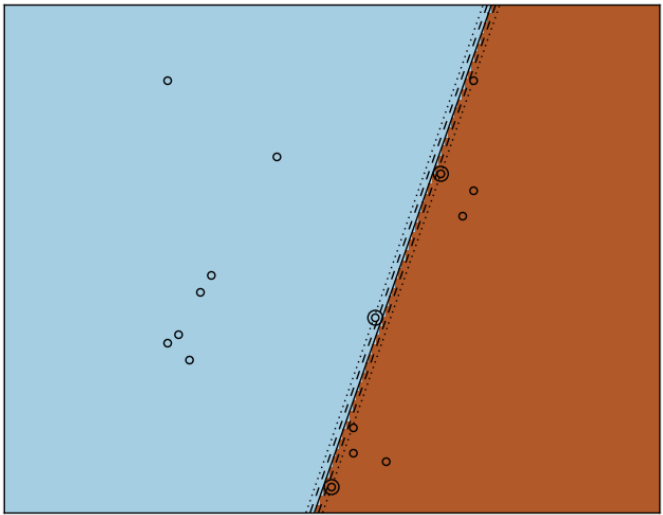
SVM Linear C=20



SVM Linear C=50



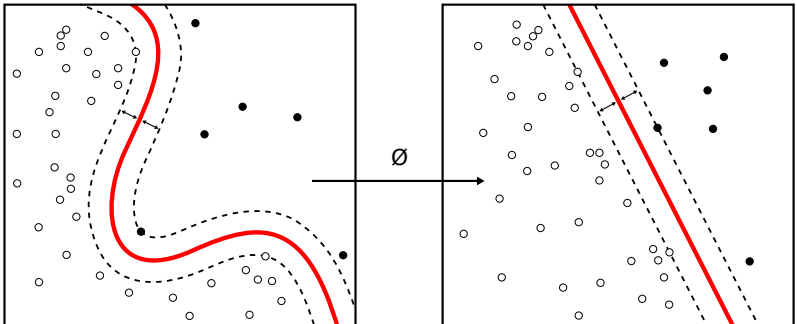
SVM Linear C=100



Benefitting from Higher Dimensionality

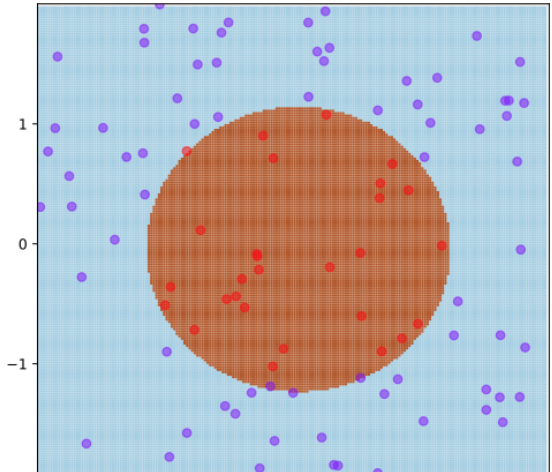
- Classifiers generally do linear separation.
- It can be very difficult to come up with features that allow for linear separation.

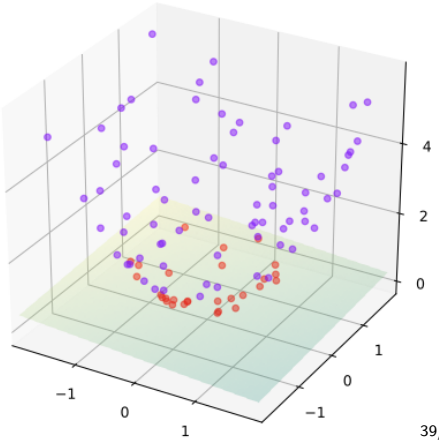
The trick: map the coordinates to another space where separation is possible:



