An Introduction to Learning Theory

Behrouz H. Soleimani

Stan Matwin

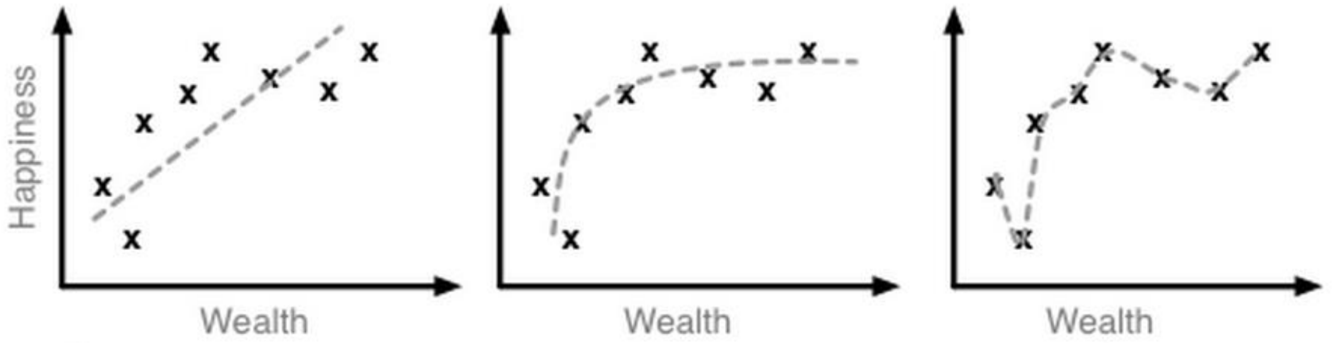
Outline

- Underfitting and Overfitting
- Bias/Variance trade-off
- Model Complexity
- How to avoid overfitting
  - Feature selection (reducing the number of parameters)
  - Regularization (example of Bayesian vs. Frequentist)
  - Model selection (information theoretic approaches, holdout set, cross validation)
- Probabilistic error bounds
  - Generalization error bounds
  - Sample complexity
- Shattering and Vapnik-Chervonenkis (VC) dimension
  - VC dimension of linear classifiers
  - What is the VC dimension of SVM?
- SVM and its strengths
  - Why it is accurate?
  - Why it does not overfit?

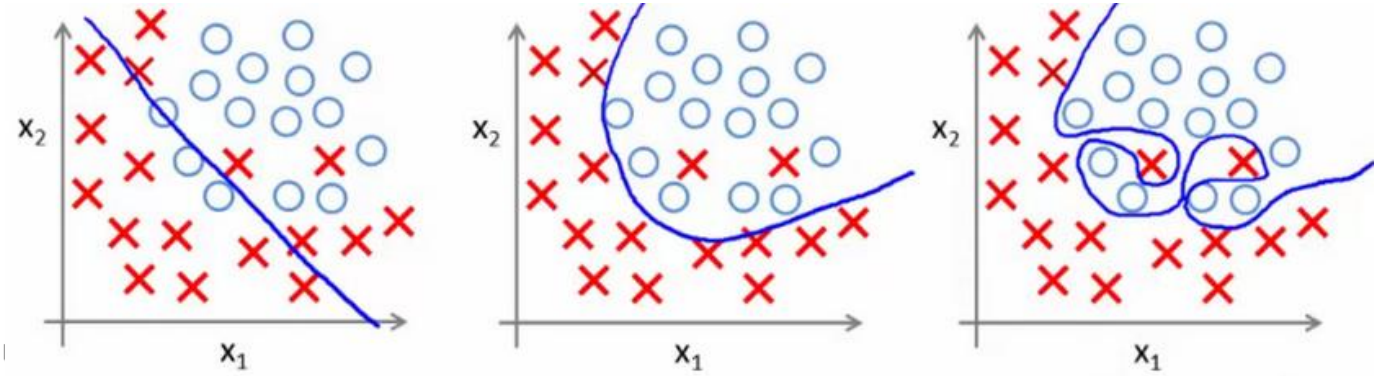
Underfitting & Overfitting

Underfitting and Overfitting

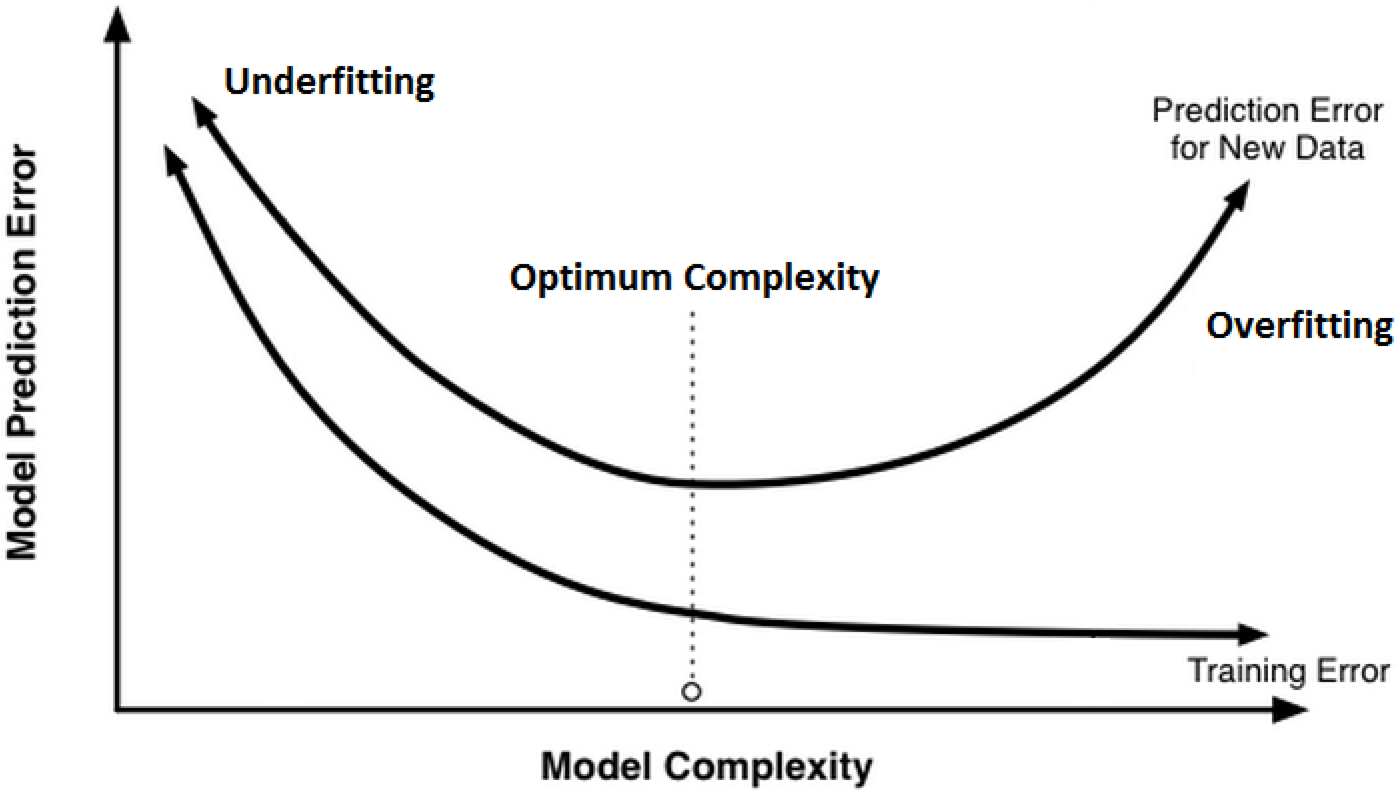
- Regression



- Classification



Underfitting and Overfitting



Bias/Variance Trade-off

Bias/Variance (Concept)

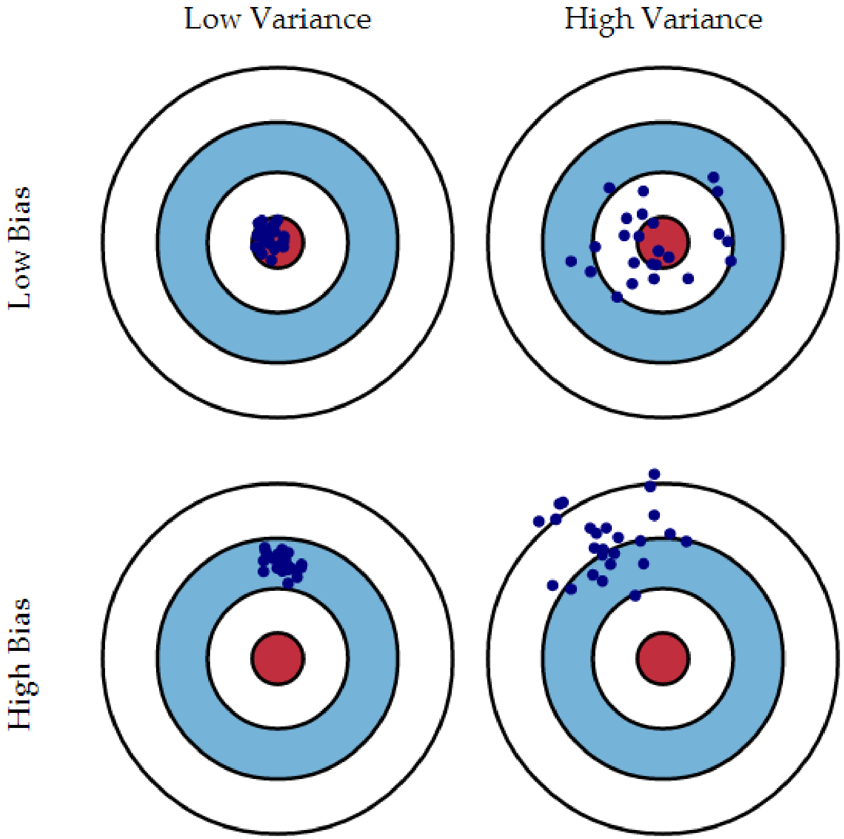
- **Bias:** The error due to bias is taken as the difference between the expected (or average) prediction of our model and the correct value which we are trying to predict.

$$Err_{bias} = \frac{1}{m}\sum_{i=1}^{m}\left(E\left[\hat{f}(x_i)\right] - f(x_i)\right) = E_X\left[E\left[\hat{f}(x)\right] - f(x)\right]$$

- **Variance:** The error due to variance is taken as the variability of a model prediction for a given data point. Again, imagine you can repeat the entire model building process multiple times. The variance is how much the predictions for a given point vary between different realizations of the model.

$$Err_{var} = Var\left(\hat{f}(x)\right) = E\left[\hat{f}(x) - E\left[\hat{f}(x)\right]\right]^2$$

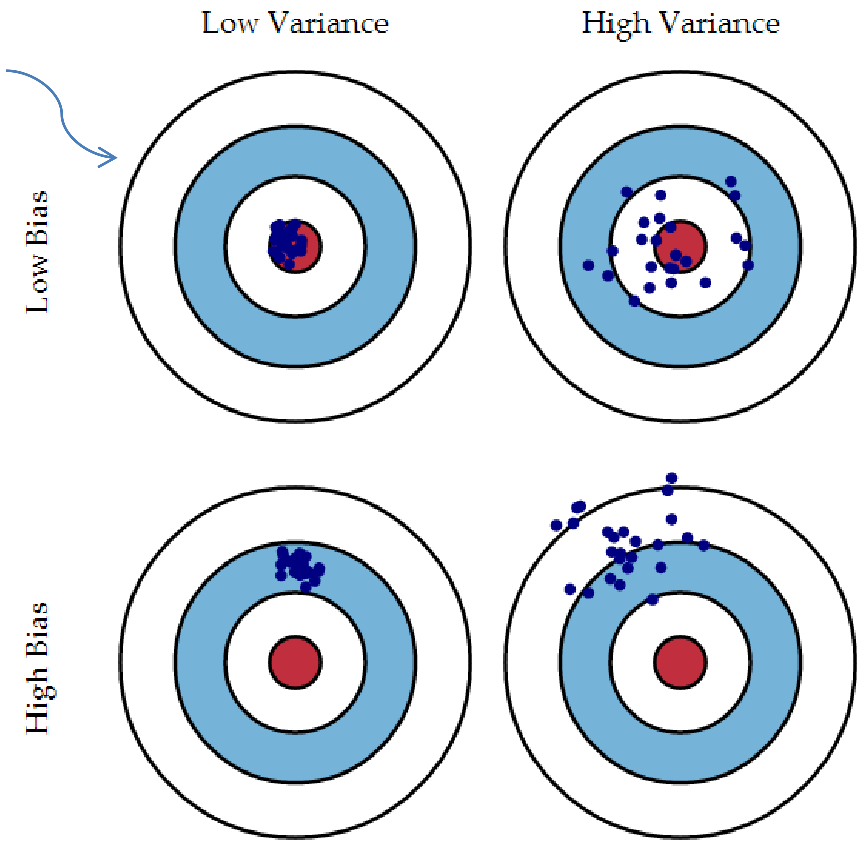
Bias/Variance (Graphical Presentation)



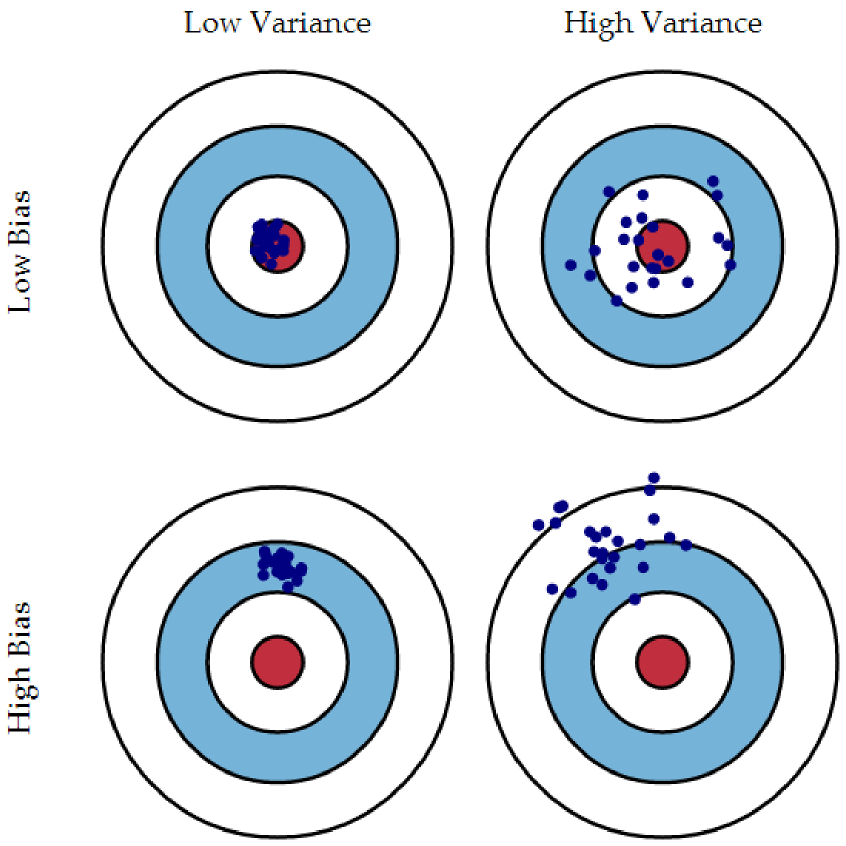
Bias/Variance (Graphical Presentation)

