Midterm Recap

David Rosenberg

New York University

June 9, 2015

New Features

- (**True or False**, 1 pt) When using (unregularized) linear regression, adding new features always improves the performance on training data, or at least never make it worse.
- (**True or False**, 1 pt)When using a (unregularized) linear regression, adding new features always improves the performance on test data, or at least never make it worse.

- Adding new features makes a bigger hypothesis space
  - decrease training error
  - could lead to overfitting
- Said "unregularized linear regression", because regularization can prevent overfitting

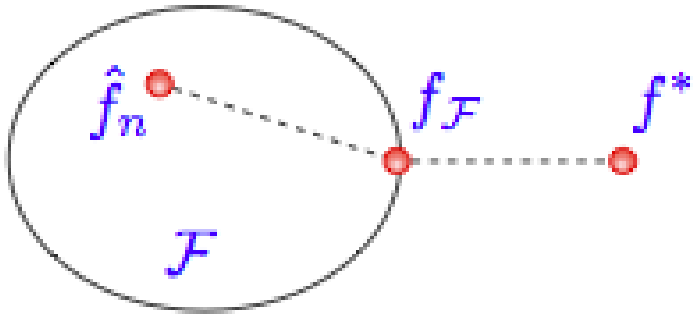
Overfitting

- (**True or False**, 1 pt) Overfitting is more likely when the set of training data is small.
- (**True or False**, 1 pt) Overfitting is more likely when the hypothesis space is small.

- Whether you overfit depends on
  - size of the training set and
  - the size of the hypothesis space

Estimation Error / Approximation Error

- (**True or False**, 1 pt) Approximation error decreases to zero as the amount of training data goes to infinity.
- (**True or False**, 1 pt) If the empirical risk function is not convex, more training data may not help estimation error.



$$f^* = \arg\min_f \mathbb{E}\ell(f(X), Y)$$

$$f_{\mathcal{F}} = \arg\min_{f \in \mathcal{F}} \mathbb{E}\ell(f(X), Y))$$

$$\hat{f}_n = \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i), y_i)$$

Duplicate Features in a Tree

- (**True or False**, 1 pt) If a decision tree is trained on data for which two features are exactly equal, the resulting tree will be the same whether or not we remove one of those two features.

Feature Rescaling

(**True or False**, 1 pt) Suppose we fit Lasso regression to a data set. If we rescale one of the features by multiplying it by 10, and we then refit Lasso regression with the same regularization parameter, then it is more likely for that feature to be excluded from the model

- Big feature values $\implies$ smaller coefficients $\implies$ less lasso penalty $\implies$ more likely to have be kept

Kernel Methods Scalability

(**True or False**, 1 pt) When you have a very large data set of size n, which is much larger than the dimension d of the feature space, kernel methods are probably not a good idea.

- At the heart of kernel methods is the kernel matrix, which is $n \times n$.
- If n is huge, kernel methods become much more difficult.
- If feature space is also much smaller, no obvious reason to use kernelized approach.

AdaBoost Converges to Zero Training Error?