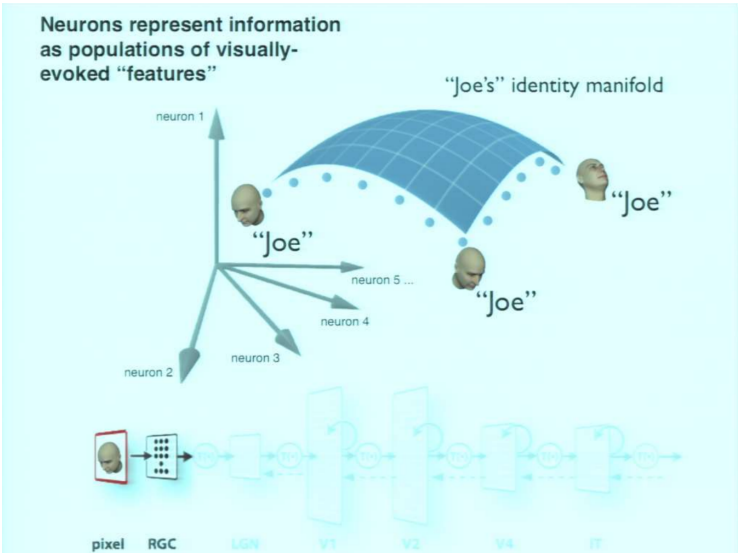
Kernel Function to a Higher Dimension

$$k(x, y) = xy + x^2y^2 \tag{1}$$

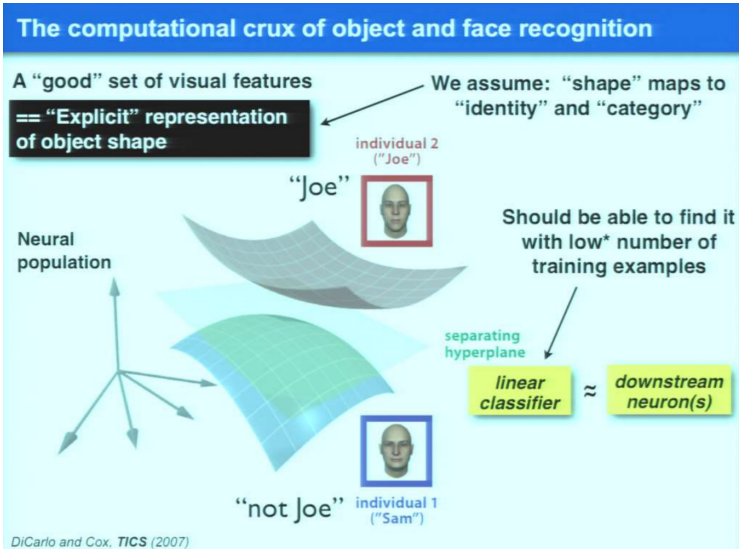




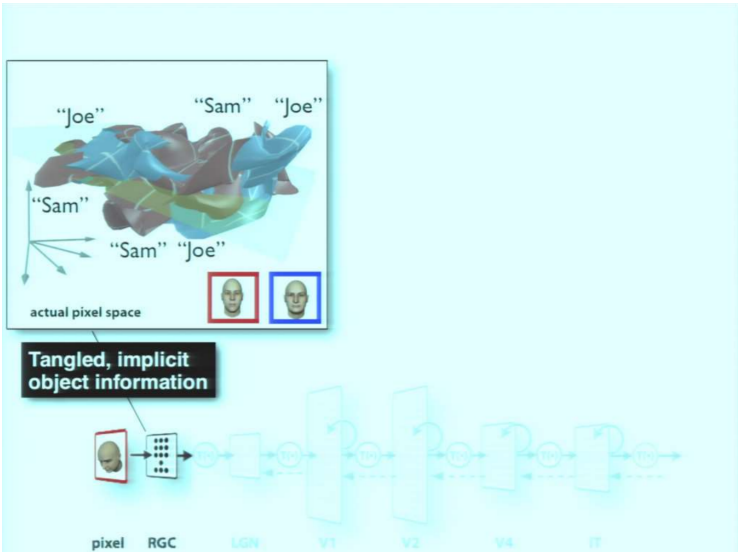
Deep NNs: Kernels on Steroids



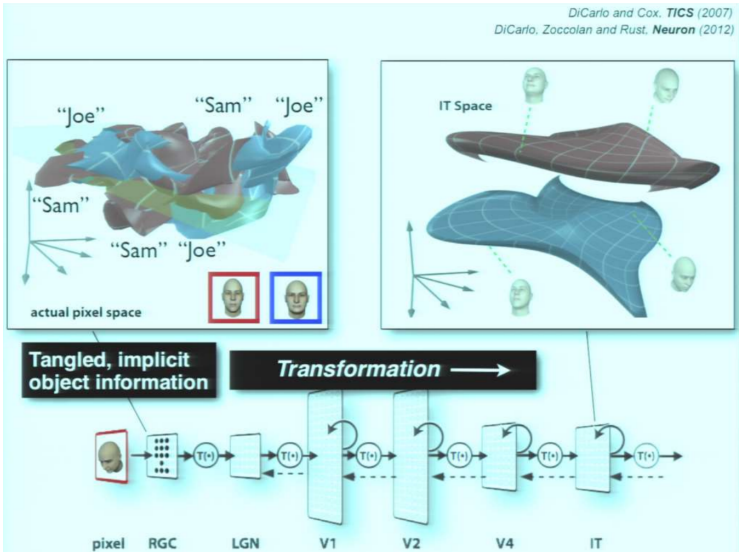
Deep NNs: Kernels on Steroids



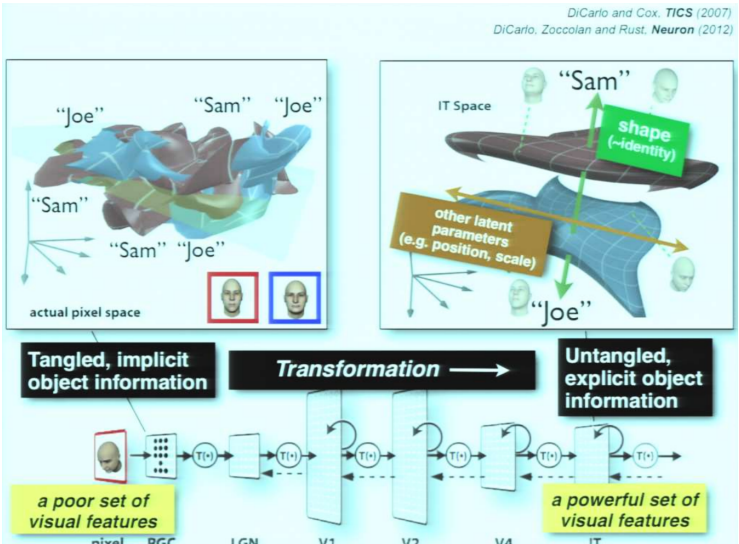
Deep NNs: Kernels on Steroids



Deep NNs: Kernels on Steroids



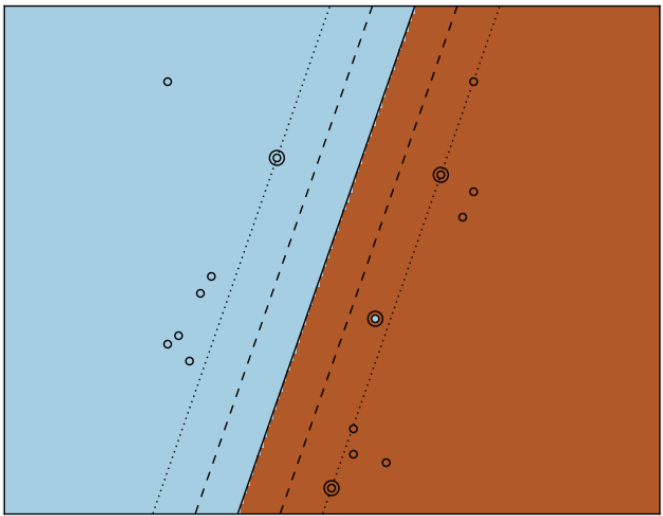
Deep NNs: Kernels on Steroids



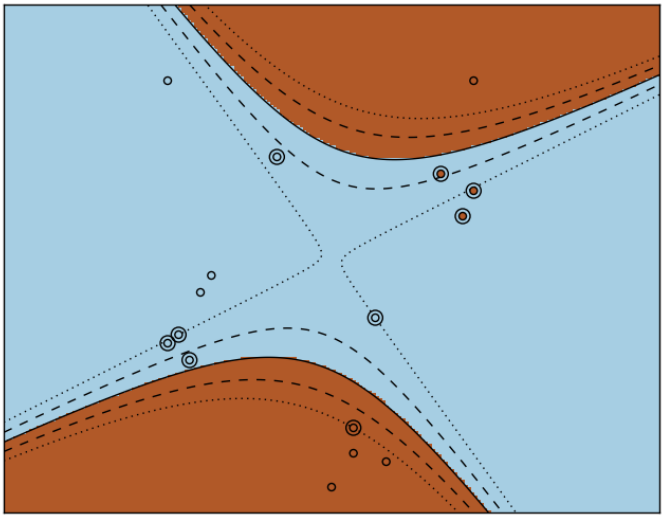
Polynomial Kernel

$$k(\mathbf{x}, \mathbf{y}) = (\gamma * \mathbf{x} \cdot \mathbf{y} + \text{coeff0})^{\text{degree}}$$

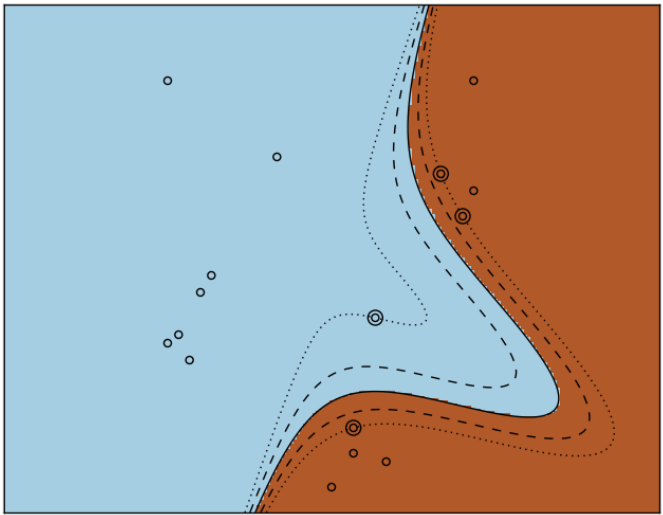
SVM Poly (degree 1)



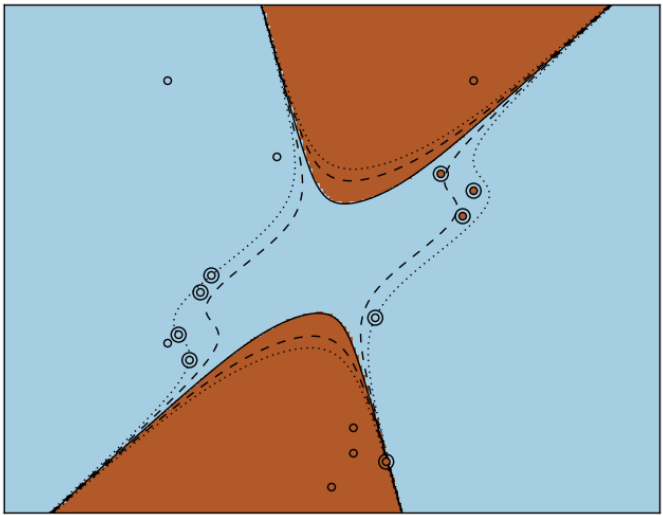
SVM Poly (degree 2)



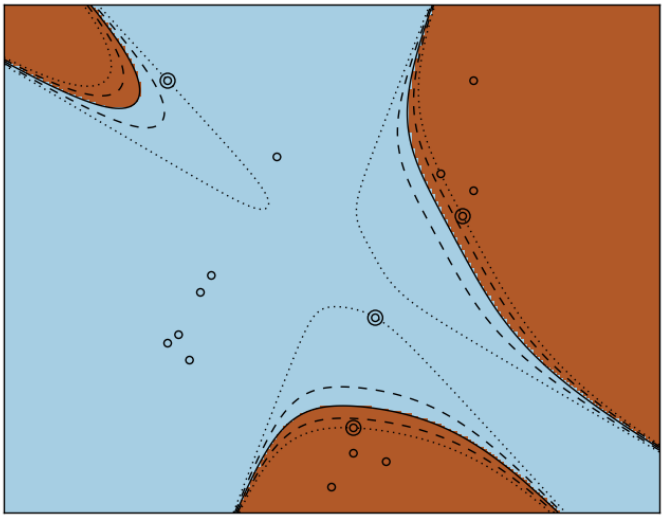
SVM Poly (degree 3)



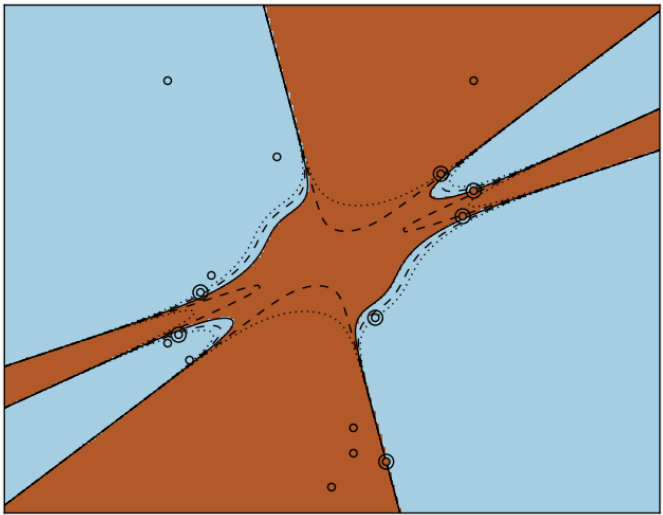
SVM Poly (degree 4)



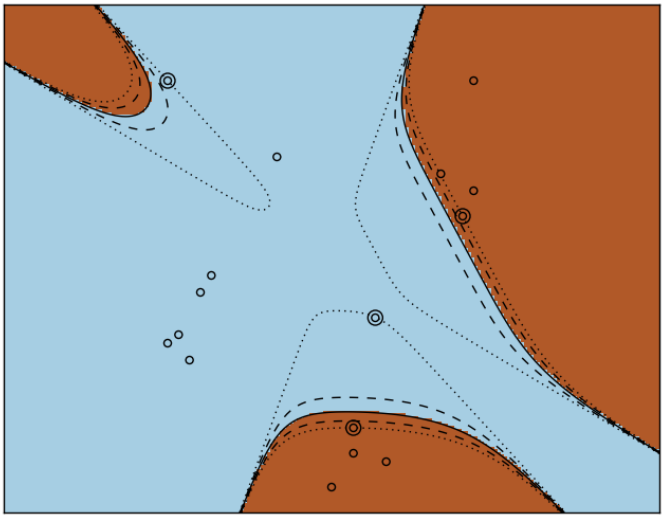
SVM Poly (degree 5)



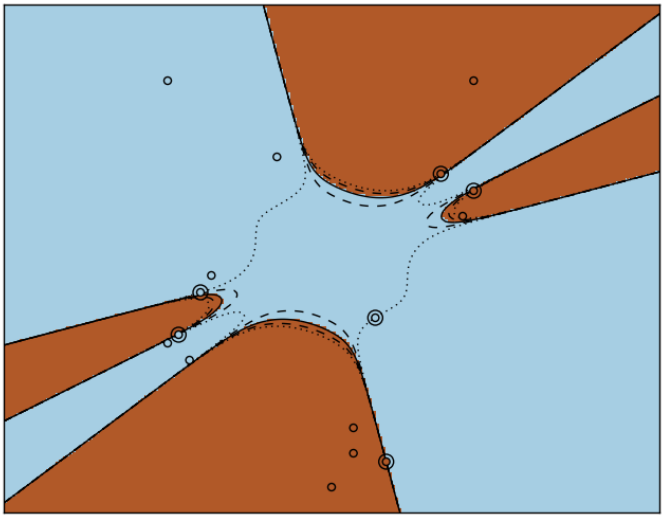
SVM Poly (degree 6)