Simple Model, Simple Data

Never going to happen



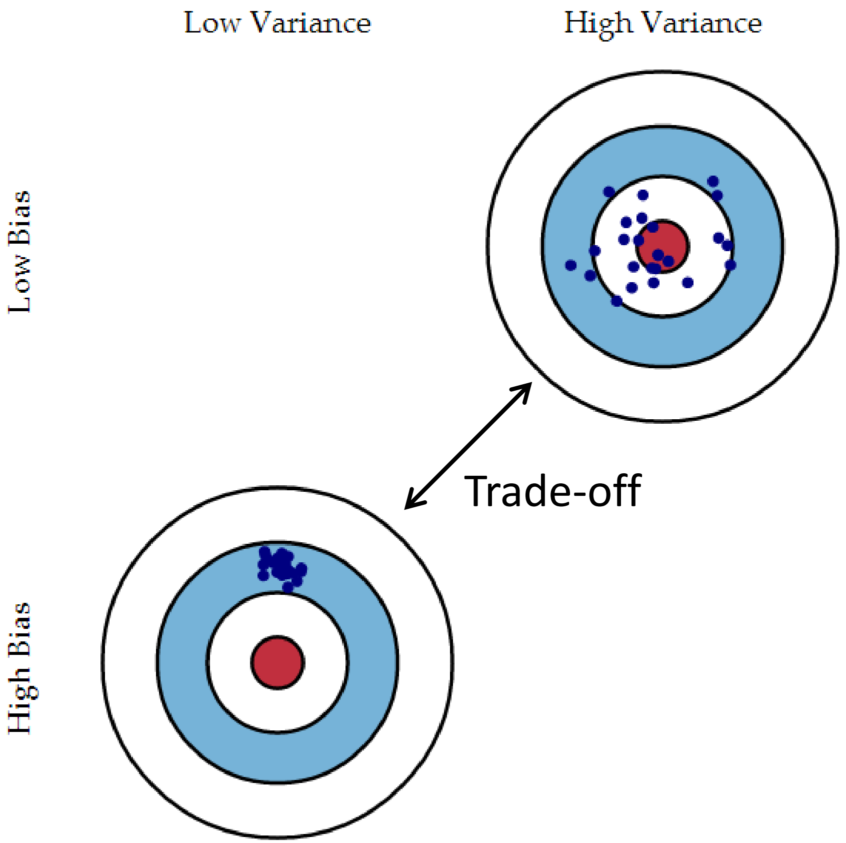
Bias/Variance (Graphical Presentation)



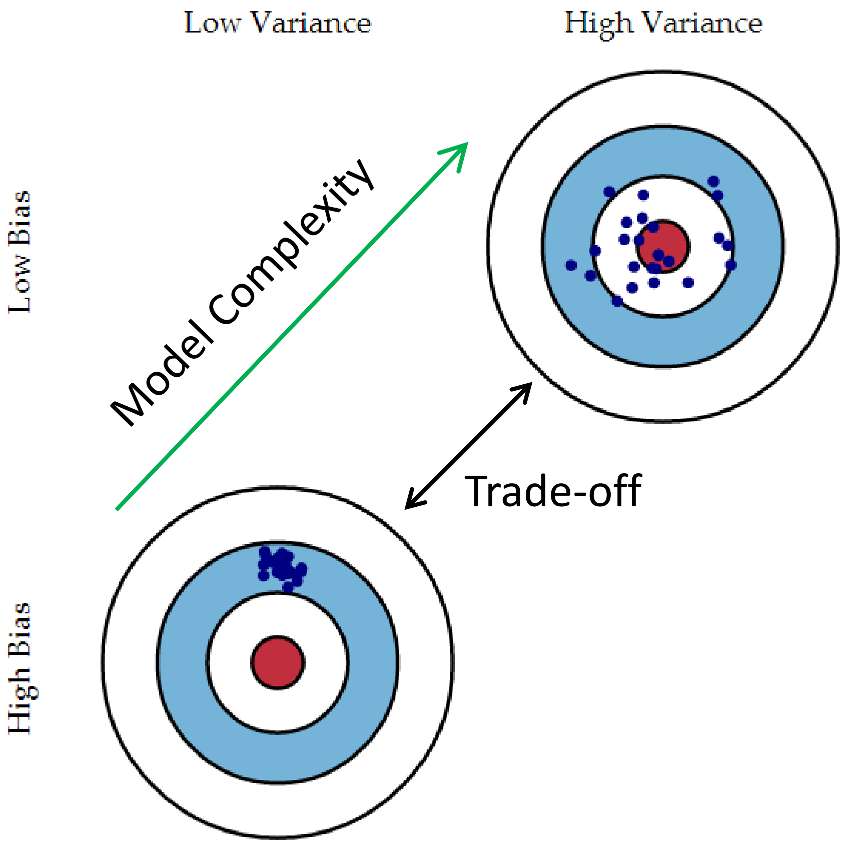
Complex Model,
Awful accuracy

Throw the method away and select another class of hypotheses

Bias/Variance (Graphical Presentation)



Bias/Variance (Graphical Presentation)



Bias/Variance (Mathematical Definition)

$$y = f(x) + \epsilon$$

$$Err(x) = E[\left(y - \hat{f}(x)\right)^2]$$

$$Err(x) = E\left[f(x)^2 - 2f(x)\hat{f}(x) + \hat{f}(x)^2\right]$$

$$Err(x) = f(x)^2 - 2f(x)E\left[\hat{f}(x)\right] + E[\hat{f}(x)^2]$$

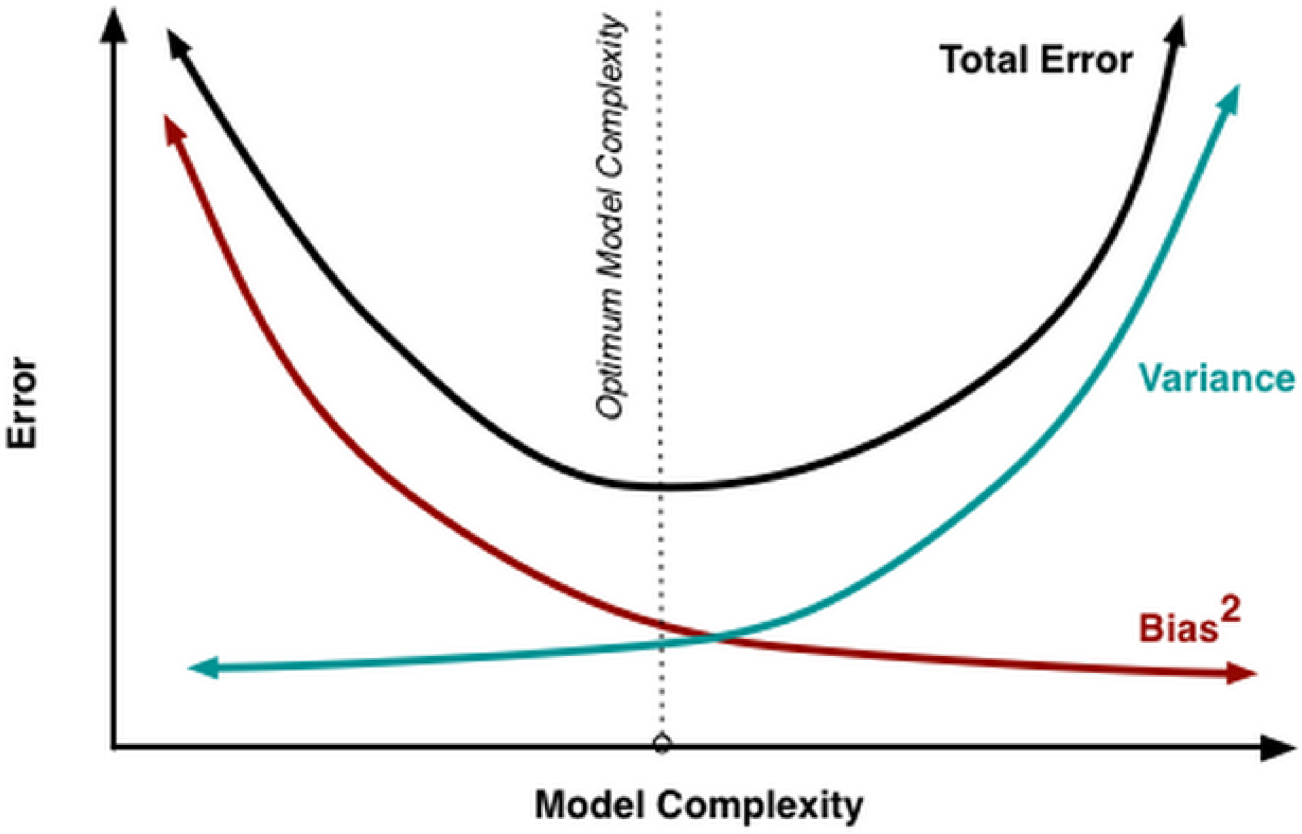
$$recall\ that: \quad Var(X) = E[X^2] - (E[X])^2$$

$$Err(x) = f(x)^2 - 2f(x)E\left[\hat{f}(x)\right] + E\left[\hat{f}(x) - E\left[\hat{f}(x)\right]\right]^2 + E\left[\hat{f}(x)\right]^2$$

$$Err(x) = \left(E\left[\hat{f}(x)\right] - f(x)\right)^2 + E\left[\hat{f}(x) - E\left[\hat{f}(x)\right]\right]^2$$

$$Err(x) = Bias^2 + Variance$$

Bias/Variance Trade-off



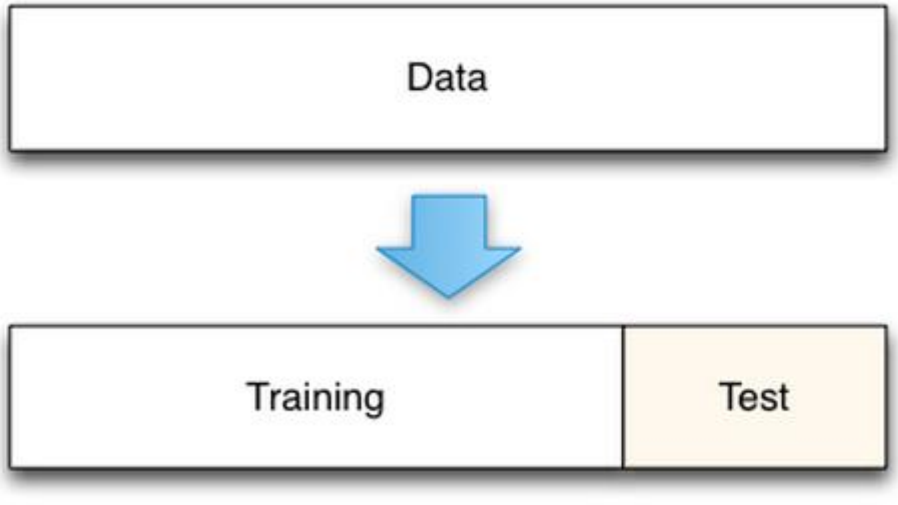
Bias/Variance (Examples)

- Linear classifiers High bias
  - Logistic regression
- K-Nearest Neighbor (KNN) Depends on K
- Decision trees Depends on splits
- Neural Networks High variance
- Ensemble methods
  - Bagging Good balance
    - Random forests
  - Boosting
    - Adaboost
- Support Vector Machines (SVMs) Good balance

Model Selection

Model Selection (Holdout set)

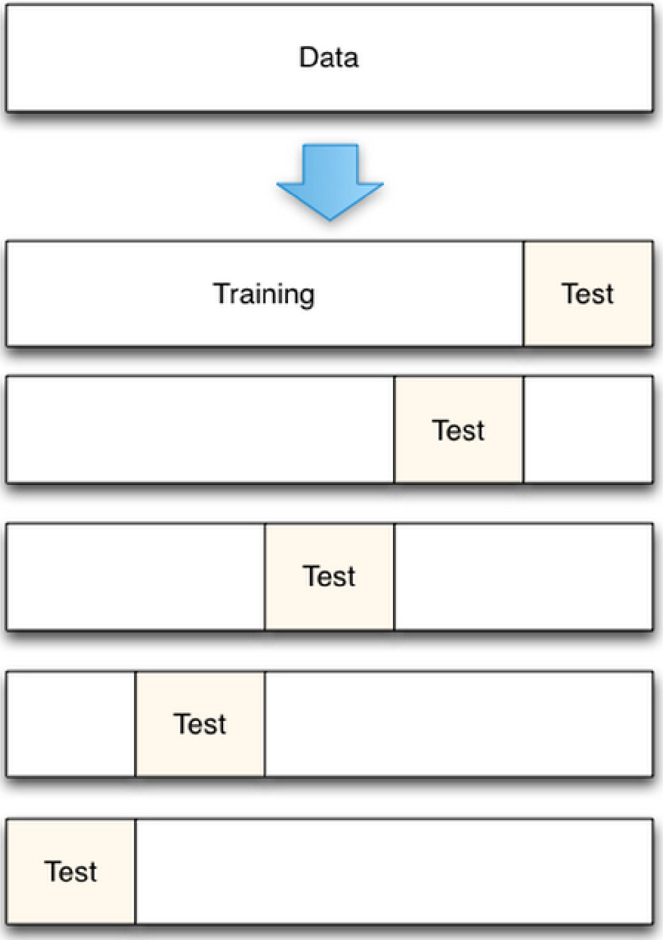
- Split the data into training and evaluation set



- Imperfect use of the data

Model Selection (Cross Validation)

- Each time $\frac{1}{K}$ of the data are used for evaluation
- The extreme ➔ Leave one out
- Greater use of the data
- Computationally expensive
- K=5 and K=10 are common choices



Model Selection (Information Theoretic)

- Akaike information criterion (AIC) ➔ Asymptotically correct

$$AIC = 2k - 2\ln(L)$$

- AICc (finite sample size)

$$AICc = 2k - 2\ln(L) + \frac{2k(k+1)}{n-k-1}$$

- Bayesian information criterion (BIC)

$$BIC = k \ln n - 2 \ln L$$

  - L is the maximized value of the likelihood function for the estimated model
  - It needs the likelihood value

Feature Selection

Feature Subset Selection

- Feature selection reduces the number of parameters in the model

$$F = \{f_1, f_2, f_3, \dots, f_n\}$$
$$A \subseteq F$$

- Number of possible feature subsets: 2^n
- Well-known examples
  - Forward selection: begin with an empty feature set and greedily add feature that decreases cross validation error most
  - Backward elimination: begin with the entire feature set and greedily eliminate feature that decreases cross validation error most

- Computationally intensive ➔ for each evaluation of the feature set you must train your learning algorithm

Feature Selection (Information theoretic)

- Mutual Information

$$I(X;Y) = \sum_{y\in Y}\sum_{x\in X} p(x,y)\log\left(\frac{p(x,y)}{p(x)p(y)}\right)$$

- Kullback-Leibler divergence ➔ Information loss

$$D_{KL}(Y||X) = \sum_{i} \ln\left(\frac{y(i)}{x(i)}\right) y(i)$$

$$I(X;Y) = D_{KL}(p(x,y)\big|\big|p(x)p(y)\big)$$

Feature Transformation (Dimensionality Reduction)

