Machine Learning

Prof. Sergei Gleyzer

Lecture

PH451, PH551

Jan 30, 2025

Announcements

- HS #2 due on Tue. Feb 4 1pm
- Quiz on Thu. Feb 13 during class

Bias Variance Trade-Off

Sources of generalization error:

- **Bias:**
  - (Wrong) assumptions about data and model
- **Variance:**
  - Model DoF and sensitivity to variations in training dataset (many dof, high variance - overfitting model)
- **Resolution** (Irreducible error):
  - Noisy data

Trade-off! e.g. Increase model complexity:

- **reduce bias**, **increase variance** and vice versa

Some Known Issues in ML

Data effectiveness

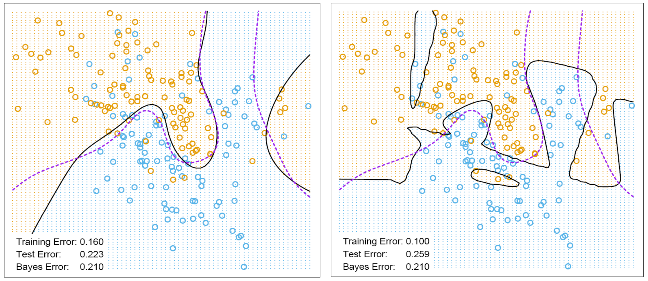
- More data better results (training data complexity)?

Feature and Data relevance/quality

- GIGO

Non-representative data

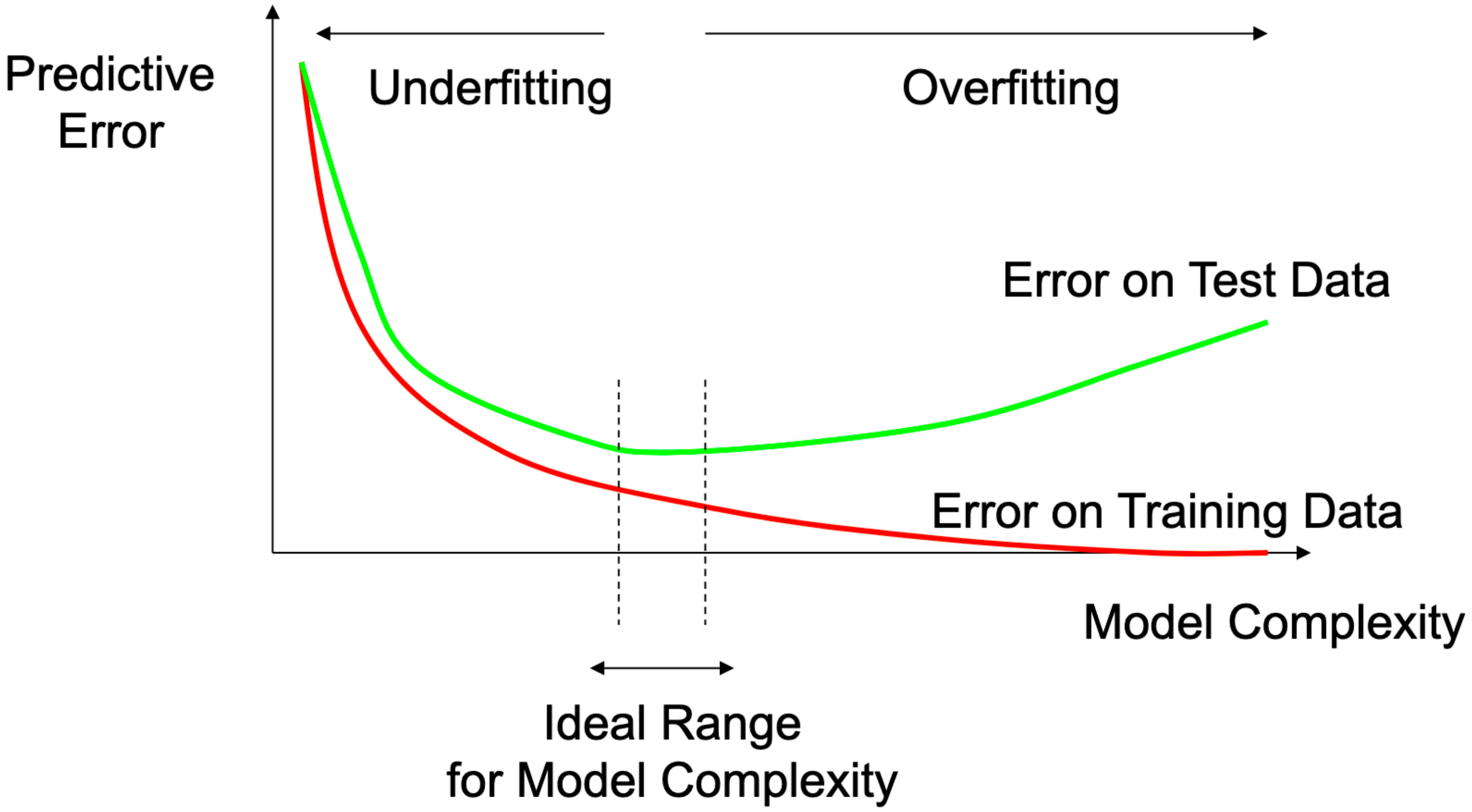
- “Sampling bias”