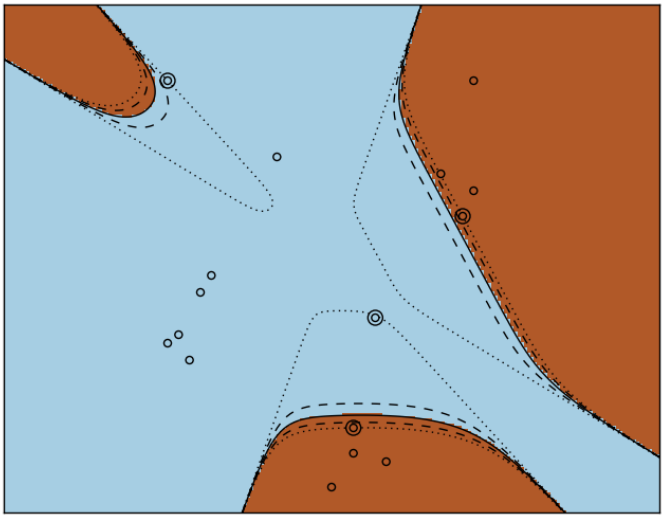
SVM Poly (degree 7)



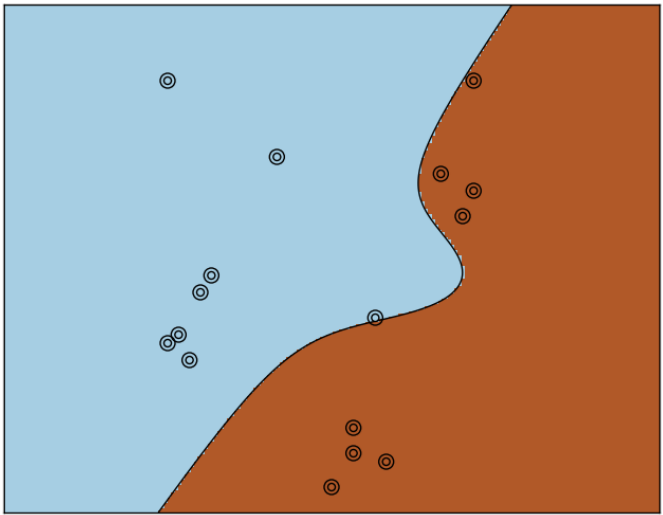
SVM Poly (degree 8)



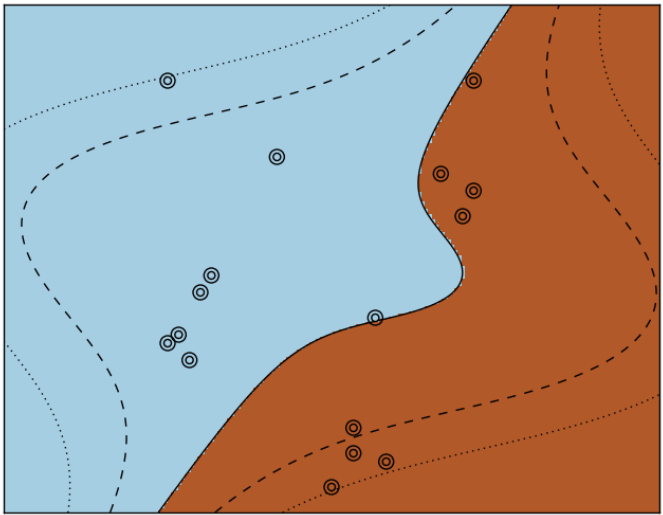
SVM Poly (degree 9)



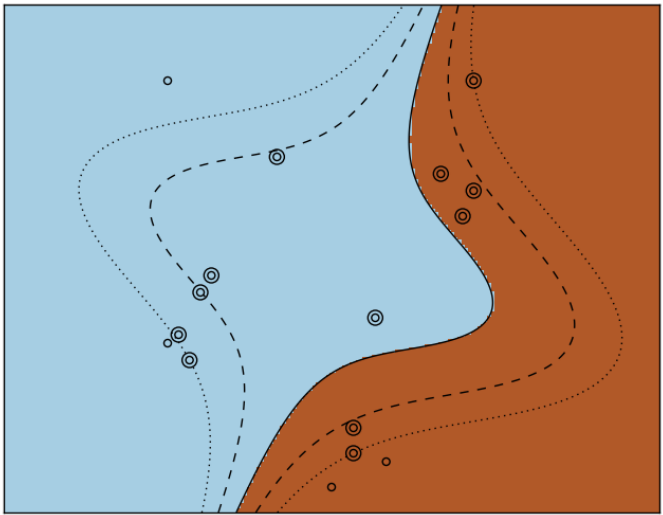
SVM Poly (degree 3, gamma 0.05)



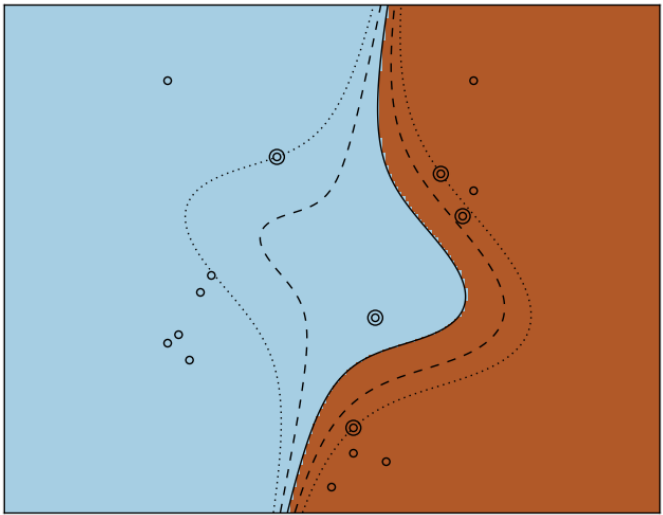
SVM Poly (degree 3, gamma 0.1)



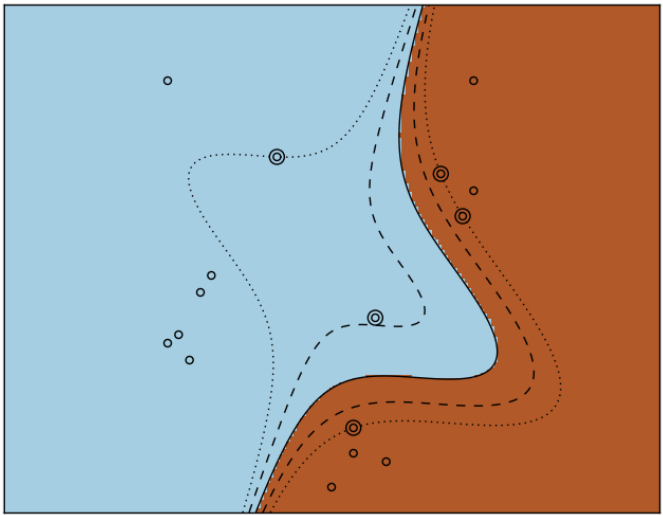
SVM Poly (degree 3, gamma 0.2)



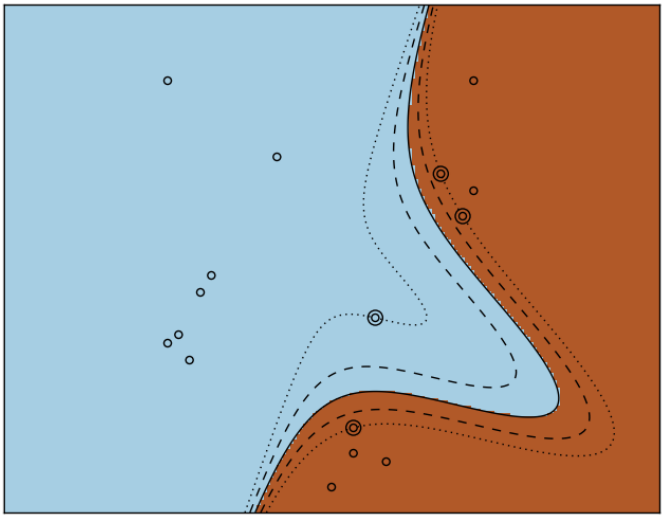
SVM Poly (degree 3, gamma 0.5)



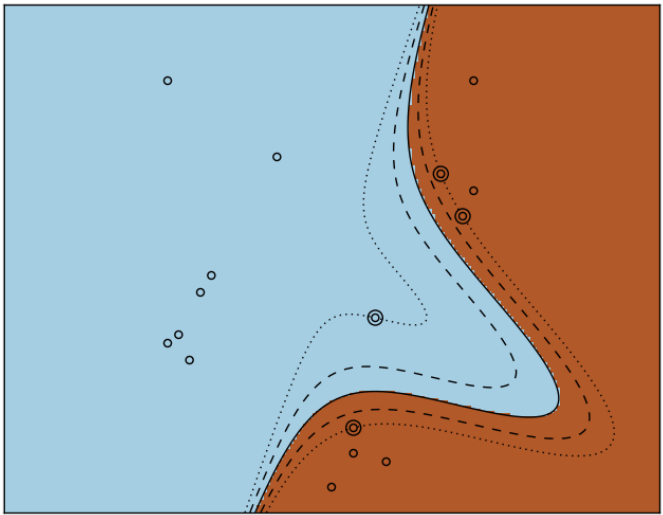
SVM Poly (degree 3, gamma 0.7)



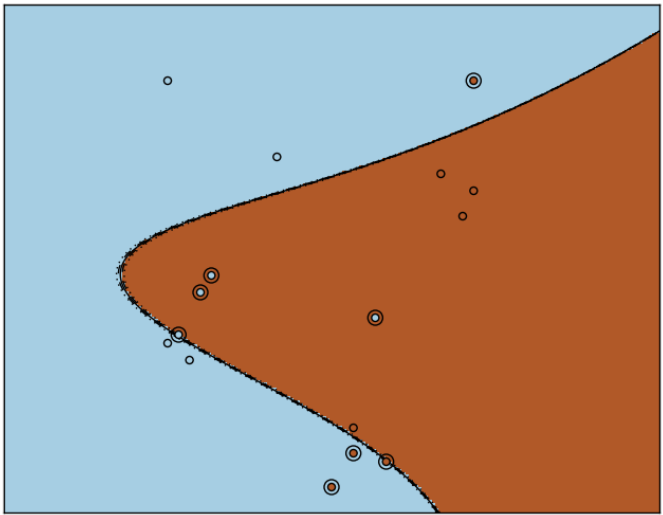
SVM Poly (degree 3, gamma 1)