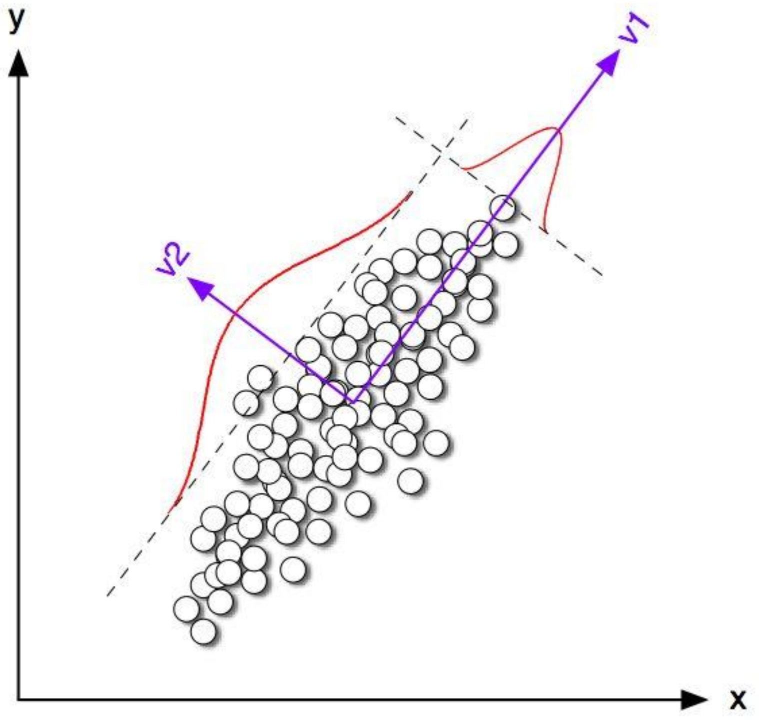
- Principal Component Analysis (PCA)
  - Maximum variance
  - Minimum error

- Linear Discriminant Analysis (LDA)
  - Fisher's linear discriminant

- Manifold learning
  - Laplacian eigenmaps
  - ISOMAP

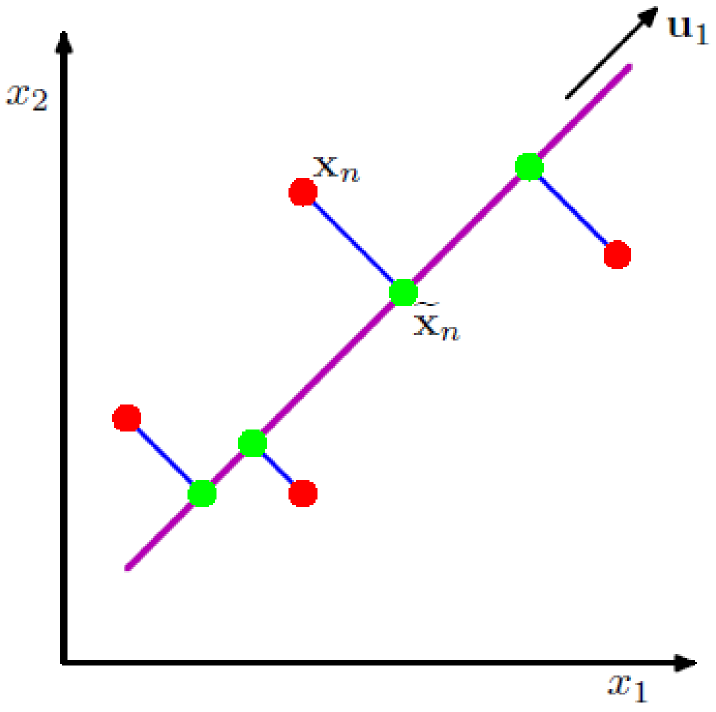
Feature Transformation (Dimensionality Reduction)

- Principal Component Analysis (PCA)
  - **Maximum variance**
  - Minimum error
- Linear Discriminant Analysis (LDA)
  - Fisher's linear discriminant
- Manifold learning
  - Laplacian eigenmaps
  - ISOMAP



Feature Transformation (Dimensionality Reduction)

- Principal Component Analysis (PCA)
  - Maximum variance
  - **Minimum error**
- Linear Discriminant Analysis (LDA)
  - Fisher's linear discriminant
- Manifold learning
  - Laplacian eigenmaps
  - ISOMAP

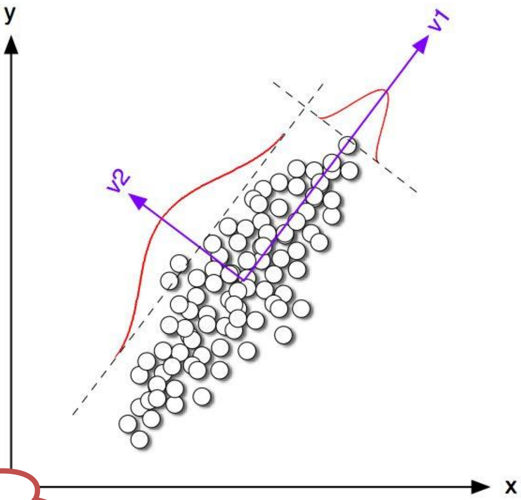


Feature Transformation (Dimensionality Reduction)

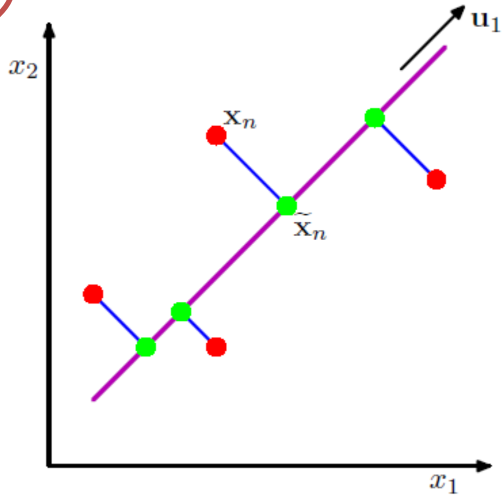
- Principal Component Analysis (PCA)
  - Maximum variance
  - Minimum error

- Linear Discriminant Analysis (LDA)
  - Fisher's linear discriminant

- Manifold learning
  - Laplacian eigenmaps
  - ISOMAP

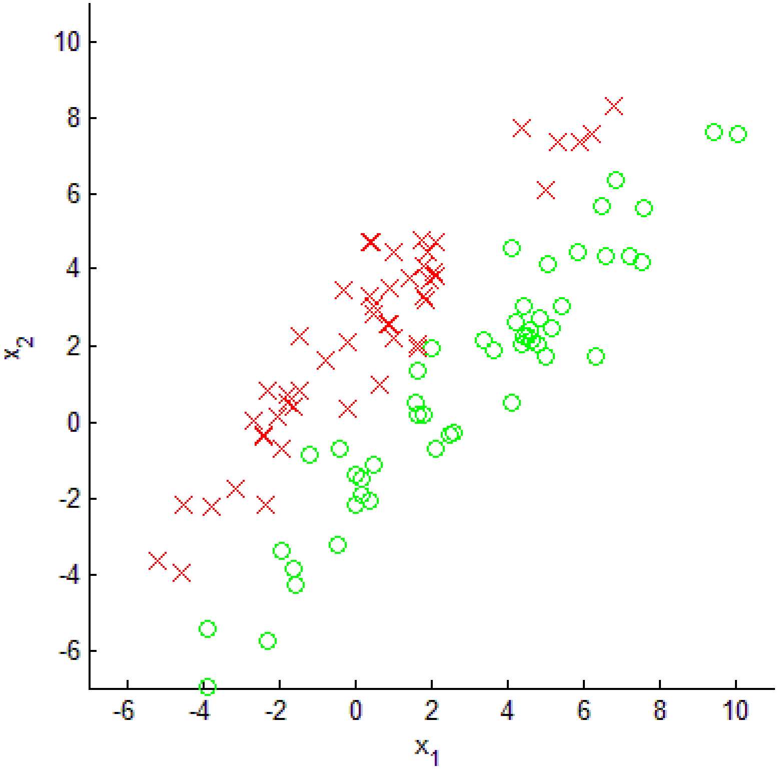


What is the PCA's problem?



Feature Transformation (Dimensionality Reduction)

- Principal Component Analysis (PCA)
  - Maximum variance
  - Minimum error

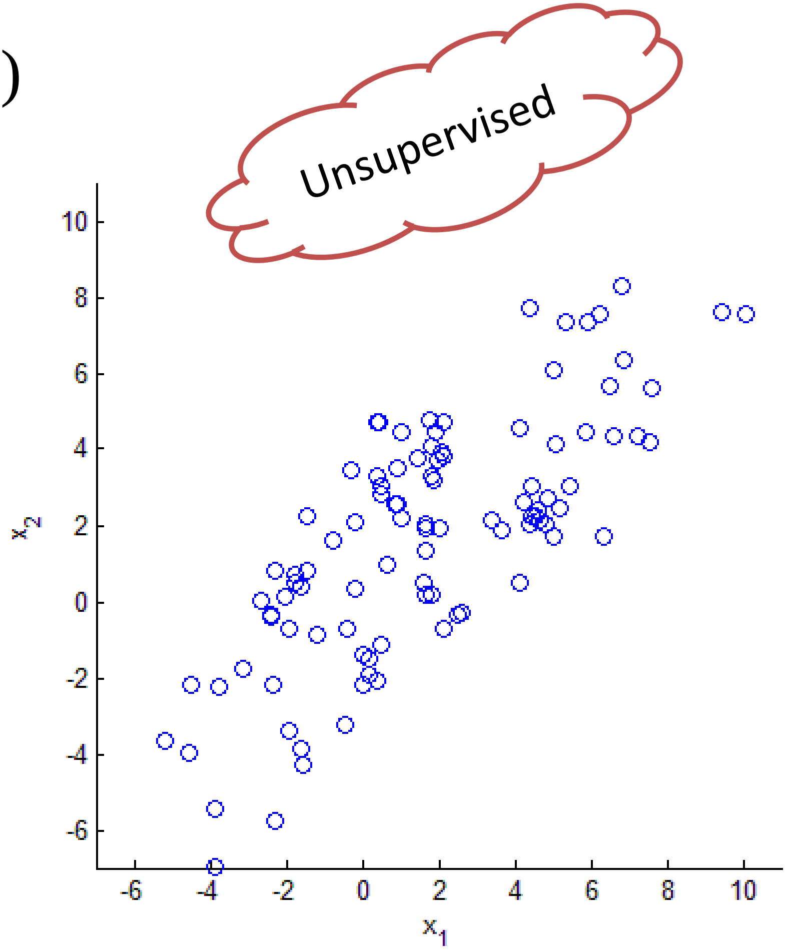


- Linear Discriminant Analysis (LDA)
  - Fisher's linear discriminant

- Manifold learning
  - Laplacian eigenmaps
  - ISOMAP

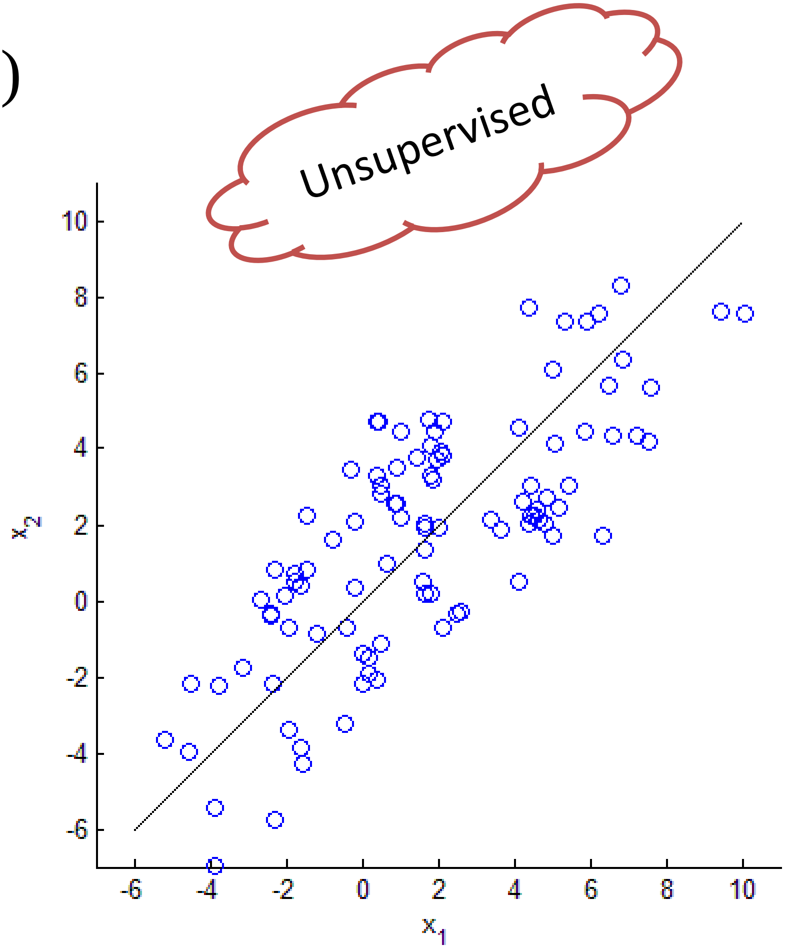
Feature Transformation (Dimensionality Reduction)

- Principal Component Analysis (PCA)
  - Maximum variance
  - Minimum error
- Linear Discriminant Analysis (LDA)
  - Fisher's linear discriminant
- Manifold learning
  - Laplacian eigenmaps
  - ISOMAP


Unsupervised

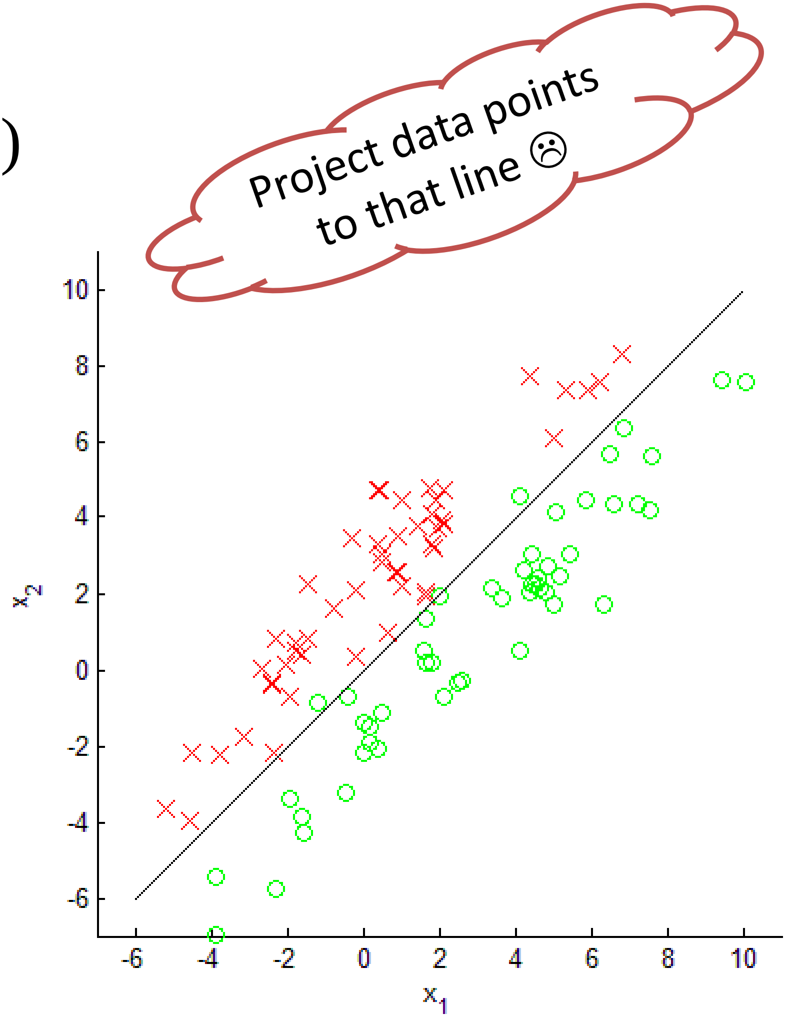
Feature Transformation (Dimensionality Reduction)