Overfitting

- Model too complex for dataset

Some Known Issues in ML



Regularization

Regularize to reduce overfitting

- **Constrain** the model
- Idea:
  - **penalize** for **large** values of θ_j
- Various types of regularization approaches:
  - **Ridge Regression**
  - **Lasso**

Ridge Regression

Regularized version of Linear Regression

$$J(\boldsymbol{\theta}) = \text{MSE}(\boldsymbol{\theta}) + \alpha\frac{1}{2}\Sigma_{i=1}^{n}\theta_i^{\,2}$$

Ridge Regression
Cost Function Eqn. 4-8

- forces weights to be small as possible
- regularize only during training, apply without

Note:

1) $\alpha = 0$: no regularization
2) i starts at 1 (θ_0 is not regularized)
3) is also known as L2 penalty/regularization
(½ of square of L_2 "euclidian distance" norm)

Lasso Regression

Least **A**bsolute **S**hrinkage and **S**election **O**perator Regression

$$J(\boldsymbol{\theta}) = \mathrm{MSE}(\boldsymbol{\theta}) + \alpha \Sigma_{i=1}^{n} \left|\theta_i\right|$$

Lasso Regression
Cost Function Eqn. 4-10

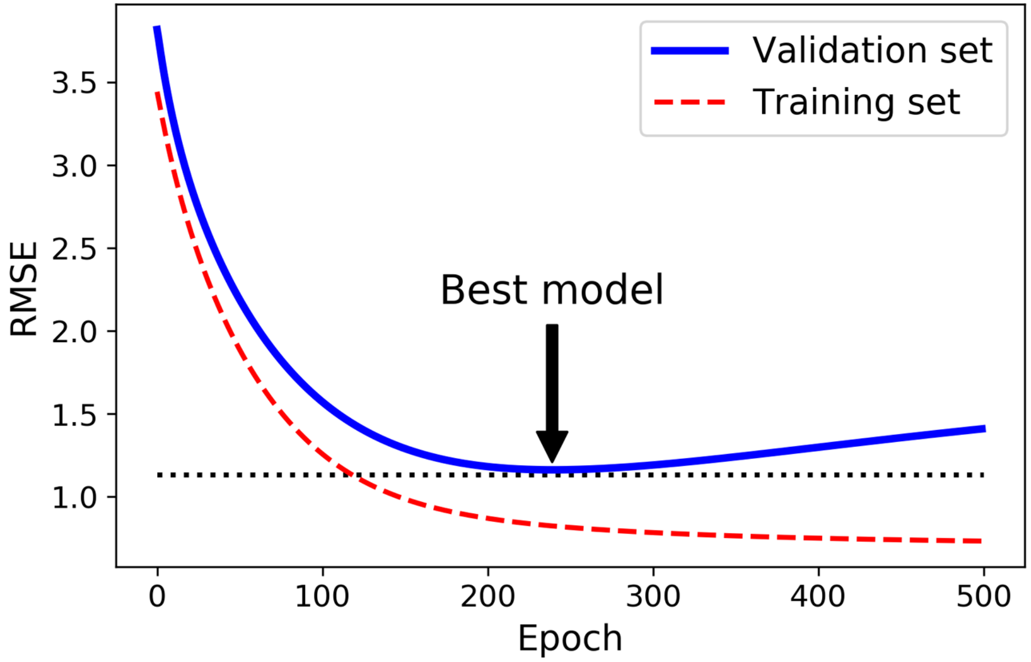
- Also known as **L1 regularization** as it uses the L1 norm
  - L1 = sum of absolutes (“Manhattan” norm)
  - Eliminates weights of least important features
  - If you suspect that there are useless features, Lasso is preferable