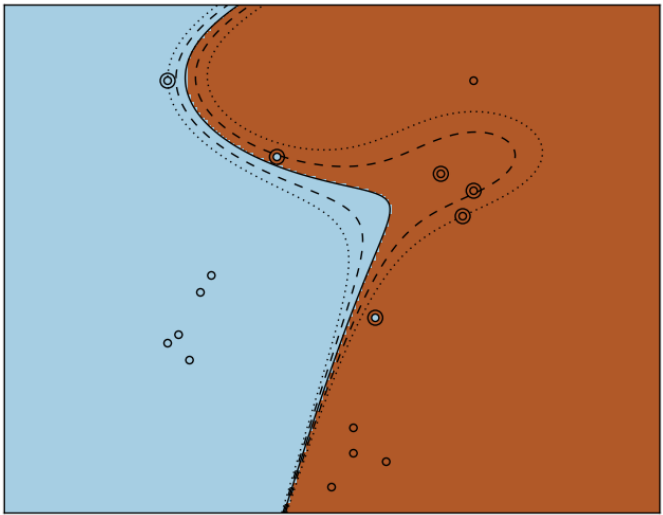
SVM Poly (degree 3, gamma 2)



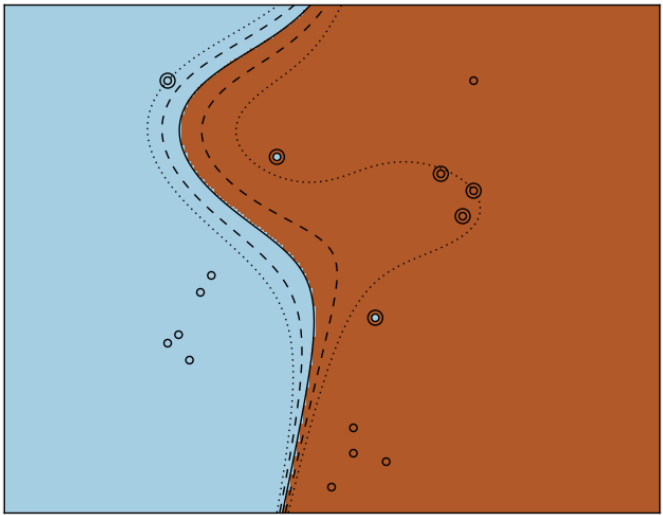
SVM Poly (d=3, g=0.5, coef=-2.0)



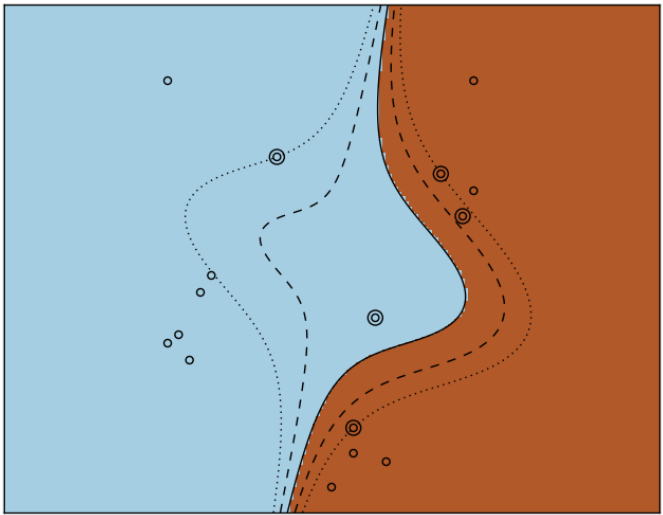
SVM Poly (d=3, g=0.5, coef=-1.0)



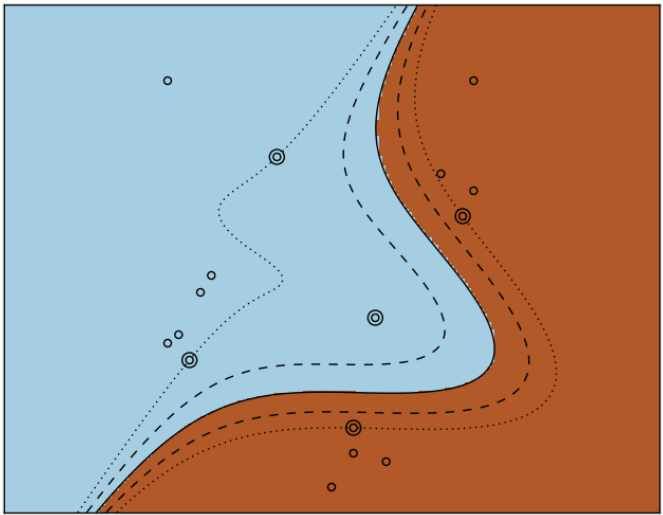
SVM Poly (d=3, g=0.5, coef=-0.50)



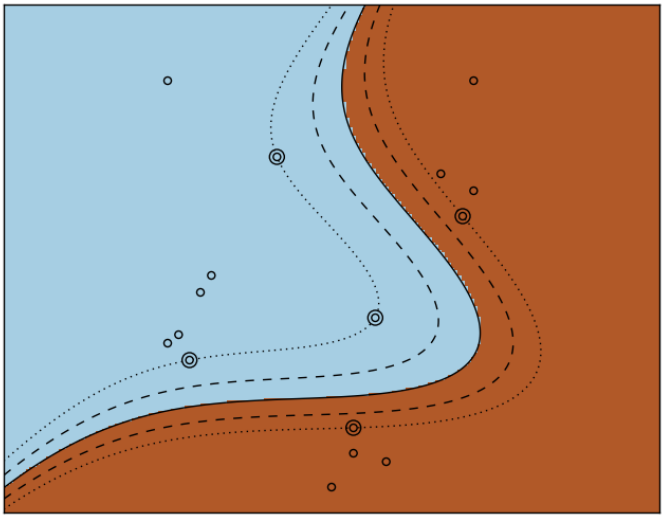
SVM Poly (d=3, g=0.5, coef=0)



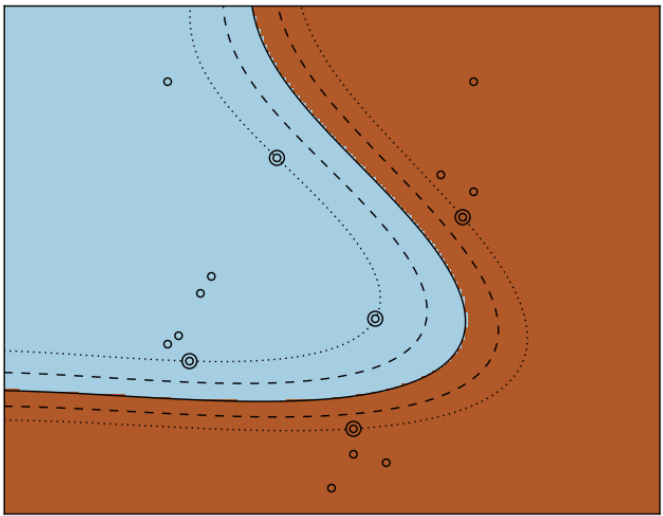
SVM Poly (d=3, g=0.5, coef=0.5)



SVM Poly (d=3, g=0.5, coef=1)

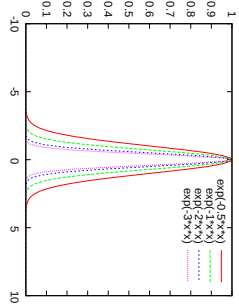


SVM Poly (d=3, g=0.5, coef=2)



RBF Kernels

$$k(\mathbf{x}, \mathbf{y}) = exp(-\gamma \|\mathbf{x} - \mathbf{y}\|^2); \gamma > 0$$



- Each training point creates its bell.
- Overall shape is the sum of the bells.
- Kind of “all nearest neighbours”.

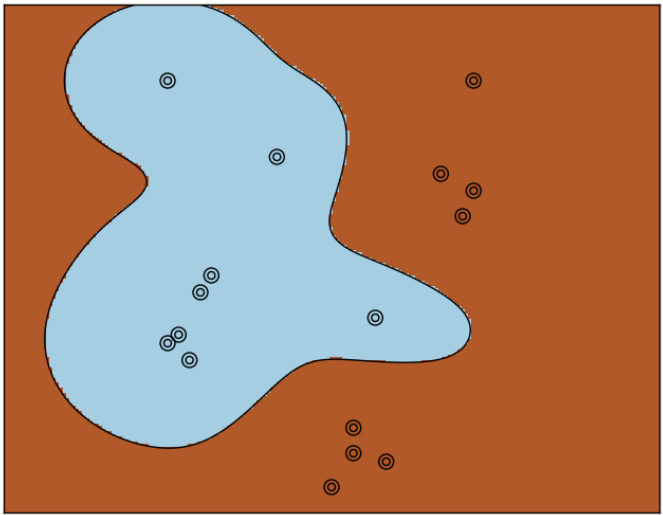
RBF Kernel Parameters

C	Decision Surface	Model	Bias	Variance
Low	Smooth	Simple	High	Low
High	Peaked	Complex	Low	High

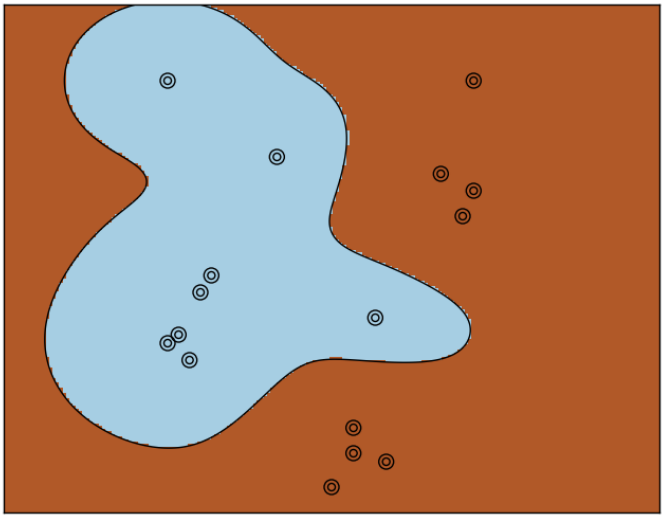
gamma	Affected Points
Low	can be far from training examples
High	must be close to training examples