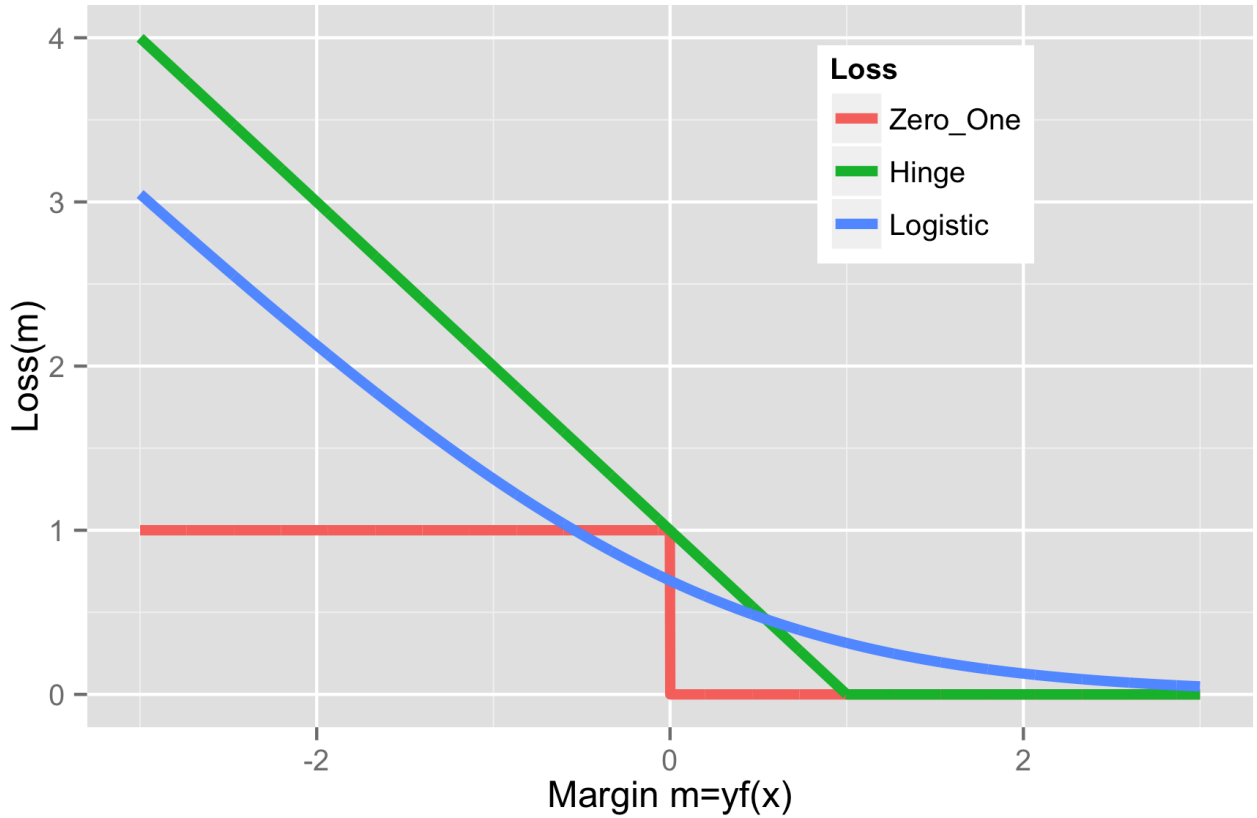
- (**True or False**, 1 pt) Adaboost with decision stumps will eventually reach zero training error, provided we run enough rounds of boosting.
- The best any method can do is to converge to

$$\min_f \frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i), y_i)$$

- But this may not be zero.

Support Vectors

- (1 pt) Circle all of the loss functions that may lead to sparsity of support vectors: **exponential loss, hinge loss, squared hinge loss, logistic loss, square loss**.



ℓ_1/ℓ_2 regularization

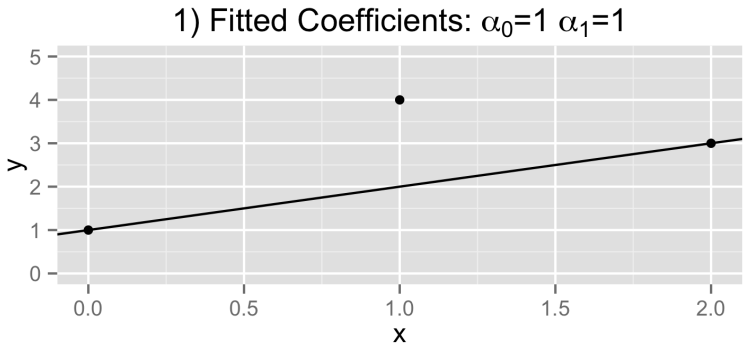
- (4 pts) We have a dataset $\mathcal{D} = \{(0,1), (1,4), (2,3)\}$ that we fit by minimizing an objective function of the form:

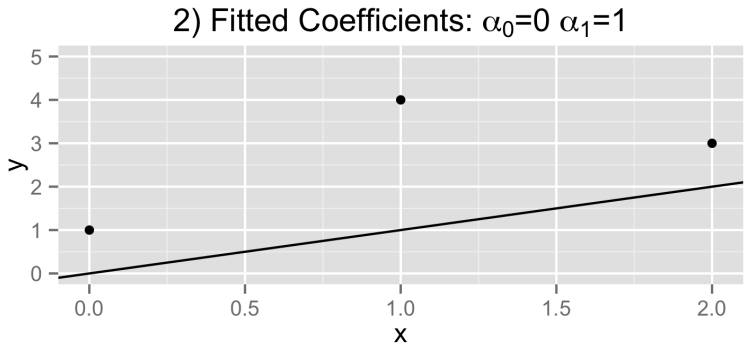
$$J(\alpha_0, \alpha_1) = \sum_{i=1}^{3} (\alpha_0 + \alpha_1 x_i - y_i)^2 + \lambda_1 |\alpha_0 + \alpha_1| + \lambda_2(\alpha_0^2 + \alpha_1^2),$$