- Principal Component Analysis (PCA)
  - Maximum variance
  - Minimum error
- Linear Discriminant Analysis (LDA)
  - Fisher's linear discriminant
- Manifold learning
  - Laplacian eigenmaps
  - ISOMAP


Unsupervised

Feature Transformation (Dimensionality Reduction)

- Principal Component Analysis (PCA)
  - Maximum variance
  - Minimum error
- Linear Discriminant Analysis (LDA)
  - Fisher's linear discriminant
- Manifold learning
  - Laplacian eigenmaps
  - ISOMAP



Project data points to that line ☹

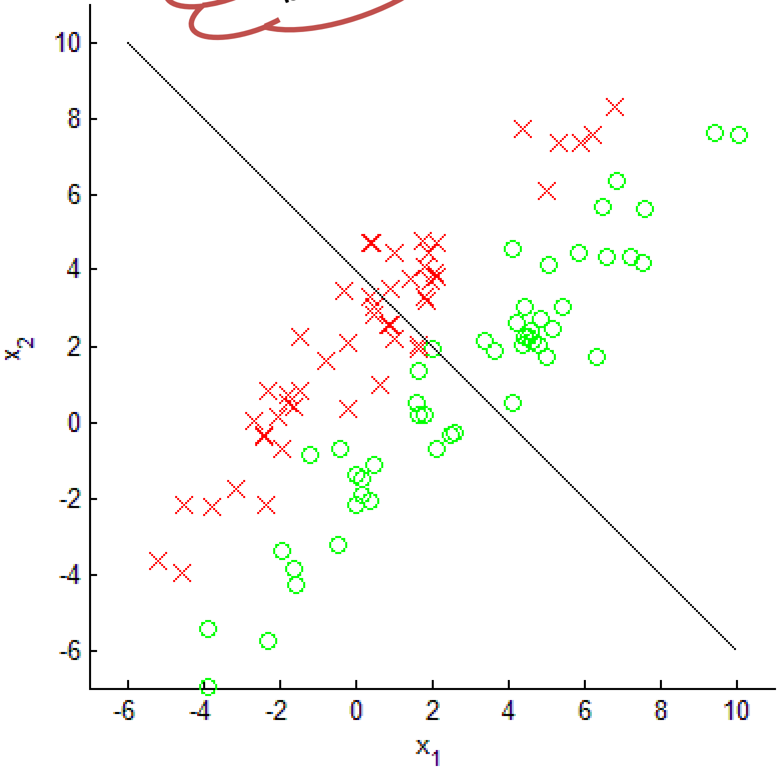
Feature Transformation (Dimensionality Reduction)

- Principal Component Analysis (PCA)
  - Maximum variance
  - Minimum error

- Linear Discriminant Analysis (LDA)
  - Fisher's linear discriminant

- Manifold learning
  - Laplacian eigenmaps
  - ISOMAP

This line has better separability

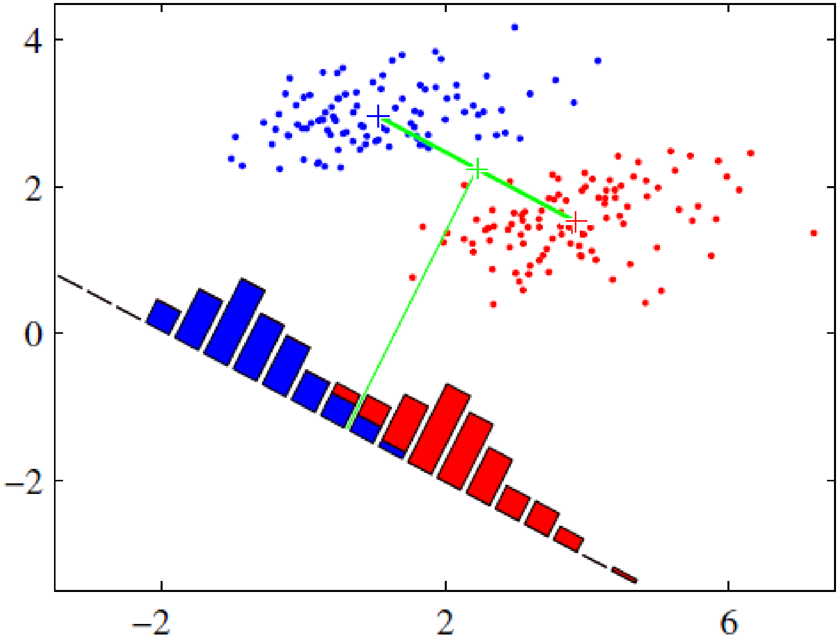


Feature Transformation (Dimensionality Reduction)

- Principal Component Analysis (PCA)
  - Maximum variance
  - Minimum error

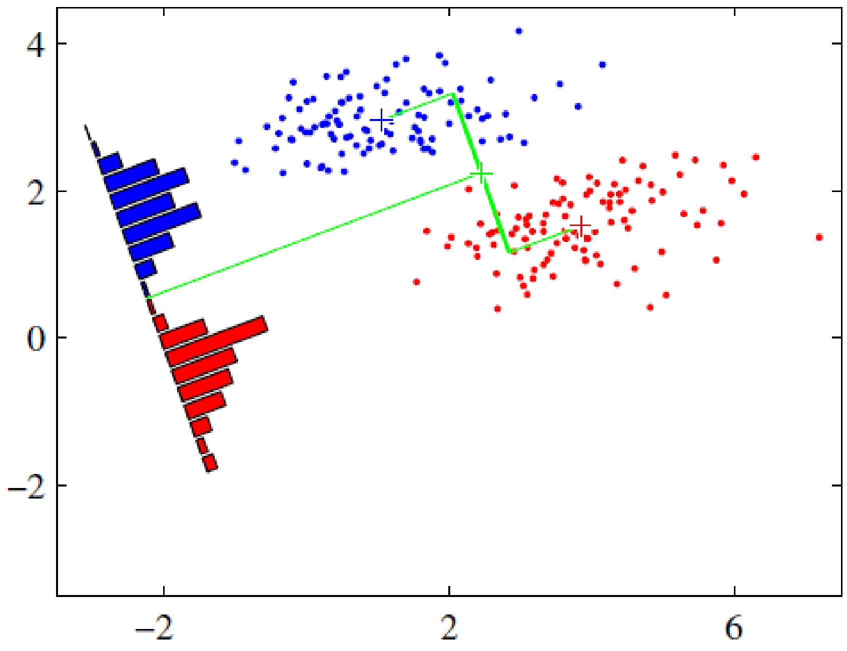
- Linear Discriminant Analysis (LDA)
  - Fisher's linear discriminant

- Manifold learning
  - Laplacian eigenmaps
  - ISOMAP



Feature Transformation (Dimensionality Reduction)

- Principal Component Analysis (PCA)
  - Maximum variance
  - Minimum error
- Linear Discriminant Analysis (LDA)
  - Fisher's linear discriminant
- Manifold learning
  - Laplacian eigenmaps
  - ISOMAP

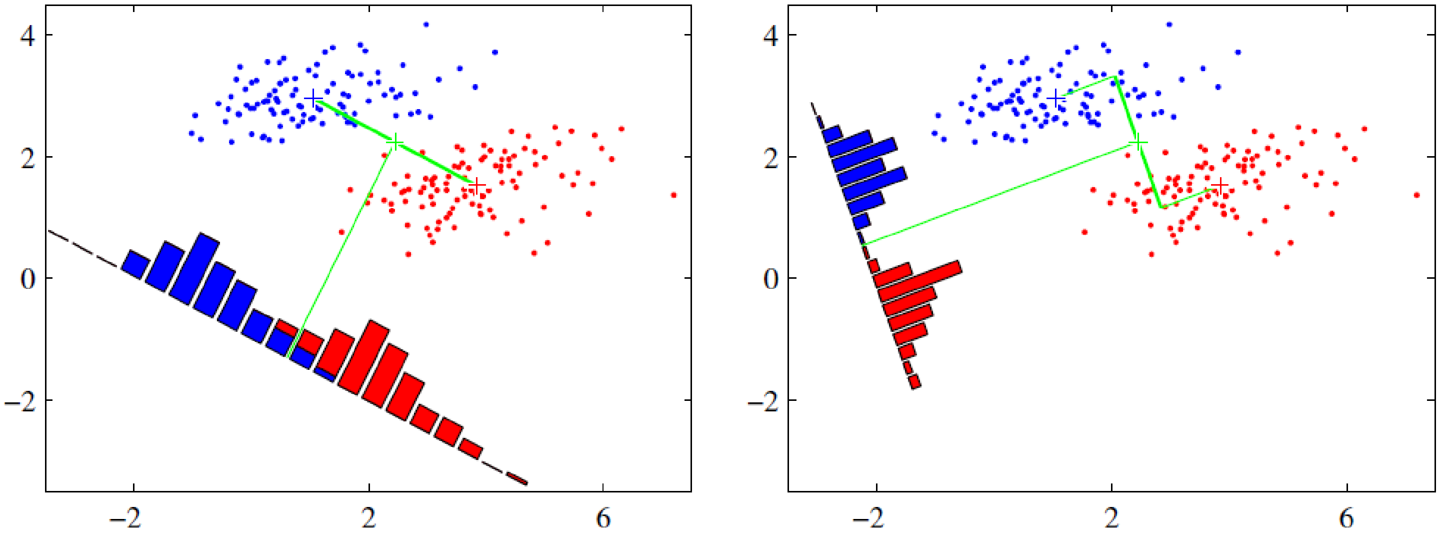


Feature Transformation (Dimensionality Reduction)

- Principal Component Analysis (PCA)
  - Maximum variance
  - Minimum error

- Linear Discriminant Analysis (LDA)
  - Fisher's linear discriminant

- Manifol
  - Lapla
  - ISOM

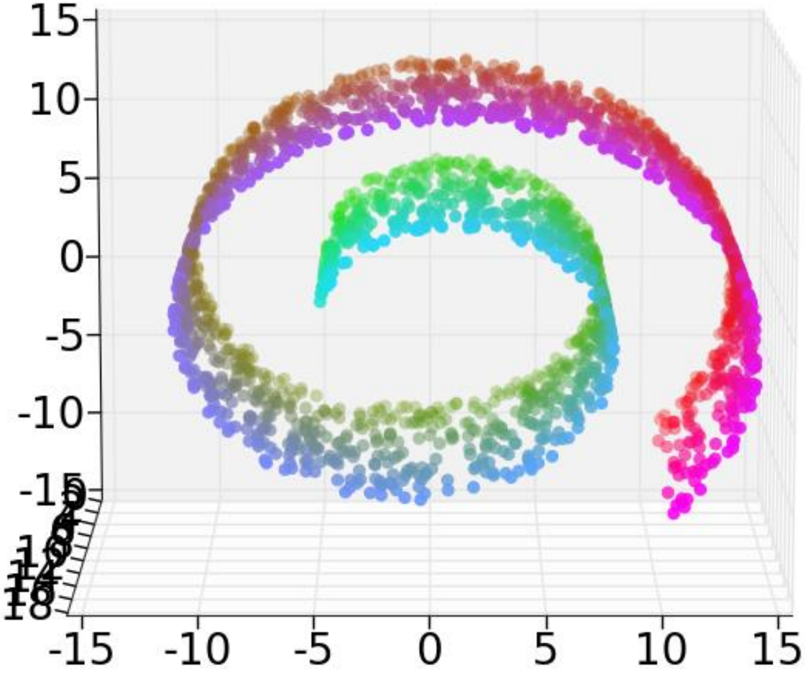


Feature Transformation (Dimensionality Reduction)

- Principal Component Analysis (PCA)
  - Maximum variance
  - Minimum error

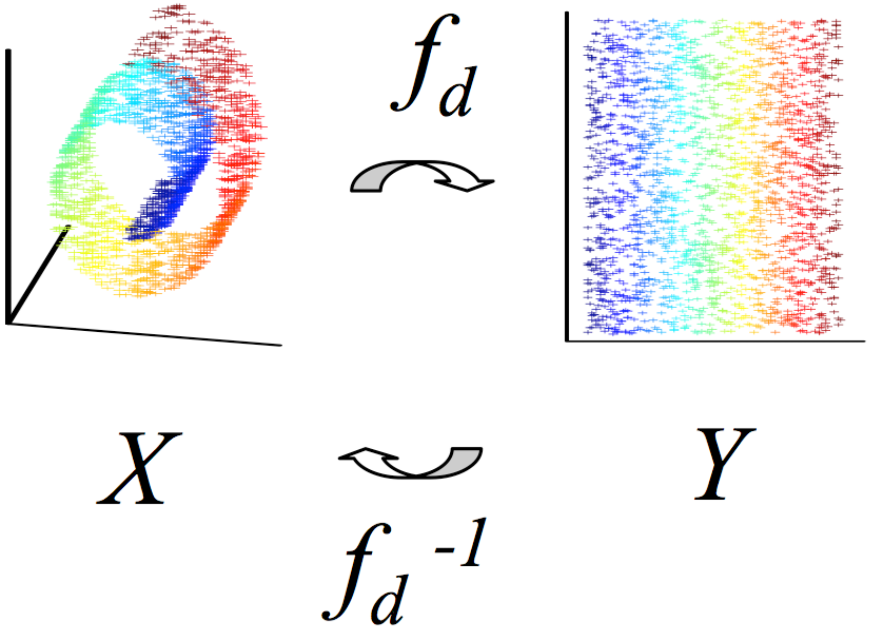
- Linear Discriminant Analysis (LDA)
  - Fisher's linear discriminant

- Manifold learning
  - Laplacian eigenmaps
  - ISOMAP



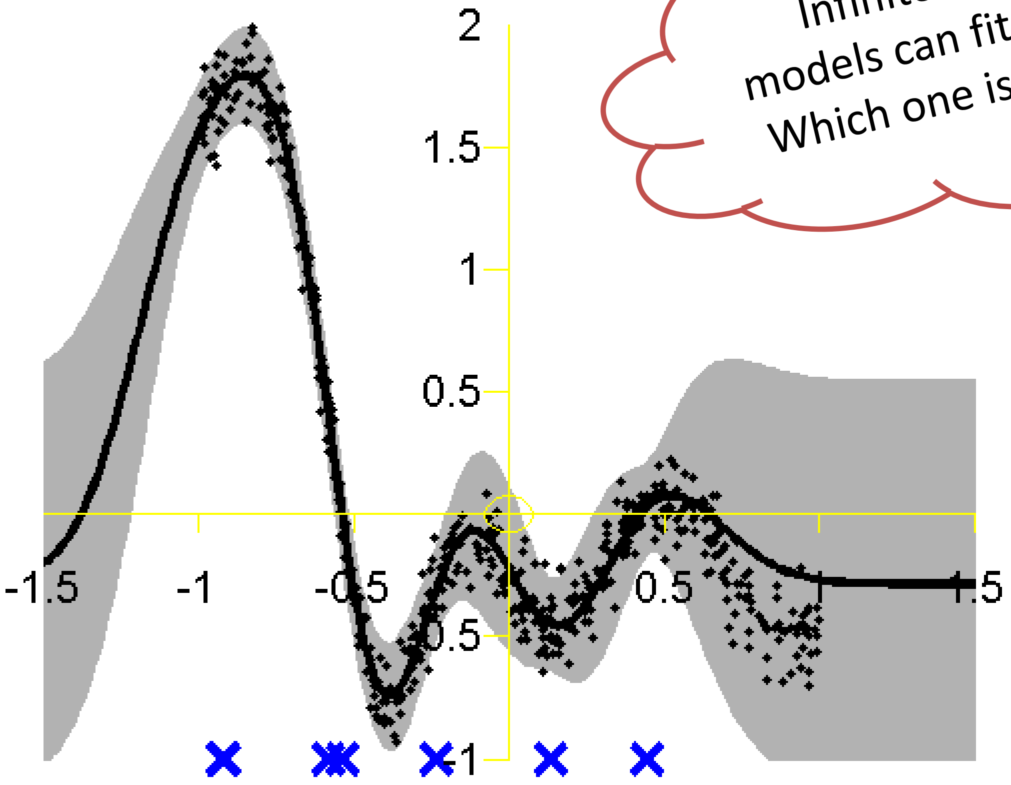
Feature Transformation (Dimensionality Reduction)

- Principal Component Analysis (PCA)
  - Maximum variance
  - Minimum error
- Linear Discriminant Analysis (LDA)
  - Fisher's linear discriminant
- Manifold learning
  - Laplacian eigenmaps
  - ISOMAP



Regularization

Regularization (Concept)



Infinite number of models can fit to my data
Which one is preferred?