- Does higher gamma lead to higher variance?
- Choice critical for SVM performance.
- Advised to use GridSearchCV for C and gamma:
  - exponentially spaced probes
  - wide range

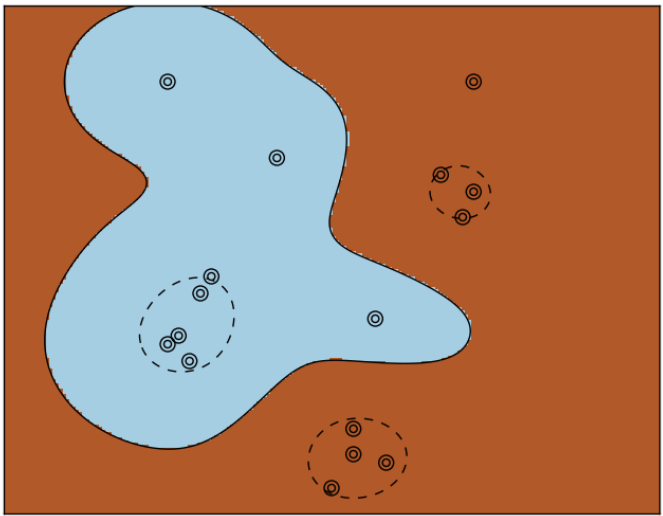
SVM RBF (C=0.05, gamma=2)



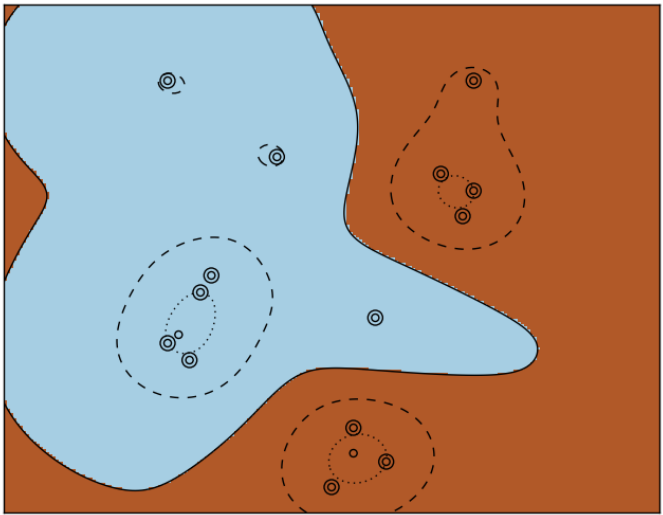
SVM RBF (C=0.1, gamma=2)



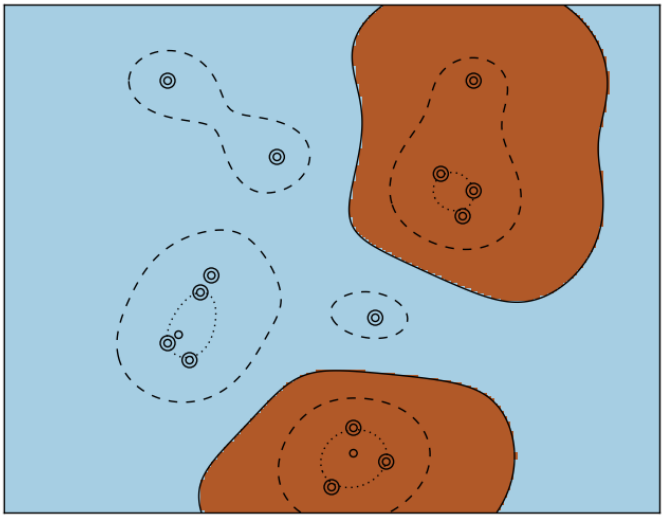
SVM RBF (C=0.2, gamma=2)



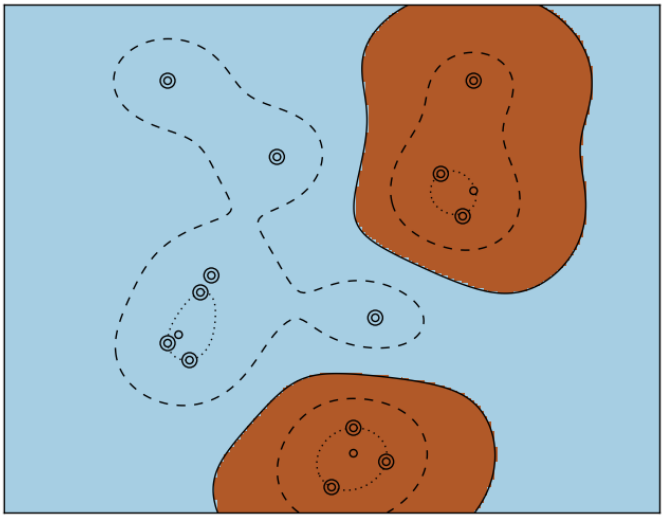
SVM RBF (C=0.5, gamma=2)



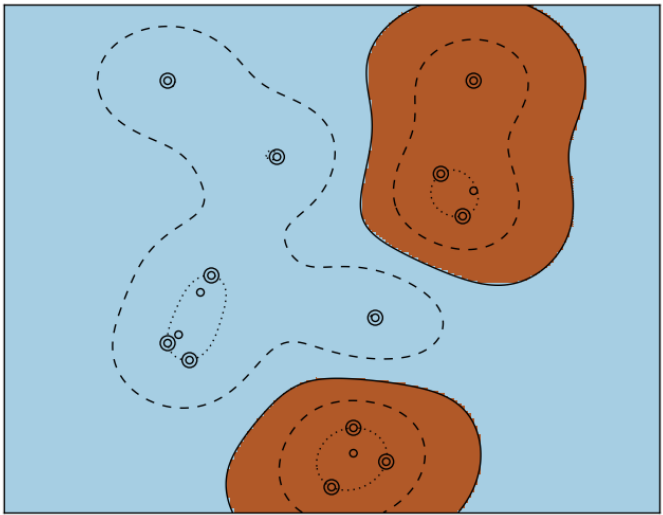
SVM RBF (C=0.6, gamma=2)



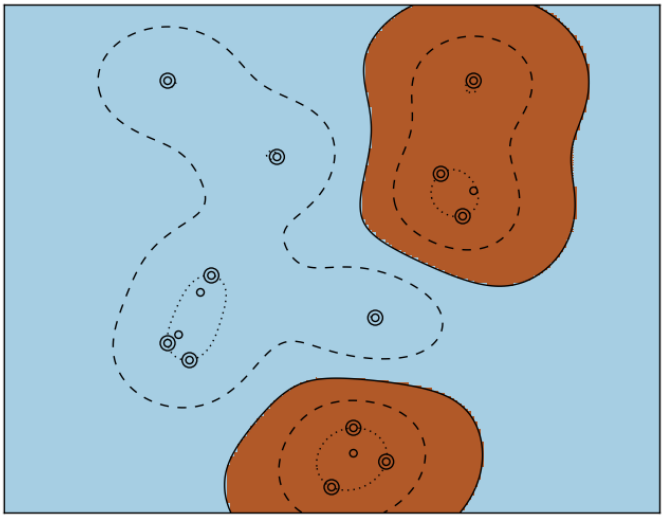
SVM RBF (C=0.7, gamma=2)



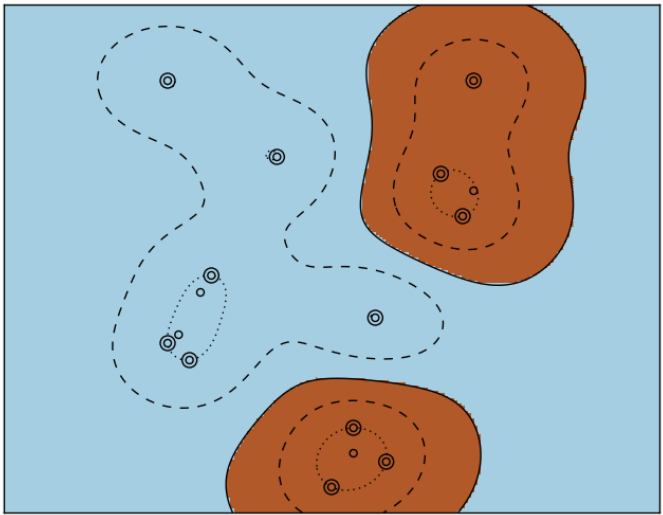
SVM RBF (C=1, gamma=2)



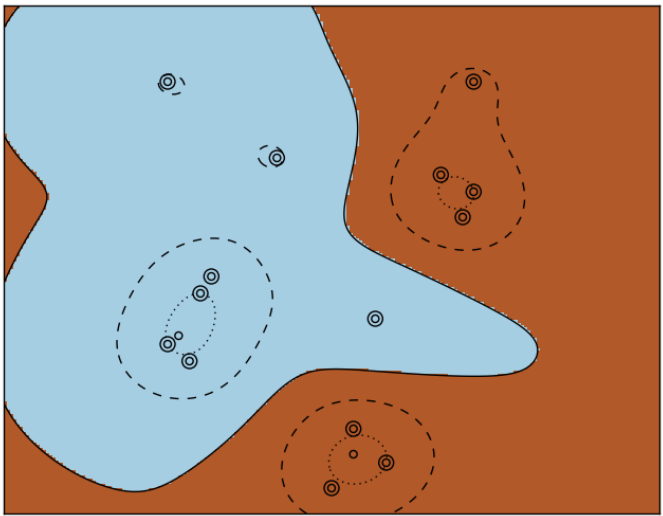
SVM RBF (C=2, gamma=2)