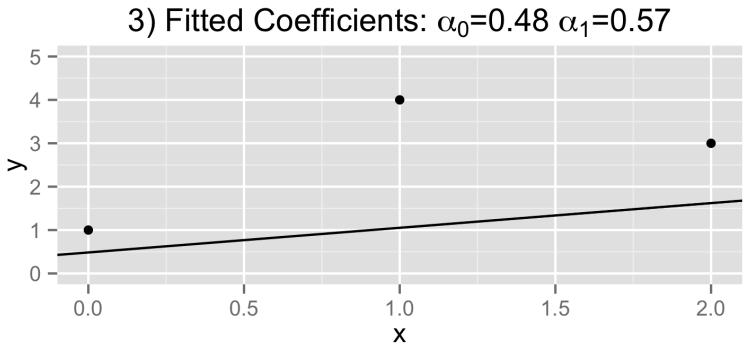
and the corresponding fitted function is given by $f(x) = \alpha_0 + \alpha_1 x$. We tried four different settings of λ_1 and λ_2, and the results are shown in Figure. For each of the following parameter settings, give the number of the plot that shows the resulting fit.

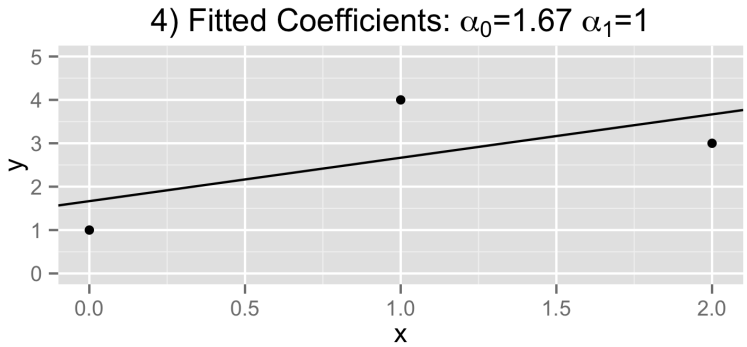
1. (1 pt) $\lambda_1 = 0$ and $\lambda_2 = 0$.
2. (1 pt) $\lambda_1 = 5$ and $\lambda_2 = 0$.
3. (1 pt) $\lambda_1 = 0$ and $\lambda_2 = 10$.
4. (1 pt) $\lambda_1 = 0$ and $\lambda_2 = 2$.

ℓ_1/ℓ_2 regularization









Kernel Function

- Show that the following kernel function is a Mercer kernel (i.e. it represents an inner product):

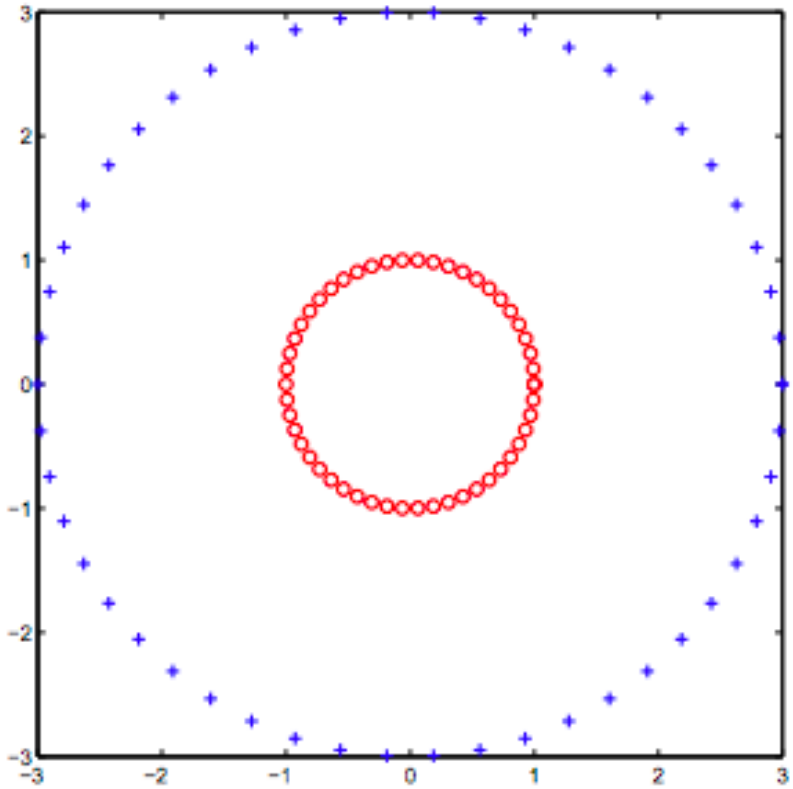
$$k(x, y) = \frac{x^T y}{\|x\| \|y\|},$$