Regularization ($L^p - spaces$)

- $L^P - norm$

$$x \in \mathbb{R}^n$$

$$\|x\|_p = (|x_1|^p + |x_2|^p + \cdots + |x_n|^p)^{\frac{1}{p}}$$

- $L^1 - norm$

$$\|x\|_1 = |x_1| + |x_2| + \cdots + |x_n|$$

- $L^2 - norm\ (Euclidean\ norm)$

$$\|x\|_2 = (x_1^2 + x_2^2 + \cdots + x_n^2)^{\frac{1}{2}} = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}$$

- $\dots$
- $L^\infty - norm$

$$\|x\|_\infty = (|x_1|^\infty + |x_2|^\infty + \cdots + |x_n|^\infty)^{\frac{1}{\infty}} = \sqrt[\infty]{|x_1|^\infty + |x_2|^\infty + \cdots + |x_n|^\infty}$$

$$\|x\|_\infty = \max\{|x_1|, |x_2|, \dots, |x_n|\}$$

Regularization (metric spaces)

- $L^1 - space$

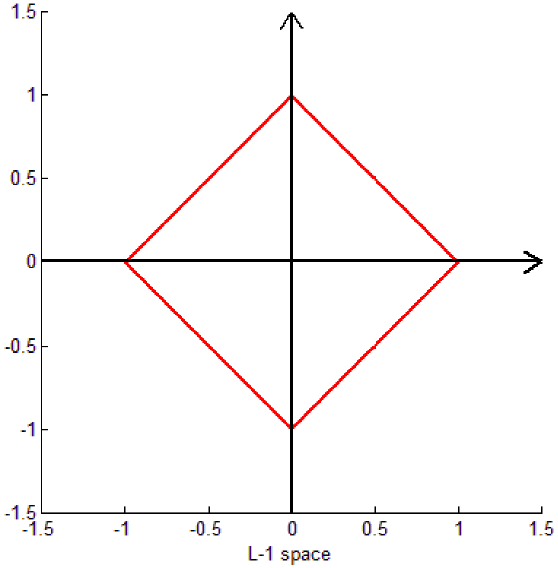
$$\|x\|_1 = |x_1| + |x_2| + \cdots + |x_n|$$

- $L^2 - space\ (Euclidean\ space)$

$$\|x\|_2 = (x_1^2 + x_2^2 + \cdots + x_n^2)^{\frac{1}{2}} = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}$$

- $L^\infty - space$

$$\|x\|_\infty = \max\{|x_1|, |x_2|, \dots, |x_n|\}$$



Regularization (metric spaces)

- $L^1 - space$

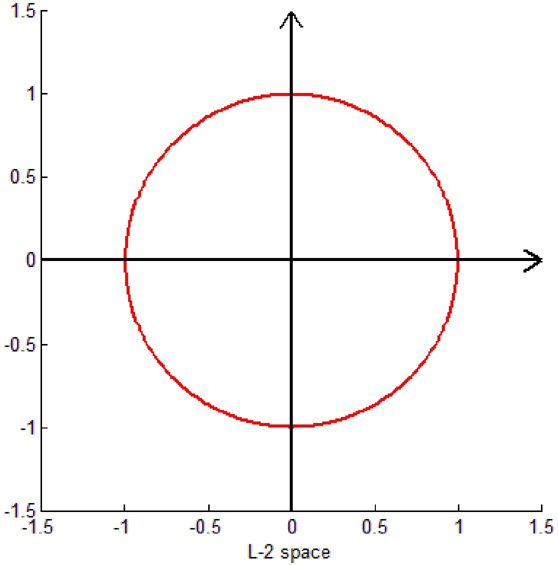
$$\|x\|_1 = |x_1| + |x_2| + \cdots + |x_n|$$

- $L^2 - space\ (Euclidean\ space)$

$$\|x\|_2 = (x_1^2 + x_2^2 + \cdots + x_n^2)^{\frac{1}{2}} = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}$$

- $L^\infty - space$

$$\|x\|_\infty = \max\{|x_1|, |x_2|, \dots, |x_n|\}$$



Dalhousie University
Behrouz H. Soleimani

Regularization (metric spaces)

- $L^1 - space$

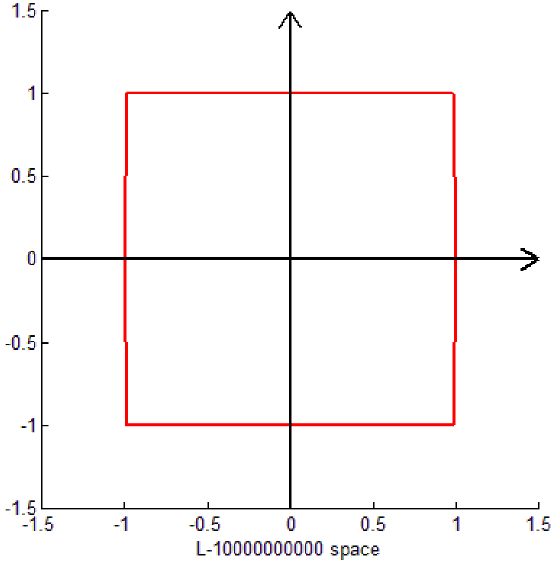
$$\|x\|_1 = |x_1| + |x_2| + \cdots + |x_n|$$

- $L^2 - space\ (Euclidean\ space)$

$$\|x\|_2 = (x_1^2 + x_2^2 + \cdots + x_n^2)^{\frac{1}{2}} = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}$$

- $L^\infty - space$

$$\|x\|_\infty = \max\{|x_1|, |x_2|, \dots, |x_n|\}$$

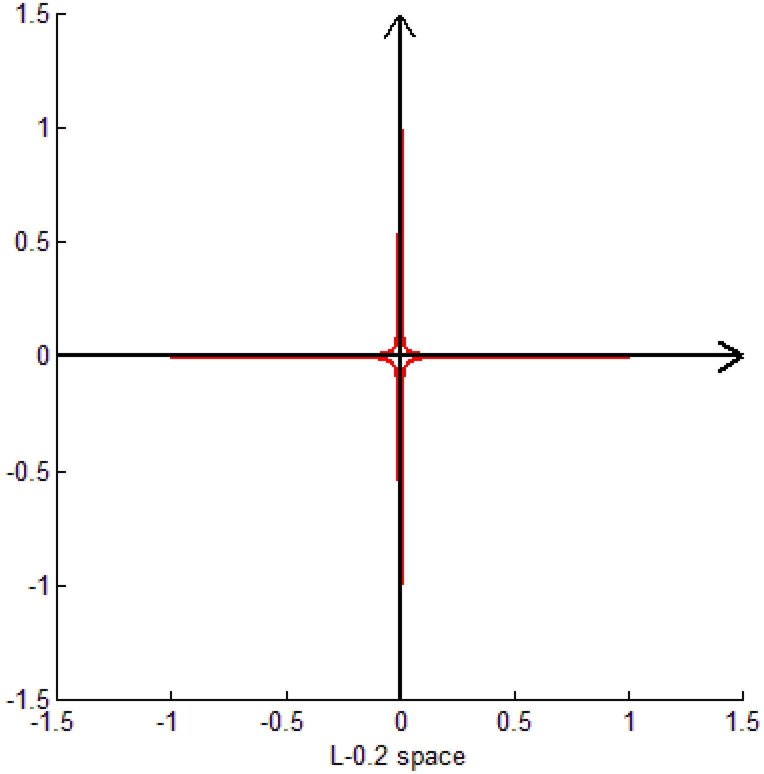


Regularization (metric spaces)

Animation

☺

P = 0.1 , 0.2 , … , 500



Regularization

- Objective function (Loss function)

$$\mathcal{J}(\boldsymbol{W}) = \sum_{i=1}^{n} \boldsymbol{\mathcal{L}}\big(y_i, f(\boldsymbol{x}_i, \boldsymbol{W})\big)$$

$$\boldsymbol{\mathcal{L}}(y, \hat{y}) = (y - \hat{y})^2$$

$$\boldsymbol{\mathcal{L}}(y, \hat{y}) = I\{y \neq \hat{y}\}$$

$$\mathcal{J}^*(\boldsymbol{W}) = \min_{\boldsymbol{W}} \mathcal{J}(\boldsymbol{W}) = \min_{\boldsymbol{W}} \sum_{i=1}^{n} \boldsymbol{\mathcal{L}}\big(y_i, f(\boldsymbol{x}_i, \boldsymbol{W})\big)$$

$$\boldsymbol{W}^* = \arg\min_{\boldsymbol{W}} \sum_{i=1}^{n} \boldsymbol{\mathcal{L}}\big(y_i, f(\boldsymbol{x}_i, \boldsymbol{W})\big)$$

Why we always use squared error?

$|y - \hat{y}|$ **instead of** $(y - \hat{y})^2$

Regularization

$$\mathcal{J}(\boldsymbol{W}) = \sum_{i=1}^{n} \boldsymbol{\mathcal{L}}\big(y_i, f(\boldsymbol{x}_i, \boldsymbol{W})\big)$$

$$\widehat{y_i} = \boldsymbol{W}^T \boldsymbol{x}_i$$

$$\mathcal{J}(\boldsymbol{W}) = \sum_{i=1}^{n} (y_i - \boldsymbol{W}^T \boldsymbol{x}_i)^2$$

$$\mathcal{J}(\boldsymbol{W}) = (\boldsymbol{\mathcal{Y}} - \boldsymbol{\mathcal{X}}\boldsymbol{W})^{\boldsymbol{T}}(\boldsymbol{\mathcal{Y}} - \boldsymbol{\mathcal{X}}\boldsymbol{W}) = \|\boldsymbol{\mathcal{Y}} - \boldsymbol{\mathcal{X}}\boldsymbol{W}\|_2^2$$

$$\boldsymbol{\mathcal{Y}}_{n\times 1}, \boldsymbol{\mathcal{X}}_{n\times d}, \boldsymbol{W}_{d\times 1}$$

Regularization (Mathematical Def.)

$$\mathcal{J}(\boldsymbol{W}) = \sum_{i=1}^{n} \boldsymbol{\mathcal{L}}\big(y_i, f(\boldsymbol{x}_i, \boldsymbol{W})\big) + \lambda\|\boldsymbol{W}\|$$

- L_1 regularization (Lasso)

$$\mathcal{J}(\boldsymbol{W}) = \sum_{i=1}^{n} \boldsymbol{\mathcal{L}}\big(y_i, f(\boldsymbol{x}_i, \boldsymbol{W})\big) + \lambda\|\boldsymbol{W}\|_1$$

- L_2 regularization (Ridge regression)

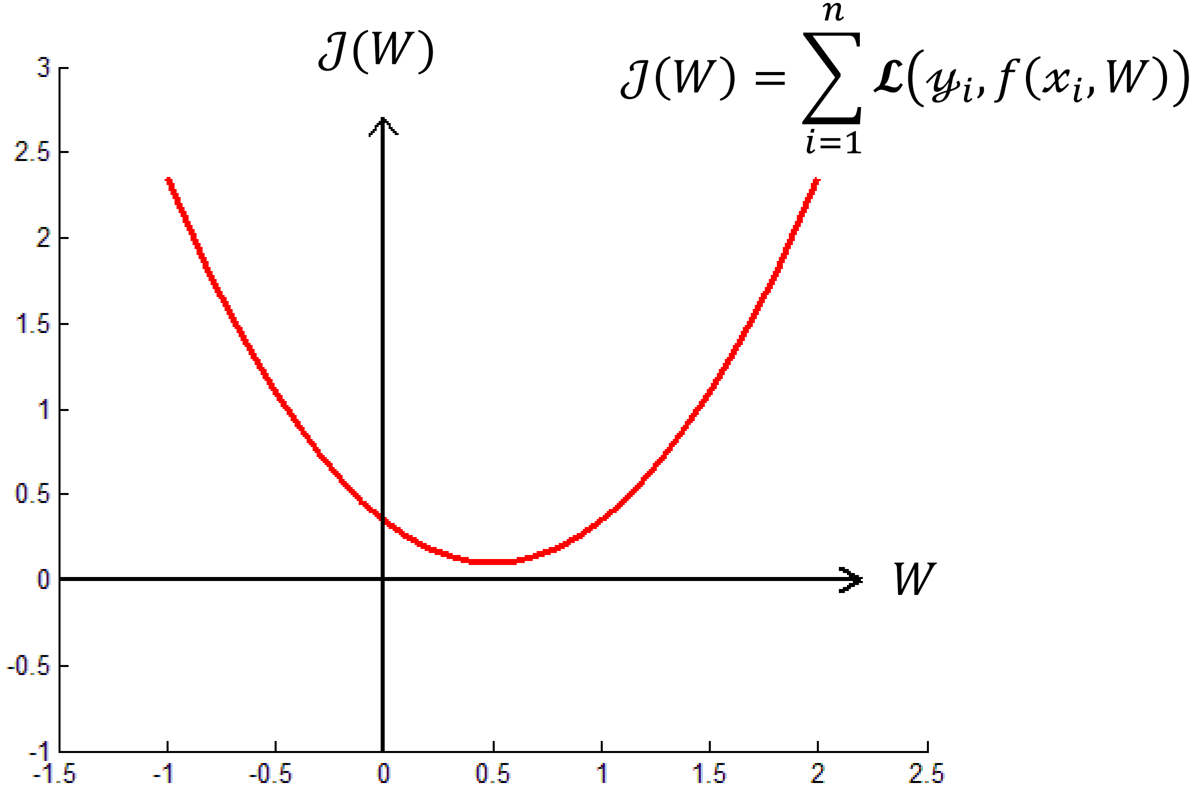
$$\mathcal{J}(\boldsymbol{W}) = \sum_{i=1}^{n} \boldsymbol{\mathcal{L}}\big(y_i, f(\boldsymbol{x}_i, \boldsymbol{W})\big) + \lambda\|\boldsymbol{W}\|_2$$

$$\|\boldsymbol{W}\|_2^2 = \boldsymbol{W}^{\mathrm{T}}\boldsymbol{W}$$

- Elastic net regularization

$$\mathcal{J}(\boldsymbol{W}) = \sum_{i=1}^{n} \boldsymbol{\mathcal{L}}\big(y_i, f(\boldsymbol{x}_i, \boldsymbol{W})\big) + \lambda_2\|\boldsymbol{W}\|_2 + \lambda_1\|\boldsymbol{W}\|_1$$