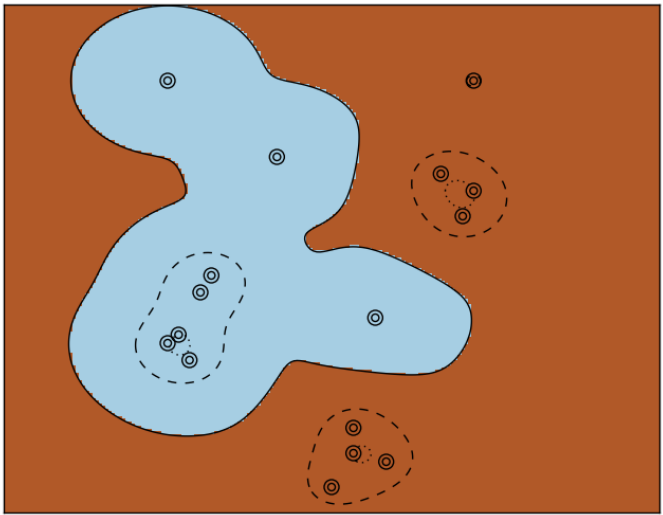
SVM RBF (C=1, gamma=2)



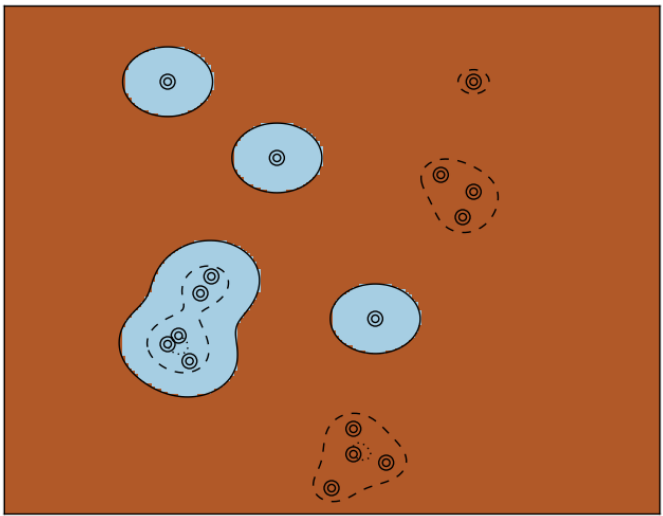
SVM RBF (C=0.5, gamma=2)



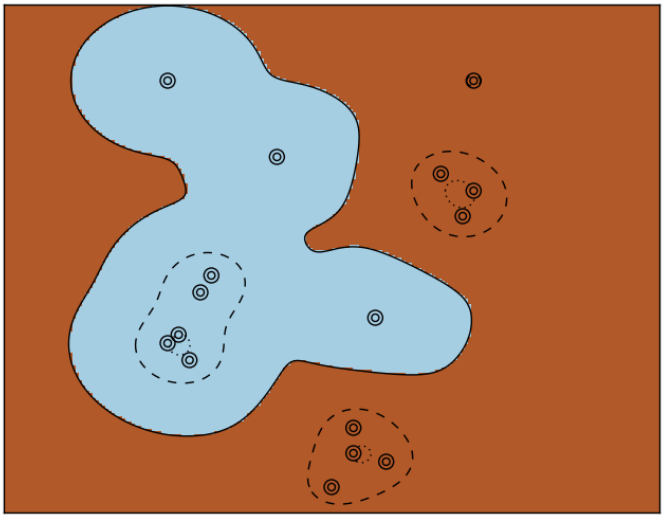
SVM RBF (C=0.5, gamma=5)



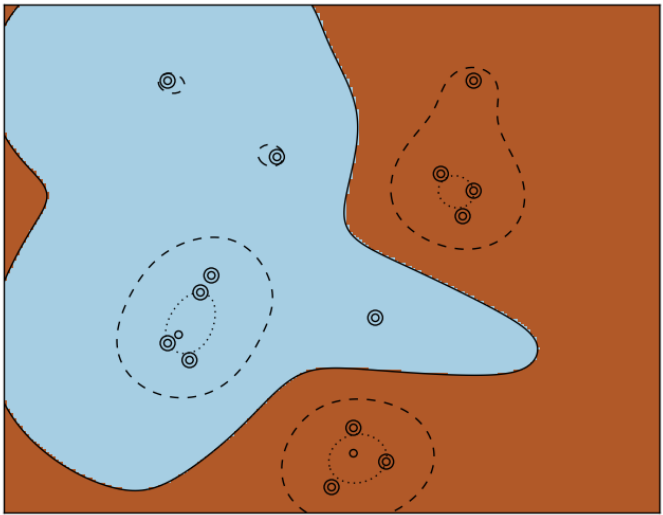
SVM RBF (C=0.5, gamma=10)



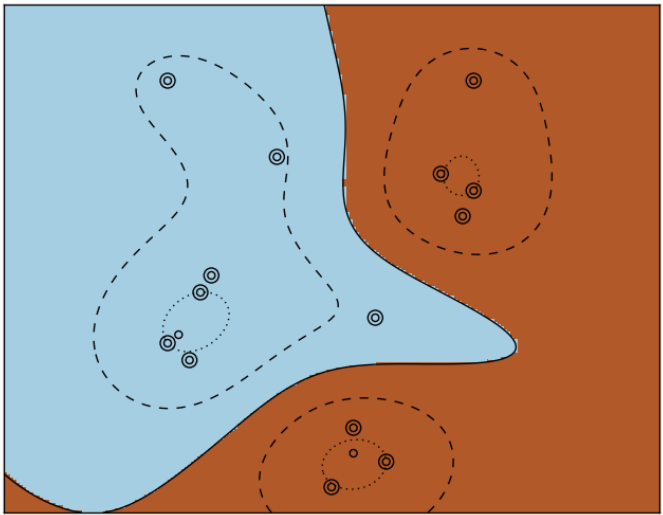
SVM RBF (C=0.5, gamma=5)



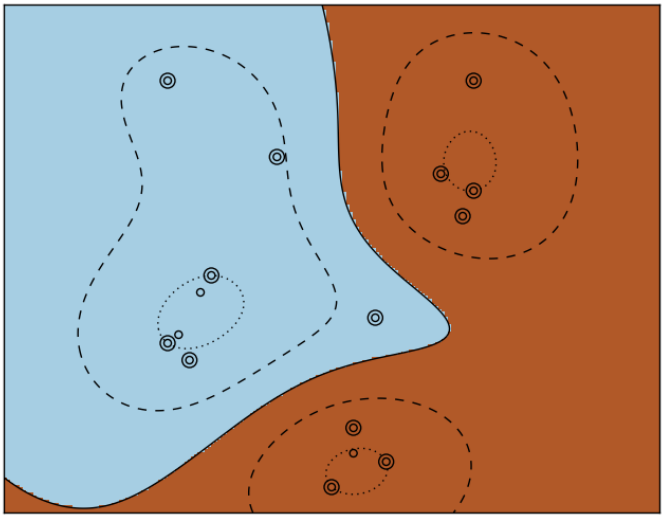
SVM RBF (C=0.5, gamma=2)



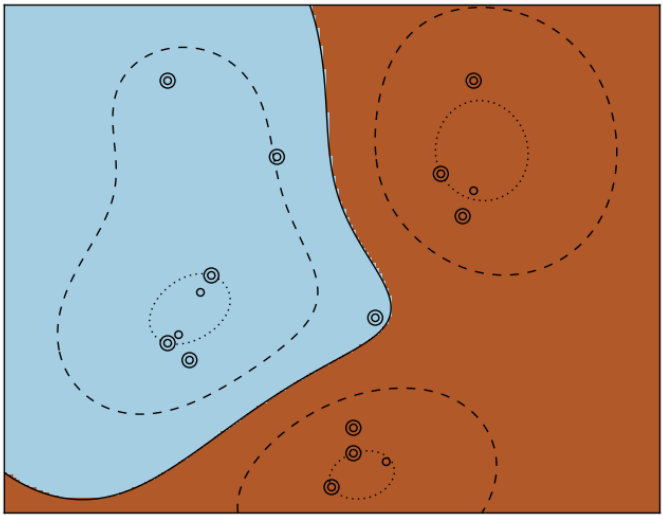
SVM RBF (C=0.5, gamma=1)



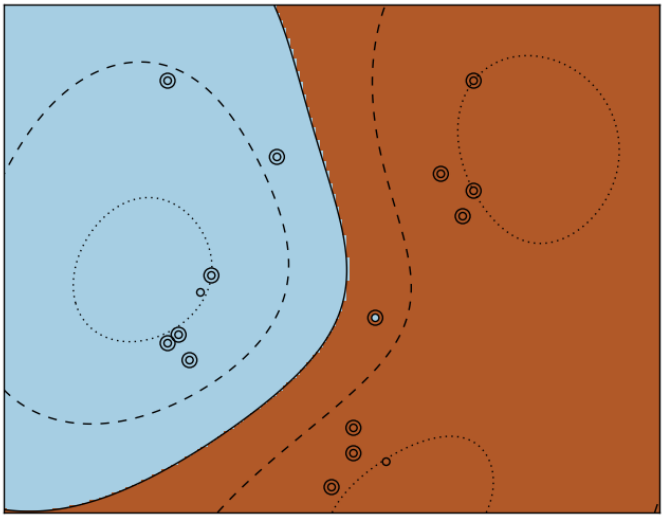
SVM RBF (C=0.5, gamma=0.7)



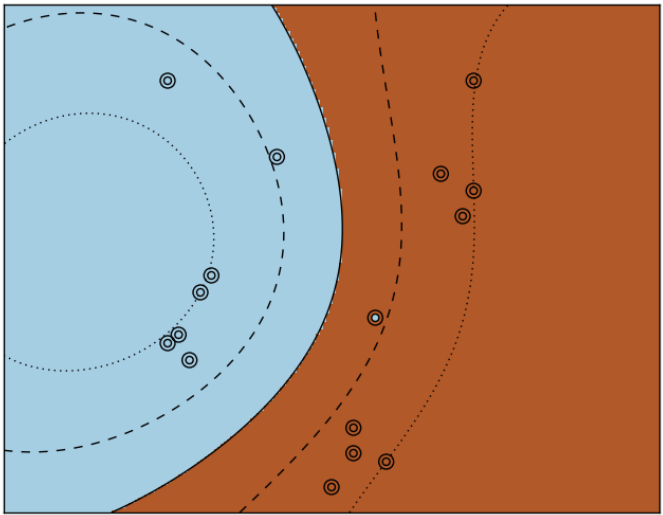
SVM RBF (C=0.5, gamma=0.5)