where $x, y \in \mathbf{R}^d$.
For $\phi(x) = \frac{x}{\|x\|}$, we have

$$k(x, y) = \langle \phi(x), \phi(y) \rangle .$$

Nonlinear Feature Mappings

- (2 pts) Consider the binary classification problem shown in Figure



Denote the input space by $\mathcal{X} = \{(x_1, x_2) \in \mathbf{R}^2\}$. Give a feature mapping for which a linear classifier could perfectly separate the two classes shown.

Nonlinear Feature Mappings

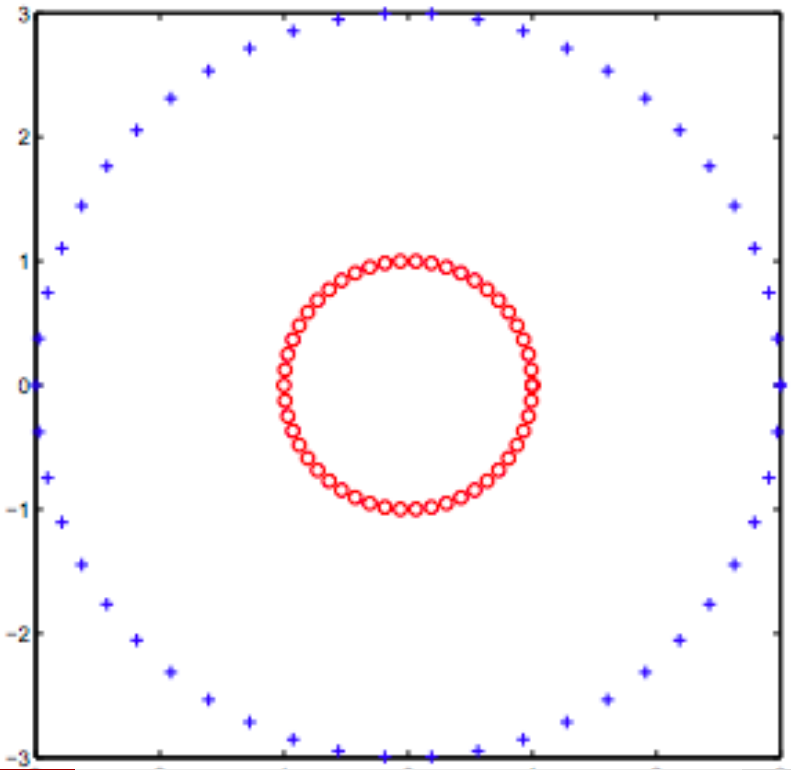
- (2 pts) Consider the binary classification problem shown in Figure

Denote the input space by $\mathcal{X} = \left\{(x_1, x_2) \in \mathbf{R}^2\right\}$. Give a feature mapping for which a linear classifier could perfectly separate the two classes shown.

- $\phi(x) = \left(1, x_1, x_2, x_1^2, x_2^2, x_1 x_2\right)$
- $\phi(x) = \left(x_1^2 + x_2^2\right)$

Hypothesis Space Decision Boundaries

1. (2 pts) For the classification problem in Figure, circle all classifiers that could perfectly separate the classes: **linear SVM, SVM with quadratic kernel, decision stumps (i.e. classification trees with only two leaf nodes), AdaBoost with decision stumps, SVM with radial basis function kernel**.