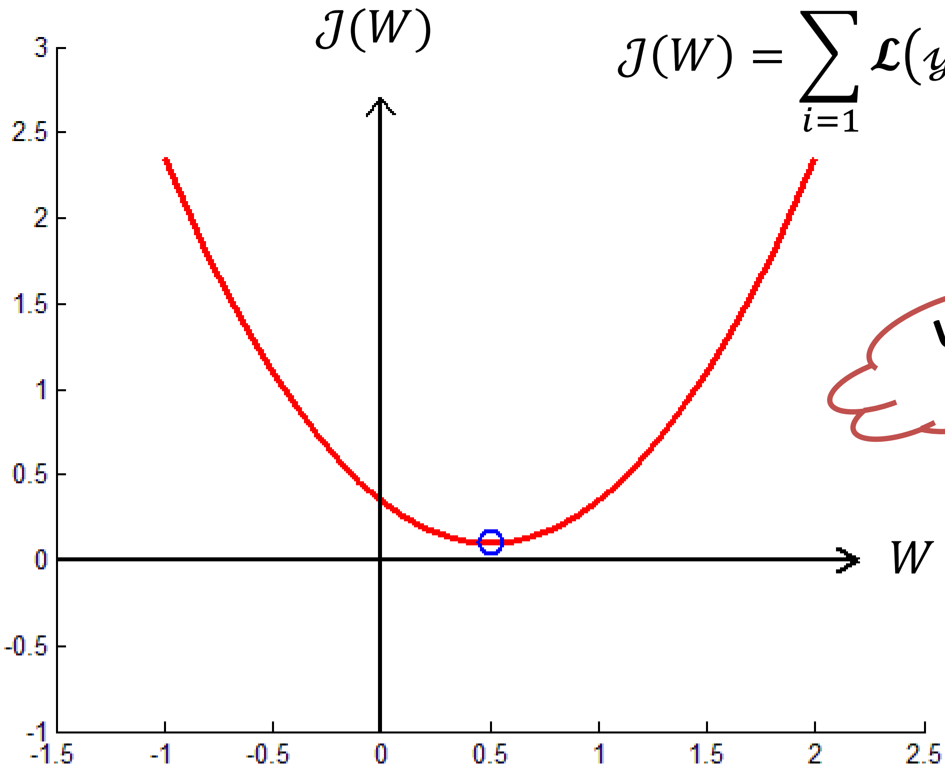
Regularization (Graphical Representation)



$$\mathcal{J}(W) = \sum_{i=1}^{n} \mathcal{L}\big(y_i, f(x_i, W)\big)$$

Regularization (Graphical Representation)

$$\mathcal{J}(W) = \sum_{i=1}^{n} \boldsymbol{\mathcal{L}}\big(y_i, f(x_i, W)\big)$$

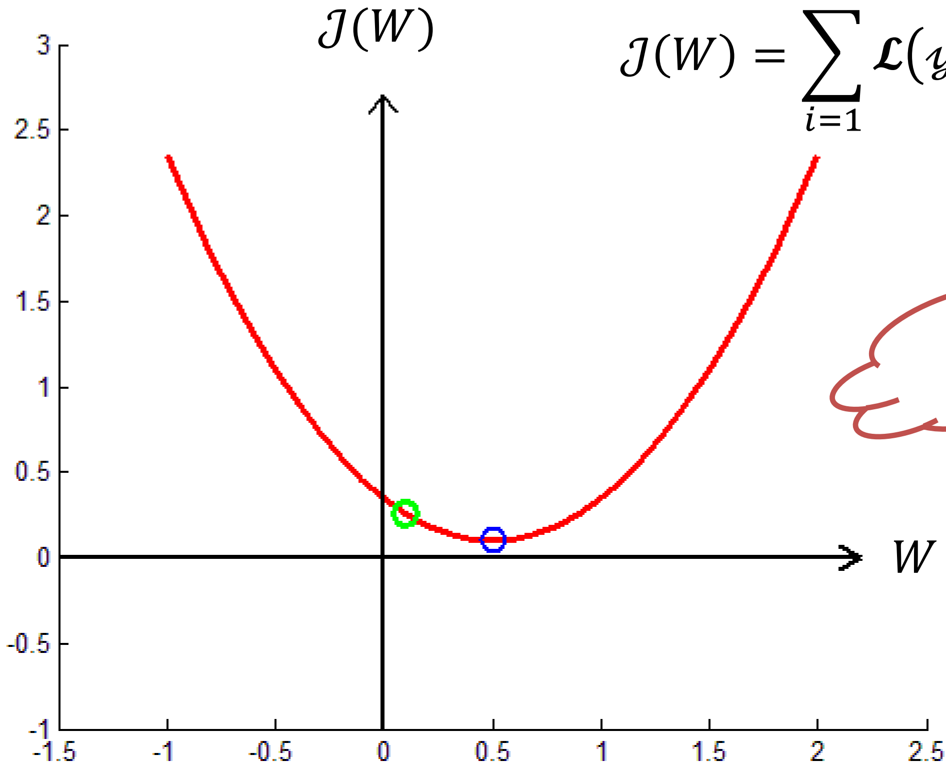


Unconstrained optimization

$$\mathcal{J}(\boldsymbol{W}) = \sum_{i=1}^{n} \boldsymbol{\mathcal{L}}\big(y_i, f(\boldsymbol{x}_i, \boldsymbol{W})\big)$$

Regularization (Graphical Representation)

$$\mathcal{J}(W) = \sum_{i=1}^{n} \boldsymbol{\mathcal{L}}\big(y_i, f(x_i, W)\big)$$

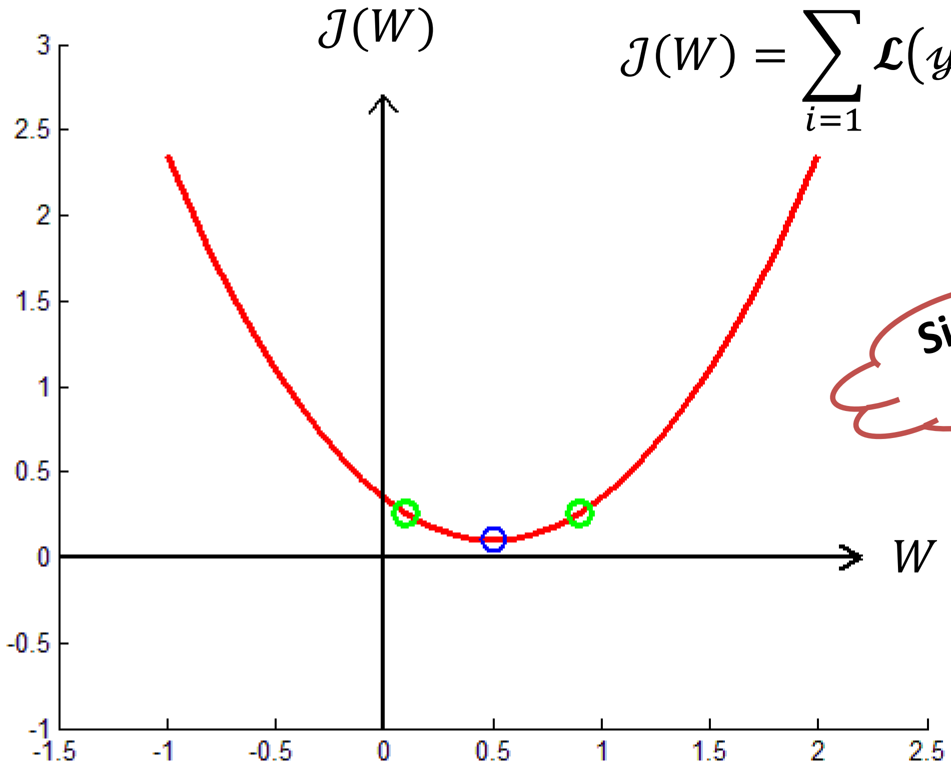


Regularized optimization

$$\mathcal{J}(\boldsymbol{W}) = \sum_{i=1}^{n} \boldsymbol{\mathcal{L}}\big(y_i, f(\boldsymbol{x}_i, \boldsymbol{W})\big) + \lambda \|\boldsymbol{W}\|_2$$

Regularization (Graphical Representation)

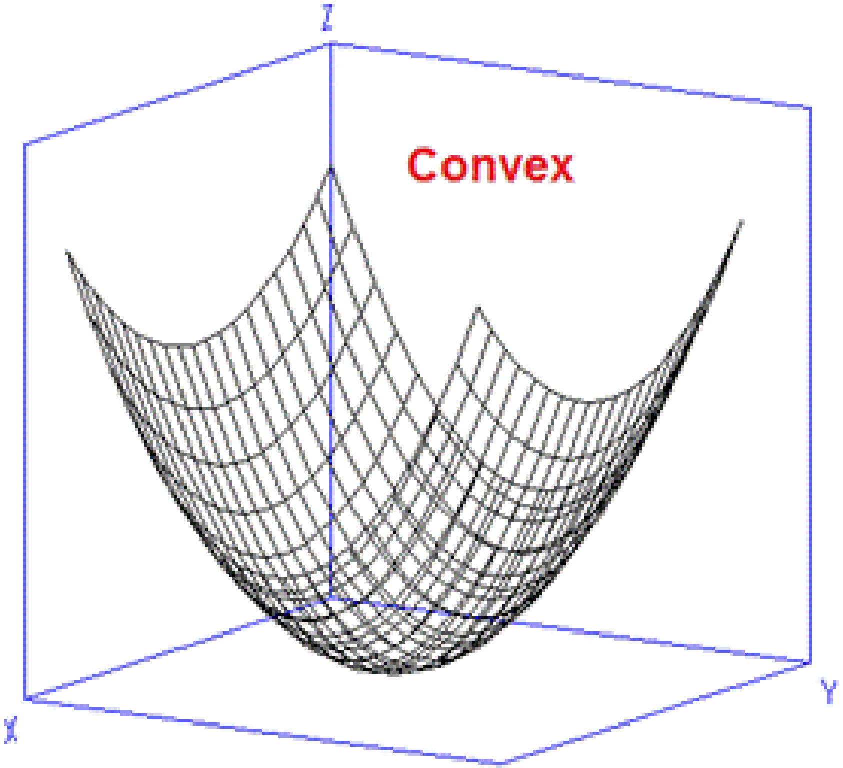
$$\mathcal{J}(W) = \sum_{i=1}^{n} \boldsymbol{\mathcal{L}}\big(y_i, f(x_i, W)\big)$$

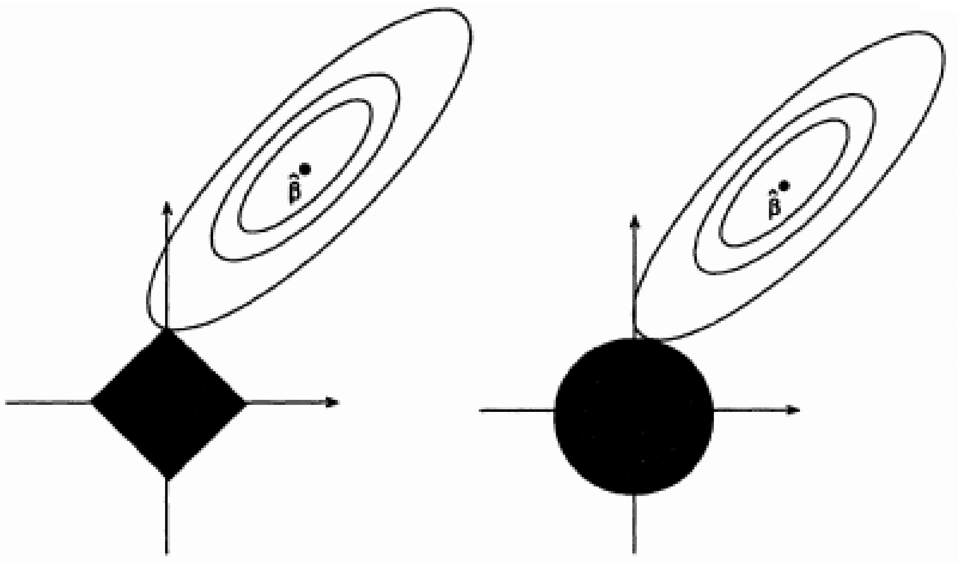


Simpler model is preferred

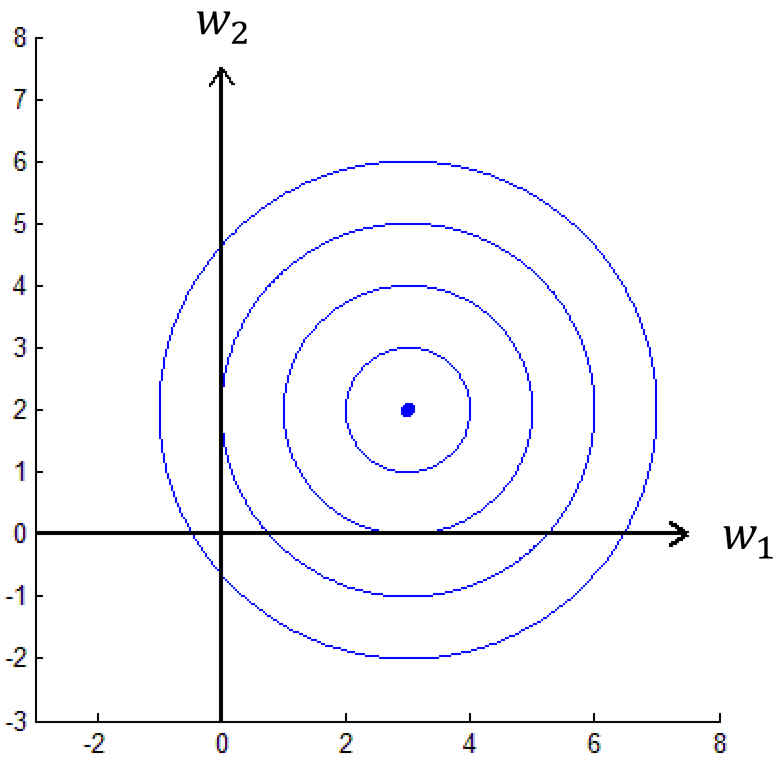
$$\mathcal{J}(\boldsymbol{W}) = \sum_{i=1}^{n} \boldsymbol{\mathcal{L}}\big(y_i, f(\boldsymbol{x}_i, \boldsymbol{W})\big) + \lambda \|\boldsymbol{W}\|_2$$

Regularization (Graphical Representation)