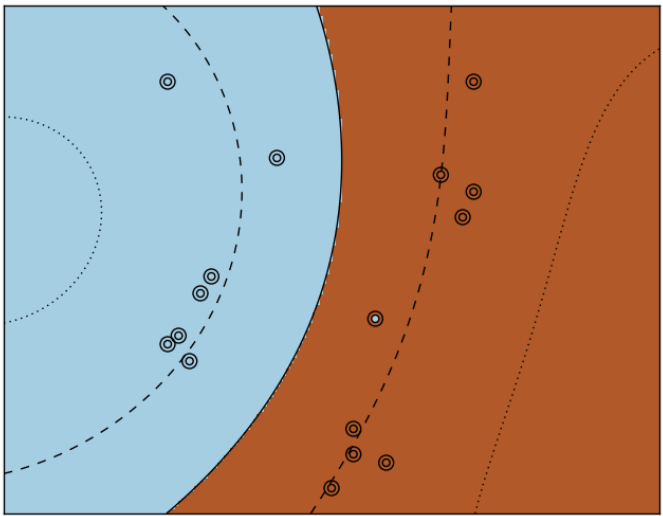
SVM RBF (C=0.5, gamma=0.2)



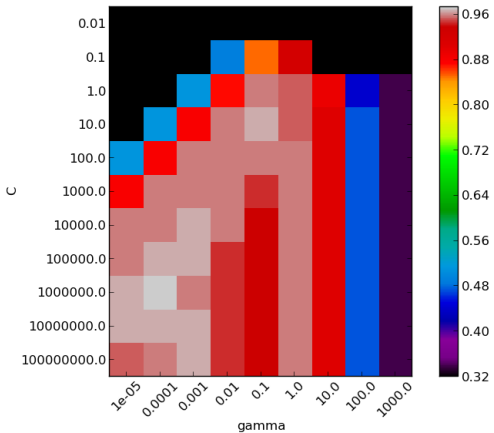
SVM RBF (C=0.5, gamma=0.1)



SVM RBF (C=0.5, gamma=0.05)



Cross-validation Heatmap



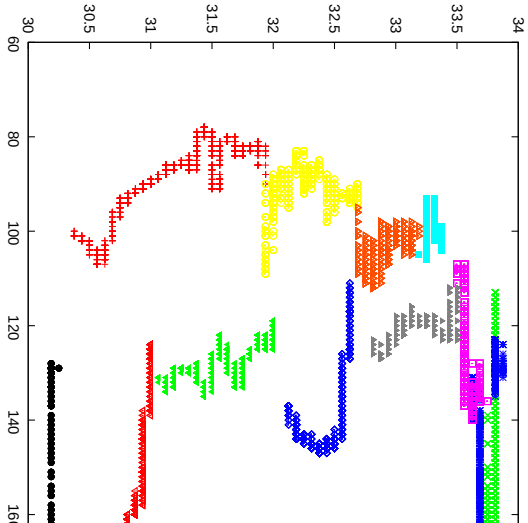
http://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html

Multi-class SVM

Two implementations in scikit-learn:

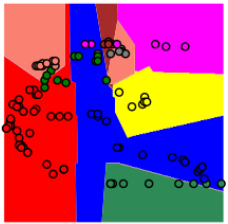
- SVC: one-against-one
  - $n(n-1)/2$ classifiers constructed
  - supports various kernels, incl. custom ones
- LinearSVC: one-vs-the-rest
  - n classifiers trained

PAMAP-easy Training Data

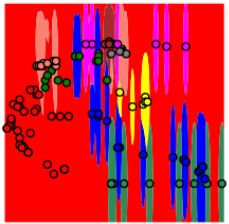


Default View (every 200)

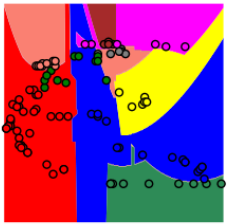
SVC with linear kernel



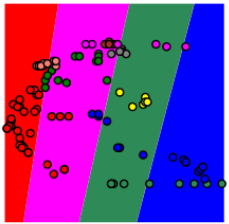
SVC with RBF kernel (gamma 0.7)



SVC with polynomial (degree 3) kernel



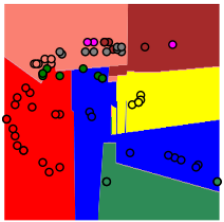
LinearSVC (linear kernel)



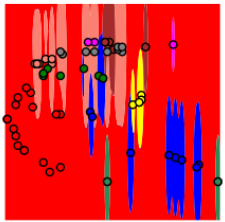
regularization: C=1.0

Default View (every 300)

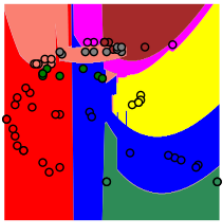
SVC with linear kernel



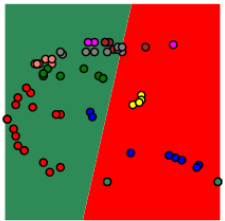
SVC with RBF kernel (gamma 0.7)



SVC with polynomial (degree 3) kernel



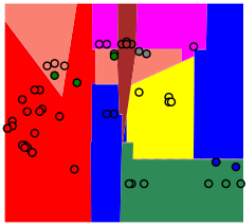
LinearSVC (linear kernel)



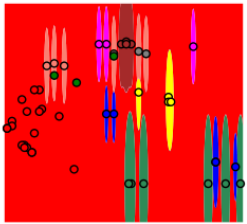
regularization: C=1.0

Default View (every 400)

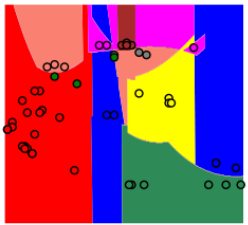
SVC with linear kernel



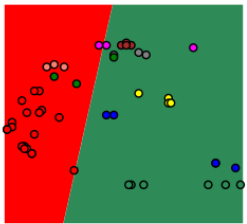
SVC with RBF kernel (gamma 0.7)



SVC with polynomial (degree 3) kernel



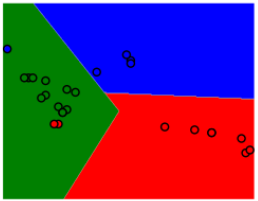
LinearSVC (linear kernel)



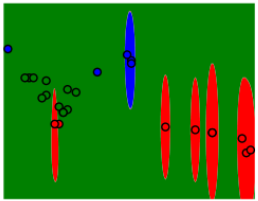
regularization: C=1.0

Regularization C=0.5

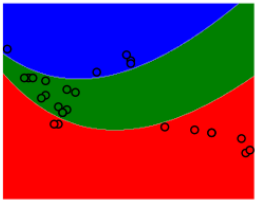
SVC with linear kernel



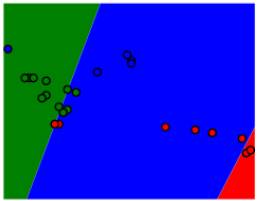
SVC with RBF kernel (gamma 0.7)



SVC with polynomial (degree 3) kernel

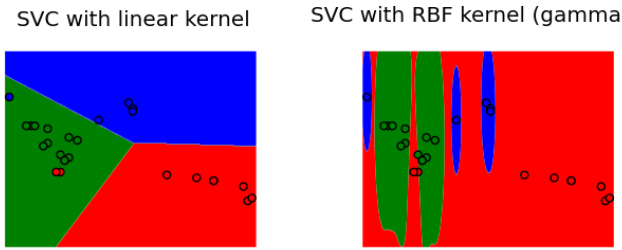


LinearSVC (linear kernel)



regularization: C=0.5

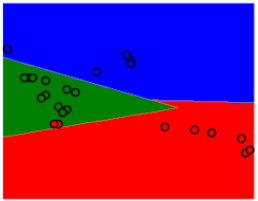
Regularization C=1



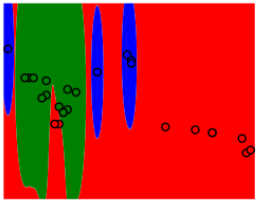
regularization: C=1.0

Regularization C=5

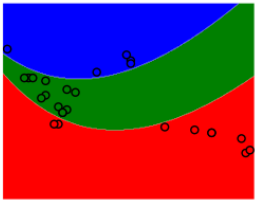
SVC with linear kernel



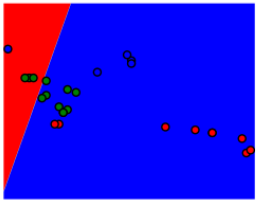
SVC with RBF kernel (gamma 0.7)



SVC with polynomial (degree 3) kernel



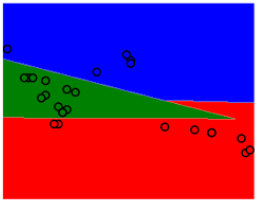
LinearSVC (linear kernel)



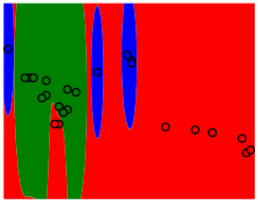
regularization: C=5.0

Regularization C=10

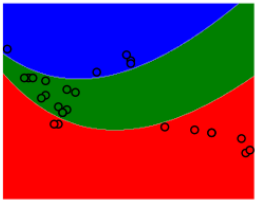
SVC with linear kernel



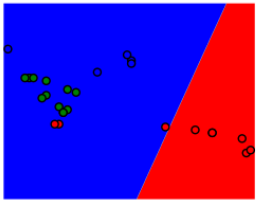
SVC with RBF kernel (gamma 0.7)