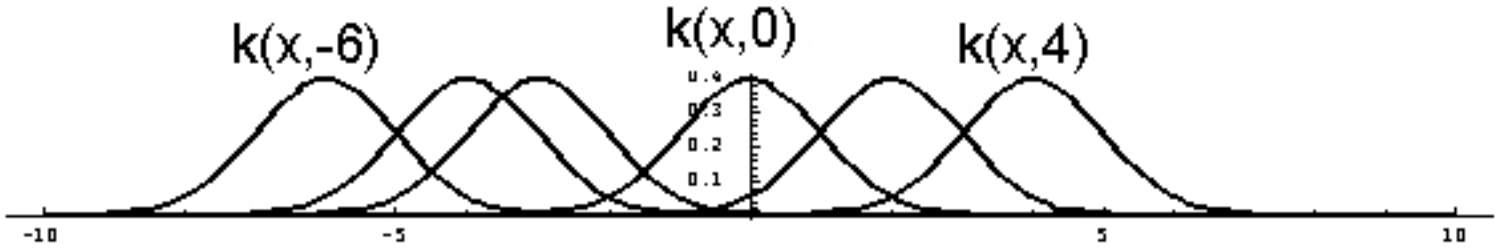


Kernel Machine Prediction Functions

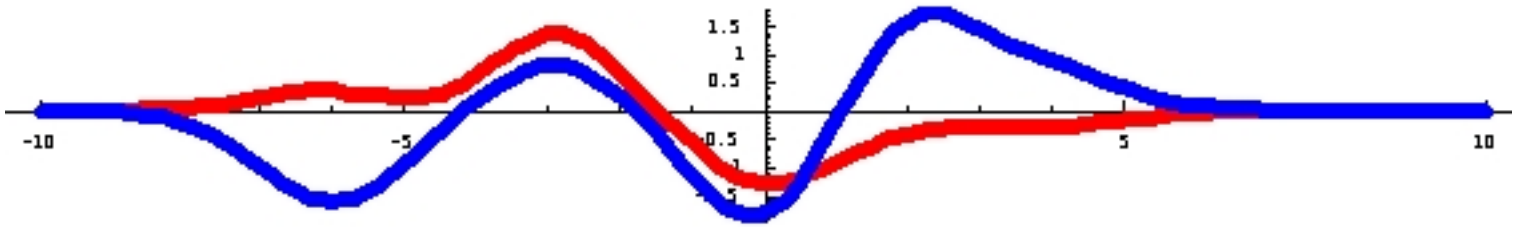
- In general, when we kernelize a linear method, prediction functions have form

$$f^*(x) \quad = \quad \sum_{i=1}^{n} \beta_i k(x_i, x)$$

- Basis function viewpoint – each $k(x_i, \cdot)$ is a basis function:



- Prediction functions look like



Trees vs Linear Classifiers

- (2 pts) Let $\mathcal{F}_1 = \{\text{binary decision trees of depth 2}\}$. Let $\mathcal{F}_2 = \{\text{all linear classifiers}\}$. Draw a binary classification dataset for which a member of $\mathcal{F}_1$ can perfectly separate the data, while no member of $\mathcal{F}_2$ can. Show the splits and the decision boundary for the tree.
- (2 pts) Same $\mathcal{F}_1$ and $\mathcal{F}_2$ as in the previous problem. Draw a binary classification dataset for which a member of $\mathcal{F}_2$ can perfectly separate the data, while no member of $\mathcal{F}_1$ can.

Unnecessary Features

- (1 pt) Consider the following two hypothesis spaces:

$$\mathcal{F}_1 = \{f(x) = e^{w_1}x + w_2 x \mid w_1, w_2 \in \mathbf{R}\} \qquad \mathcal{F}_2 = \{f(x) = wx \mid w \in \mathbf{R}\}$$

- Suppose we are selecting hypotheses using empirical risk minimization (without any penalty).
- Are there any situations in which one of these hypothesis spaces would be preferred to the other? Why?

Restricted Feature

- (1 pt) Consider the following two hypothesis spaces:

$$\mathcal{F}_1 = \{f(x) = e^{w_1} x \mid w_1 \in \mathbf{R}\} \qquad \mathcal{F}_2 = \{f(x) = wx \mid w \in \mathbf{R}\}$$

- Suppose we are selecting hypotheses using empirical risk minimization (without any penalty).
- Are there any situations in which one of these hypothesis spaces would be preferred to the other? Why?

Complexity Constraints for Binary Trees

- (1 pt) Consider the following two hypothesis spaces:

$$\begin{aligned} \mathcal{F}_1 &= \{\text{binary trees of depth at most 2}\} \\ \mathcal{F}_2 &= \{\text{binary trees with at most 4 leaf nodes}\} \end{aligned}$$

- $\mathcal{F}_1$ is contained in $\mathcal{F}_2$
- $\mathcal{F}_2$ allows more tree depth $\implies$ more feature interaction