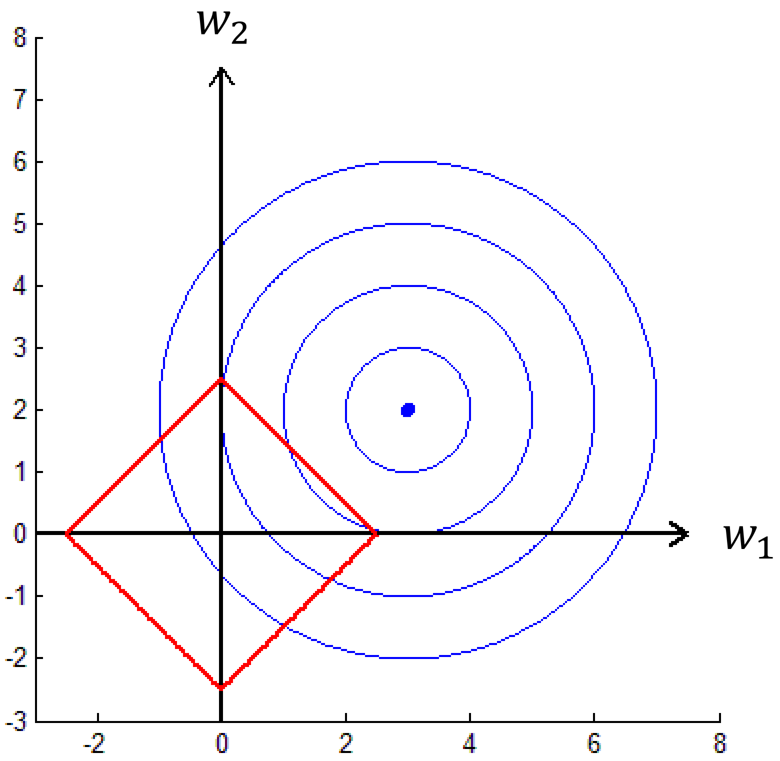




Regularization (Graphical Representation)

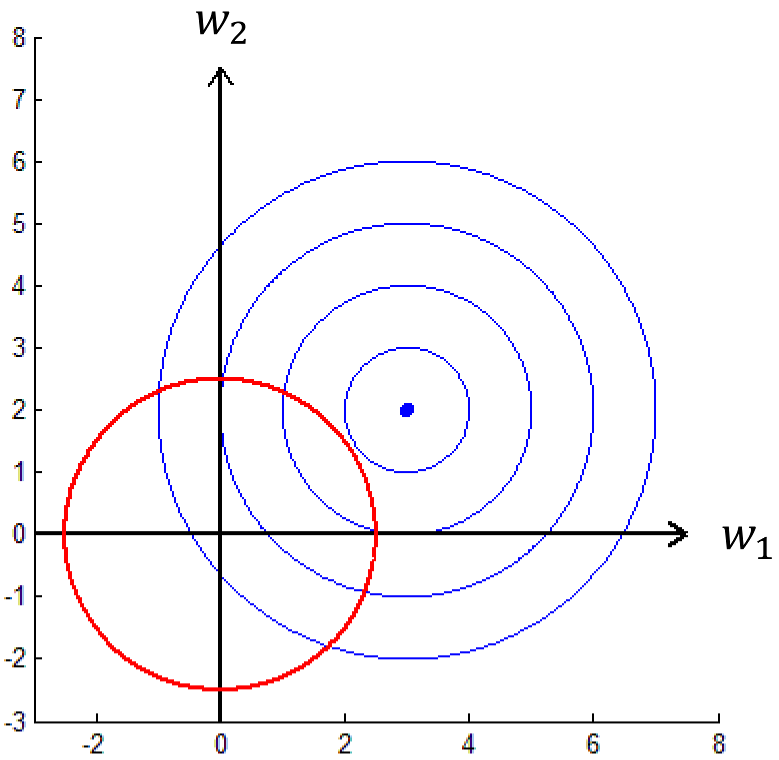


Regularization (Graphical Representation)



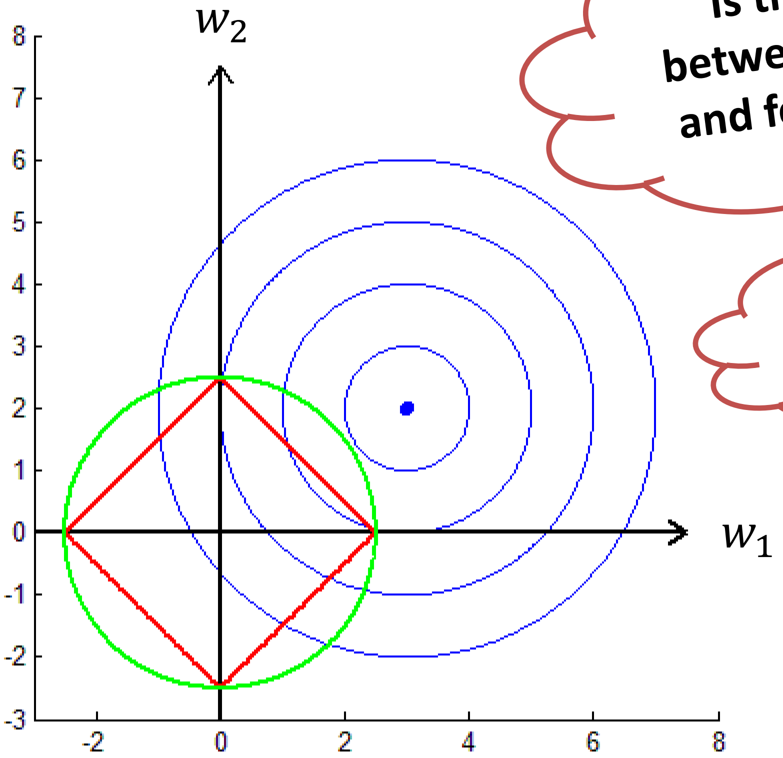
$$\mathcal{J}(\boldsymbol{W}) = \sum_{i=1}^{n} \boldsymbol{\mathcal{L}}\big(y_i, f(\boldsymbol{x}_i, \boldsymbol{W})\big) + \lambda \|\boldsymbol{W}\|_1$$

Regularization (Graphical Representation)



$$\mathcal{J}(\boldsymbol{W}) = \sum_{i=1}^{n} \boldsymbol{\mathcal{L}}\big(y_i, f(\boldsymbol{x}_i, \boldsymbol{W})\big) + \lambda \|\boldsymbol{W}\|_2$$

Regularization (Graphical Representation)



L_1 *regularization* *vs.* L_2 *regularization*

Is there a relation between regularization and feature selection?

Remember that $\widehat{y_i} = \boldsymbol{W}^T \boldsymbol{x}_i$

Probabilistic Error Bounds

Probabilistic Error Bounds

- Linear classifiers $y \in \{0, 1\}$

$$h_\theta(x) = g(\theta^T x)$$
$$g(z) = I\{z \geq 0\}$$

- Training error of h_θ (Risk)

$$\hat{\varepsilon}(h_\theta) = \frac{1}{\mathrm{m}} \sum_{i=1}^{m} I\{h_\theta(x_i) \neq y_i\}$$

- Empirical Risk Minimization (ERM)

$$\hat{\theta} = \arg\min_{\theta} \hat{\varepsilon}(h_\theta)$$

ERM generally leads to a non-convex optimization

Some methods (e.g. logistic regression and SVM) try to solve a convex approximation to this non-convex problem

Probabilistic Error Bounds

- Hypotheses class

$$\mathcal{H} = \{h_\theta : \theta \in \mathbb{R}^{n+1}\}$$
$$h_\theta \colon \mathcal{X} \to \{0,1\}$$

- Empirical Risk Minimization

$$\hat{h} = \arg\min_{h\in\mathcal{H}} \hat{\varepsilon}(h)$$

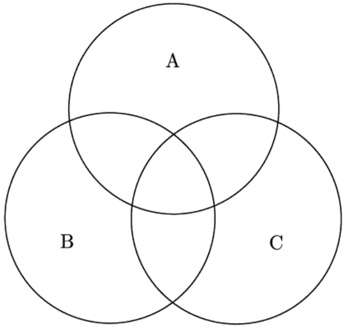
- Generalization error

$$\varepsilon(h) = P_{(x,y)\sim\mathfrak{D}}(h(x) \neq y)$$

Probabilistic Error Bounds (Preliminaries)

- Union bound
  - For a set of probabilistic events $A_1, A_2, \dots, A_k$ (not necessarily independent)

$$P(A_1 \cup A_2 \cup \cdots \cup A_k) \leq P(A_1) + P(A_2) + \cdots + P(A_k)$$



- Hoeffding inequality
  - Let $z_1, z_2, \dots, z_m$ be m I.I.D $Bernoulli(\phi)$ random variables: $P(z_i = 1) = \phi$

$$\hat{\phi} = \frac{1}{m}\sum_{i=1}^{m} z_i$$

  - For any fixed $\gamma > 0$

$$P\left(\left|\hat{\phi} - \phi\right| > \gamma\right) \leq 2\exp(-2\gamma^2 m)$$

Remember to plot a figure on the board ☺

Probabilistic Error Bounds (Preliminaries)

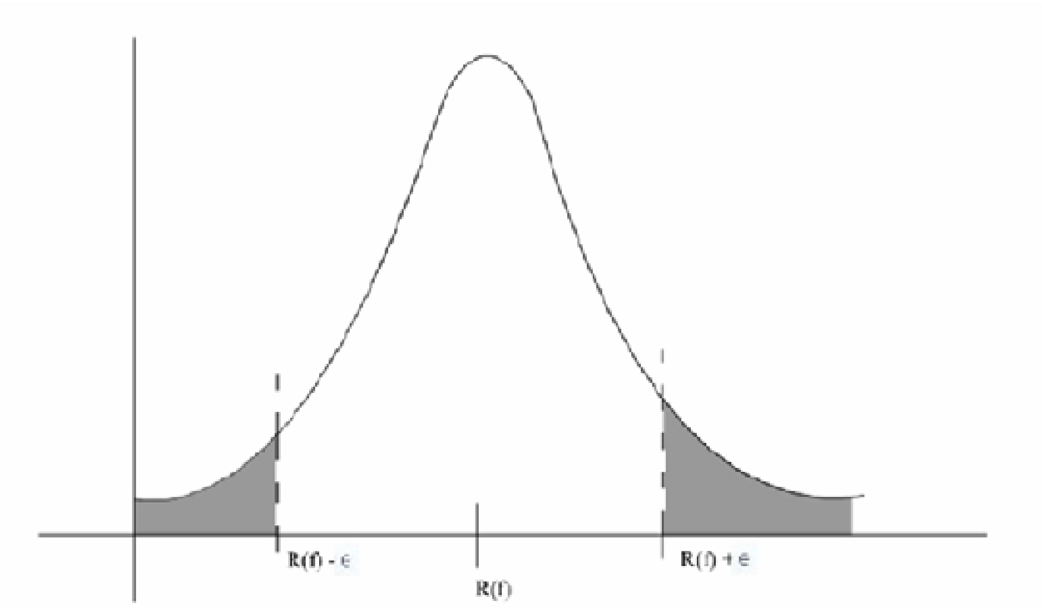
$$P\left(\left|\hat{\phi} - \phi\right| > \gamma\right) \leq 2\exp(-2\gamma^2 m)$$

- $\gamma = 0.1, m = 100$

  $$P\left(\left|\hat{\phi} - \phi\right| > 0.1\right) \leq 2\exp(-2 \times 0.01 \times 100) = 0.27$$

- $\gamma = 0.1, m = 300$

  $$P\left(\left|\hat{\phi} - \phi\right| > 0.1\right) \leq 2\exp(-2 \times 0.01 \times 300) = 0.005$$



Probabilistic Error Bounds (finite $\mathcal{H}$)

- Finite hypotheses class $\mathcal{H}$

$$\mathcal{H} = \{h_1, h_2, \dots, h_k\}$$
$$\hat{h} = \arg\min_{h \in \mathcal{H}} \hat{\varepsilon}(h)$$

- At first, we have to show that training error is a good approximation of generalization error $\hat{\varepsilon} \approx \varepsilon$
- Then we have to prove a bound on $\varepsilon(\hat{h})$
- Fix any $h_j \in \mathcal{H}$
- Define z_i

$$z_i = I\{h_j(x_i) \neq y_i\}$$
$$z_i \in \{0, 1\}$$

- By definition the probability of misclassification is the generalization error

$$P(z_i = 1) = \varepsilon(h_j)$$

$z_i\ s\ are$ *independent and identically distributed (I.I.D)*

Probabilistic Error Bounds ($\hat{\varepsilon} \approx \varepsilon$, finite $\mathcal{H}$)

$$\hat{\varepsilon}(h_j) = \frac{1}{m}\sum_{i=1}^{m} z_i = \frac{1}{\mathrm{m}}\sum_{i=1}^{m} I\{h_\theta(x_i) \neq y_i\}$$

- The true mean is given by generalization error $\varepsilon(h_j)$
- By Hoeffding inequality

$$P\left(\left|\varepsilon(h_j) - \hat{\varepsilon}(h_j)\right| > \gamma\right) \leq 2\exp(-2\gamma^2 m)$$

  - For a fixed hypothesis h_j the probability that my training error (estimation of the true mean error) is far away from the generalization error (true mean error) is small
  - Now we have to prove the bound in general for the whole hypotheses class $\mathcal{H}$ (that holds true for all of the hypotheses in $\mathcal{H}$)

Probabilistic Error Bounds ($\hat{\varepsilon} \approx \varepsilon$, finite $\mathcal{H}$)

- A_j : event that $\left|\varepsilon(h_j) - \hat{\varepsilon}(h_j)\right| > \gamma$
- We have proved that

$$P(A_j) \leq 2\exp(-2\gamma^2 m)$$

- What can we say about this:

$$P\left(\exists h_j \in \mathcal{H} : \left|\varepsilon(h_j) - \hat{\varepsilon}(h_j)\right| > \gamma\right)$$

$$= P(A_1 \cup A_2 \cup \cdots \cup A_k) \leq \sum_{i=1}^{k} P(A_i)$$

$$\leq \sum_{i=1}^{k} 2\exp(-2\gamma^2 m) = 2k\exp(-2\gamma^2 m)$$

$$P\left(\exists h_j \in \mathcal{H} : \left|\varepsilon(h_j) - \hat{\varepsilon}(h_j)\right| > \gamma\right) \leq 2k\exp(-2\gamma^2 m)$$

Probabilistic Error Bounds ($\hat{\varepsilon} \approx \varepsilon$, finite $\mathcal{H}$)

- $1 - both\ sides$

$$P\left(\nexists h_j \in \mathcal{H} : \left|\varepsilon\left(h_j\right) - \hat{\varepsilon}\left(h_j\right)\right| > \gamma\right) = P\left(\forall h_j \in \mathcal{H} : \left|\varepsilon\left(h_j\right) - \hat{\varepsilon}\left(h_j\right)\right| \leq \gamma\right)$$

$$P\left(\forall h_j \in \mathcal{H} : \left|\varepsilon\left(h_j\right) - \hat{\varepsilon}\left(h_j\right)\right| \leq \gamma\right) \geq 1 - 2k \exp(-2\gamma^2 m)$$

- With probability of at least $1 - 2k \exp(-2\gamma^2 m)$, $\hat{\varepsilon}(h)$ will be within γ of $\varepsilon(h)$ for all $h \in \mathcal{H}$

Uniform Convergence

- Uniform convergence: As m becomes large, all $\hat{\varepsilon}\left(h_j\right)$ converge to $\varepsilon\left(h_j\right)$
- So far we have proved that the training error is a good approximation of the generalization error

Probabilistic Error Bounds (Sample Complexity)

- Define $\delta = 2k\exp(-2\gamma^2 m)$
- Given δ, γ what is m ?