SVC with polynomial (degree 3) kernel



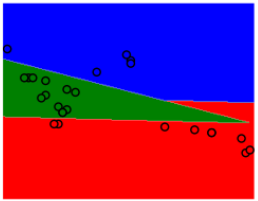
LinearSVC (linear kernel)



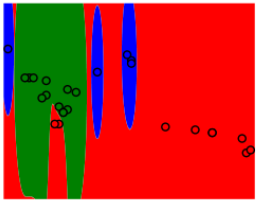
regularization: C=10.0

Regularization C=20

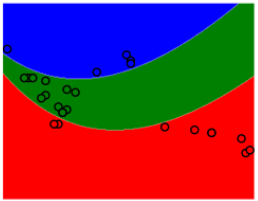
SVC with linear kernel



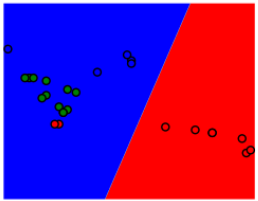
SVC with RBF kernel (gamma 0.7)



SVC with polynomial (degree 3) kernel



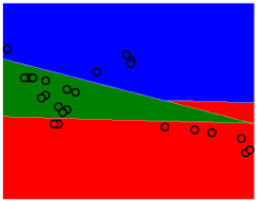
LinearSVC (linear kernel)



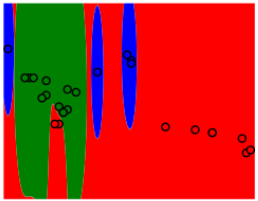
regularization: C=20.0

Regularization C=50

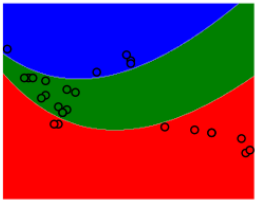
SVC with linear kernel



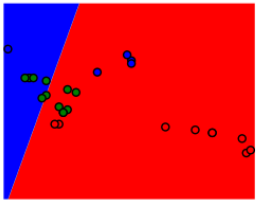
SVC with RBF kernel (gamma 0.7)



SVC with polynomial (degree 3) kernel



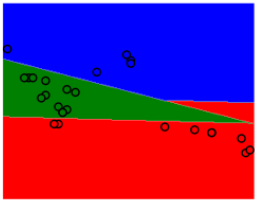
LinearSVC (linear kernel)



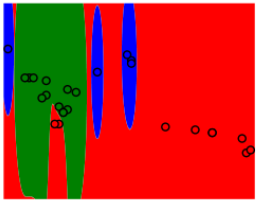
regularization: C=50.0

Regularization C=500

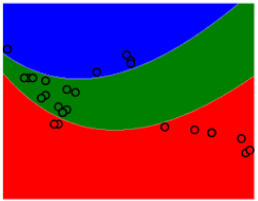
SVC with linear kernel



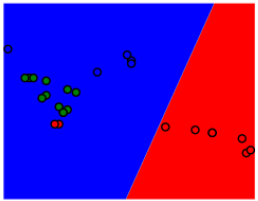
SVC with RBF kernel (gamma 0.7)



SVC with polynomial (degree 3) kernel

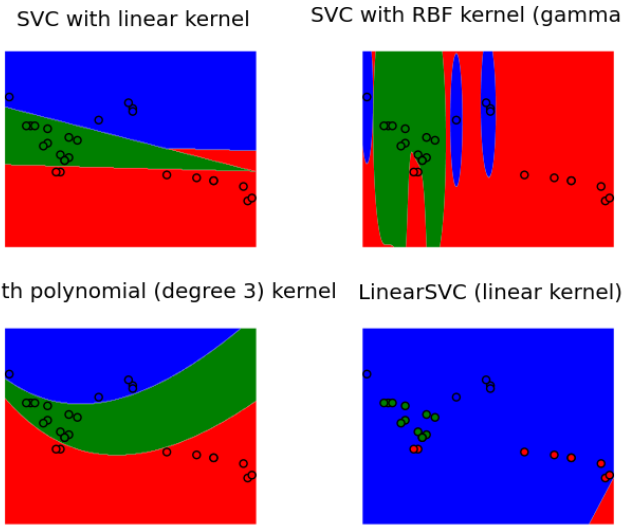


LinearSVC (linear kernel)



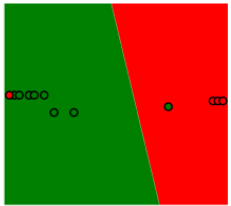
regularization: C=500.0

Regularization C=5000

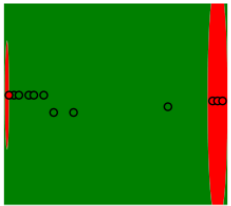


Inseparable classes 12,13 (every 200)

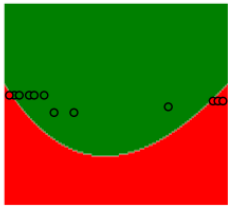
SVC with linear kernel



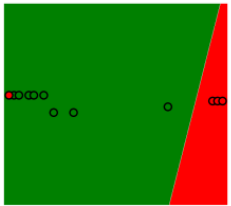
SVC with RBF kernel (gamma 0.7)



SVC with polynomial (degree 3) kernel

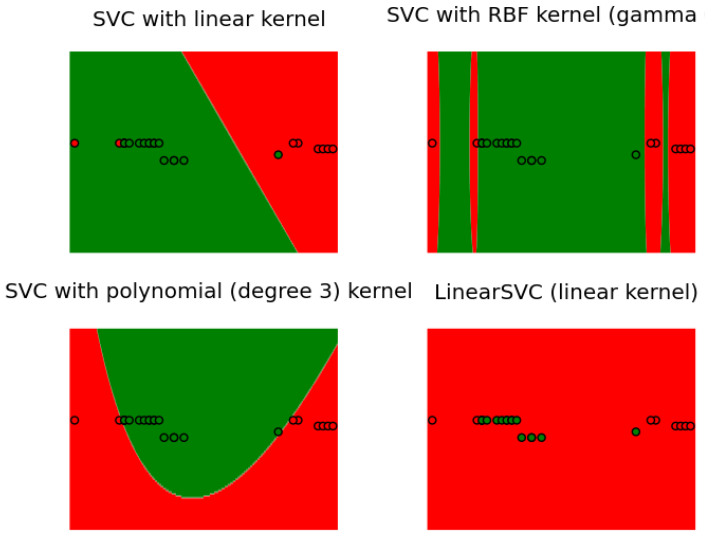


LinearSVC (linear kernel)



regularization: C=1.0

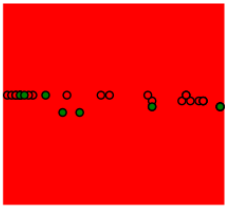
Inseparable classes 12,13 (every 100)



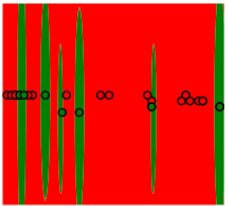
regularization: C=1.0

Inseparable classes 12,13 (every 80)

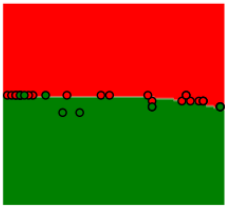
SVC with linear kernel



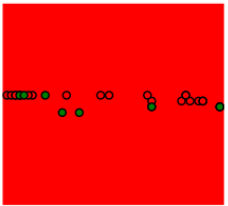
SVC with RBF kernel (gamma 0.7)



SVC with polynomial (degree 3) kernel

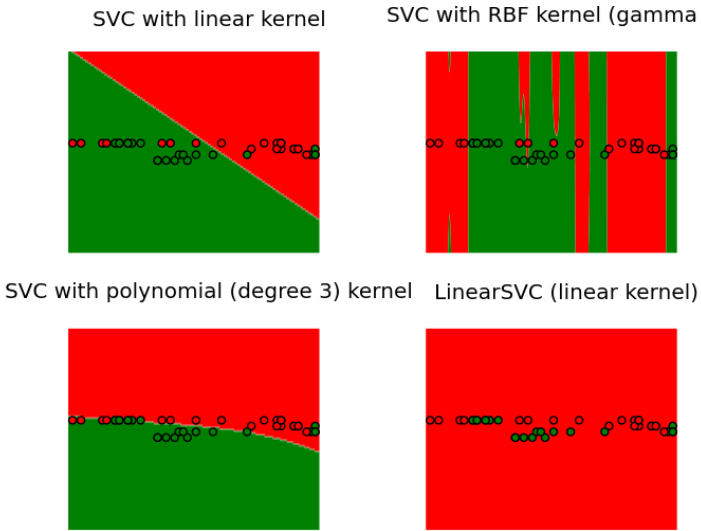


LinearSVC (linear kernel)



regularization: C=1.0

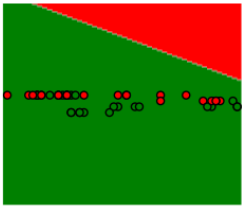
Inseparable classes 12,13 (every 60)



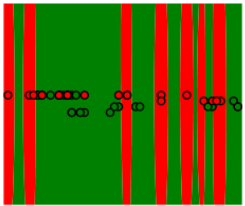
regularization: C=1.0

Inseparable classes 12,13 (every 55)

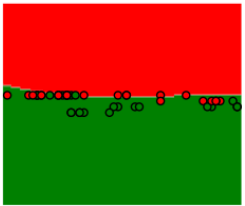
SVC with linear kernel



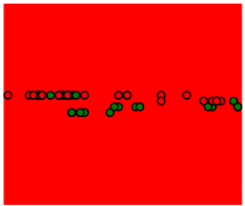
SVC with RBF kernel (gamma 0.7)



SVC with polynomial (degree 3) kernel



LinearSVC (linear kernel)



regularization: C=1.0

Summary

Diagnostics:

- Visualization is the first, quick and easy but very effective.
- Principled diagnostics:
  - Bias vs. Variance.
  - Optimizer (Search) Error vs. Objective function (Modelling) Error.
  - Error Analysis vs. Ablative Analysis

Kernels and Effects on Hyperparameters Visualizations:

- Linear, Polynomial and RBF Kernels.
- Cross-validation for the choice of hyperparameters.
- Multi-class SVM.

For hw_gridsearch see the web.