$$m \geq \frac{1}{2\gamma^2}\log\frac{2k}{\delta}$$

- As long as $m \geq \frac{1}{2\gamma^2}\log\frac{2k}{\delta}$ then with probability of at least $1-\delta$ we have $\left|\varepsilon(h_j) - \hat{\varepsilon}(h_j)\right| \leq \gamma$ for all $h \in \mathcal{H}$

$$m = O(\frac{1}{\gamma^2}\log\frac{k}{\delta})$$

Sample complexity bound

Probabilistic Error Bounds (Generalization Bound)

- Define $\delta = 2k\exp(-2\gamma^2 m)$
- Given δ, m what is γ?

$$|\varepsilon(h_j) - \hat{\varepsilon}(h_j)| \leq \sqrt{\frac{1}{2m}\log\frac{2k}{\delta}}$$

γ

- With probability of at least $1-\delta$ we have that for all $h \in \mathcal{H}$,

$$|\varepsilon(h_j) - \hat{\varepsilon}(h_j)| \leq \sqrt{\frac{1}{2m}\log\frac{2k}{\delta}}$$

Generalization error bound

Probabilistic Error Bounds (Final Step)

- We have showed that $\forall h \in \mathcal{H}$ uniform convergence holds with high probability

$$\left|\varepsilon\left(h_j\right) - \hat{\varepsilon}\left(h_j\right)\right| \leq \gamma$$

$$\hat{h} = \arg\min_{h \in \mathcal{H}} \hat{\varepsilon}(h)$$
$$h^* = \arg\min_{h \in \mathcal{H}} \varepsilon(h)$$

$$\varepsilon\left(\hat{h}\right) \leq \hat{\varepsilon}\left(\hat{h}\right) + \gamma$$
$$\varepsilon\left(\hat{h}\right) \leq \hat{\varepsilon}(h^*) + \gamma$$
$$\varepsilon\left(\hat{h}\right) \leq \varepsilon(h^*) + \gamma + \gamma$$
$$\varepsilon\left(\hat{h}\right) \leq \varepsilon(h^*) + 2\gamma$$

Probabilistic Error Bounds (Bias-Variance)

- Theorem
- Let $|\mathcal{H}| = k$ and let m, δ be fixed, then with probability of a least $1 - \delta$

$$\varepsilon(\hat{h}) \leq \underbrace{\left(\min_{h\in\mathcal{H}} \varepsilon(h)\right)}_{\boldsymbol{\varepsilon(h^*)}} + \underbrace{2\sqrt{\frac{1}{2m}\log\frac{2k}{\delta}}}_{\boldsymbol{\gamma}}$$

- Switch from $\mathcal{H}$ to a larger class of hypotheses $\mathcal{H}'$

$$\mathcal{H} \subseteq \mathcal{H}'$$

Linear **Quadratic**

  - Bias decreases and variance increases

Probabilistic Error Bounds (Sample Complexity)

- Corollary
- Let $|\mathcal{H}| = k$ and let δ, γ be fixed, then in order to guarantee that

$$\varepsilon(\hat{h}) \leq \left(\min_{h\in\mathcal{H}} \varepsilon(h)\right) + 2\gamma$$

with probability of a least $1 - \delta$, it suffices that

$$m \geq \frac{1}{2\gamma^2} \log\frac{2k}{\delta} = O(\frac{1}{\gamma^2} \log\frac{k}{\delta})$$

Probabilistic Error Bounds (infinite $\mathcal{H}$)

- What can we say when we have infinite number of hypotheses?

- Assume that my learning algorithm is parameterized by d parameters
  - In our digital computers each parameter (real value) is represented by a double precision number
  - The number of possibilities for each parameter is: 2^{64}
  - And the total number of possibilities by these d parameters: 2^{64d}

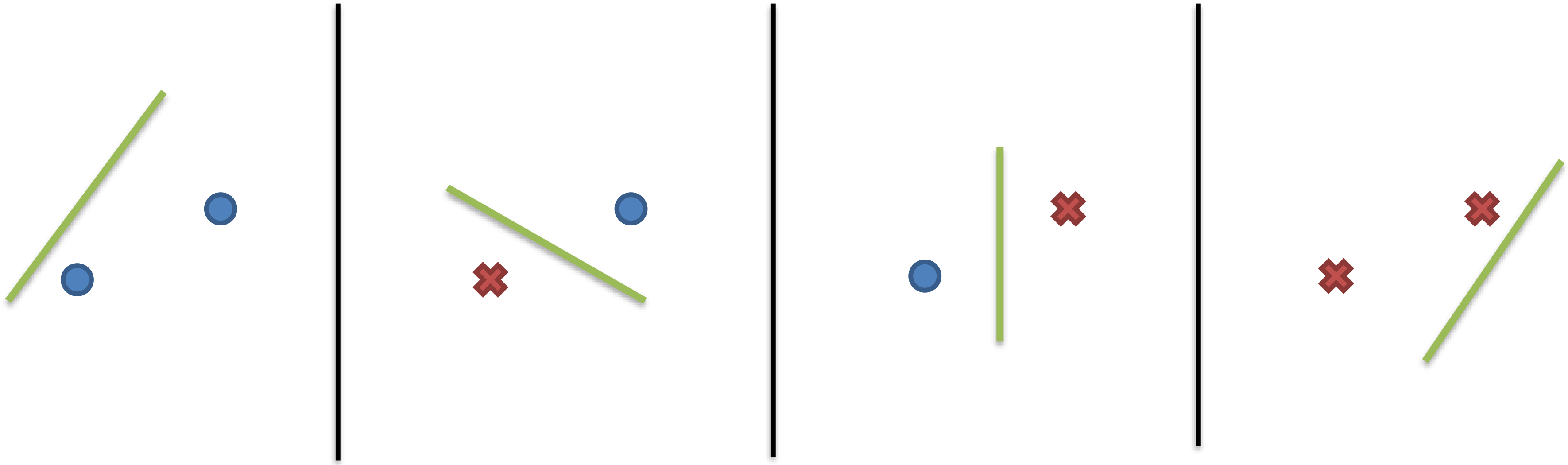
$$|\mathcal{H}| = 2^{64d}$$

$$m = O(\frac{1}{\gamma^2}\log\frac{k}{\delta}) = O(\frac{1}{\gamma^2}\log\frac{2^{64d}}{\delta})$$

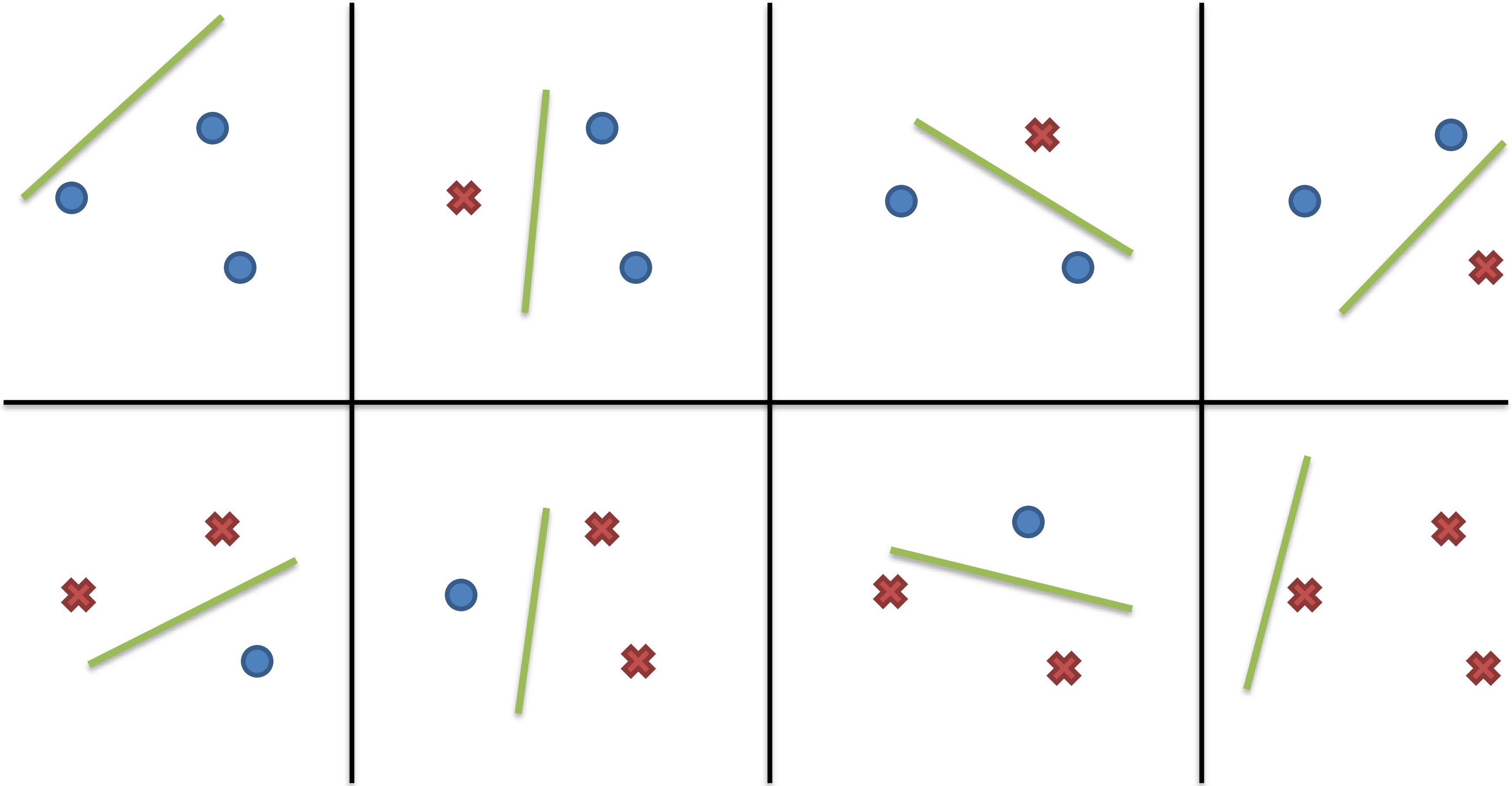
$$m = O(\frac{d}{\gamma^2}\log\frac{1}{\delta})$$

Shattering

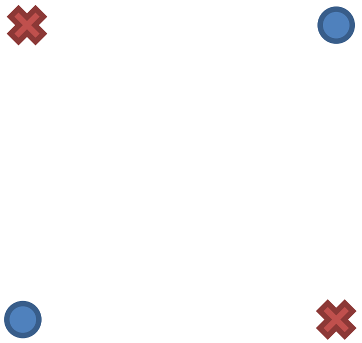
- Given a dataset $S = \{x_1, x_2, \dots, x_m\}$, we say $\mathcal{H}$ shatters S if $\mathcal{H}$ can learn any arbitrary labeling of S. In fact for any of the 2^m possible labeling $\exists h \in \mathcal{H}$ that can perfectly classify that labeling.



Shattering and VC dimension



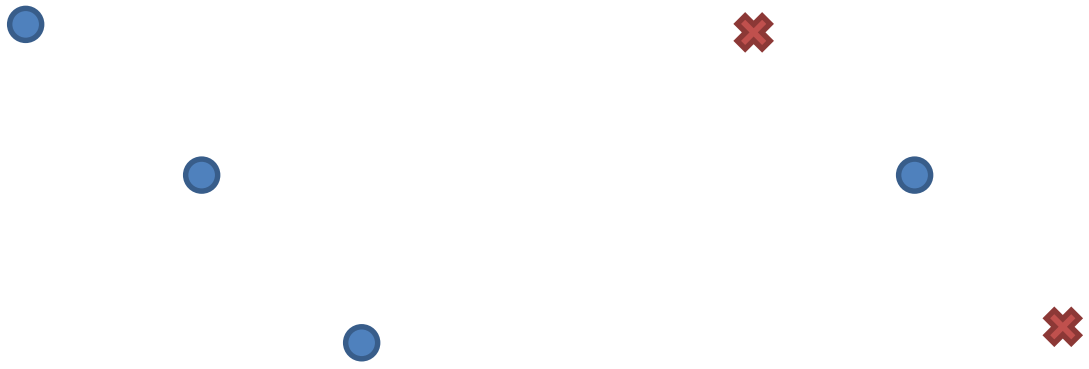
Shattering and VC dimension



- Vapnik-Chervonenkis (VC) dimension for the class of hypotheses $\mathcal{H}$ is known to be the maximum number of data points that $\mathcal{H}$ can shatter

$$VC(\mathcal{H})$$

Shattering and VC dimension



$$VC(\{linear\ classifiers\ in\ n - dimensional\ space\}) = n + 1$$

VC dimension

- Theorem

- Let $\mathcal{H}$ be given and let $VC(\mathcal{H}) = d$ then with probability of at least $1 - \delta$ we have that $\forall h \in \mathcal{H}$

$$|\varepsilon(h) - \hat{\varepsilon}(h)| \leq O\left(\sqrt{\frac{d}{m}\log\frac{m}{d} + \frac{1}{m}\log\frac{1}{\delta}}\right)$$

- Thus, with probability of at least $1 - \delta$ we also have

$$\varepsilon(\hat{h}) \leq \varepsilon(h^*) + O\left(\sqrt{\frac{d}{m}\log\frac{m}{d} + \frac{1}{m}\log\frac{1}{\delta}}\right)$$

VC dimension

- Corollary

- To guarantee that with probability of at least $1 - \delta$ we have $\varepsilon(\hat{h}) \leq \varepsilon(h^{*}) + 2\gamma$, it suffices that

$$m = O_{\gamma,\delta}(d)$$

- Sample complexity is upper bounded by the VC dimension (and also lower bounded)

VC dimension

- For most reasonable class of hypotheses (e.g. linear, logistic, …) the VC dimension is very similar to the number of parameters in the model. (informal ☺)
  - Therefore, the number of training examples you need to train your model is roughly linear to the number of parameters in your model

- Large VC dimension may lead to overfitting

Support Vector Machines (SVM)

- SVM is a large margin linear classifier
- Accuracy (low bias and very high accuracy)
  - By using kernels SVM maps the data to an infinite dimensional feature space and uses a linear classifier in the high dimensional space
  - It is highly non-linear in the original feature space but it is linear in the infinite dimensional feature space
- VC dimension and overfitting
  - It is linear in infinite dimensional feature space, so the VC dimension should be infinite and therefore it overfits !!! (not true)

VC dimension for Large Margin Classifiers

- Consider the class of linear large margin classifiers
- Assume that the minimum acceptable margin is γ
  - Those lines that are closer than γ to any data point are not acceptable
  - With this restriction, the number of possible linear hyper-planes are smaller

$$if \;\; \|x_i\|_2 \leq R$$

$$VC(\mathcal{H}) \leq \left\lceil \frac{R^2}{4\gamma^2} \right\rceil + 1$$

- This bound does not depend on the dimensionality of points. Therefore, even if we have infinite dimensional feature space, as long as we have the margin constraint the VC dimension is bounded.

Support Vector Machines (SVM)

- SVM is a large margin linear classifier
- The VC dimension is bounded
- The VC dimension does not depend on the dimensionality of the data

$$\mathcal{H} \underset{\boxed{\text{Kernel}\uparrow}}{\longrightarrow} \mathcal{H}' \underset{\boxed{\text{Margin}\downarrow}}{\longrightarrow} \mathcal{H}''$$

- By using kernels we go to a much richer class of hypotheses and then by maximizing the margin we filter out useless hypotheses (most of them).
  - Therefore, $\mathcal{H}''$ is not just any hypotheses, it is the collection of good ones ☺