"Regression Shrinkage and Selection via the lasso” Tibshirani (1996)

Early Stopping

Early stopping regularization

- Stop the training early
- Avoids overfitting



Logistic Regression

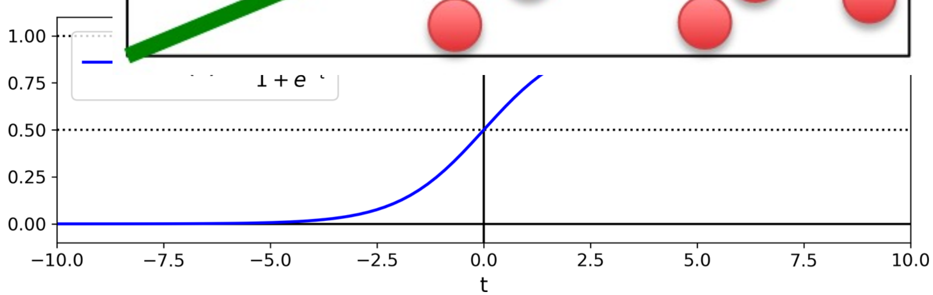
Can use some regression algorithms for classification

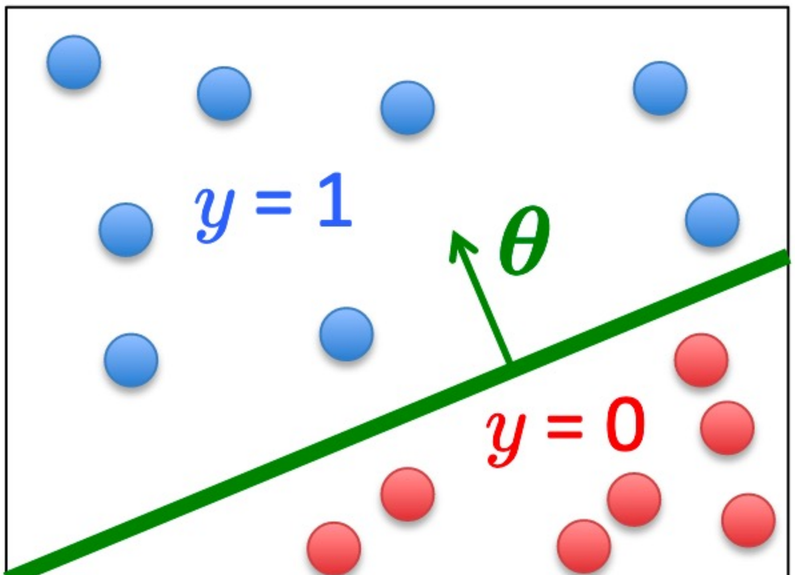
- Estimate **reshold**

Example:

- Works except
you out

$\sigma(t$ **ction** Eqn. 4-14

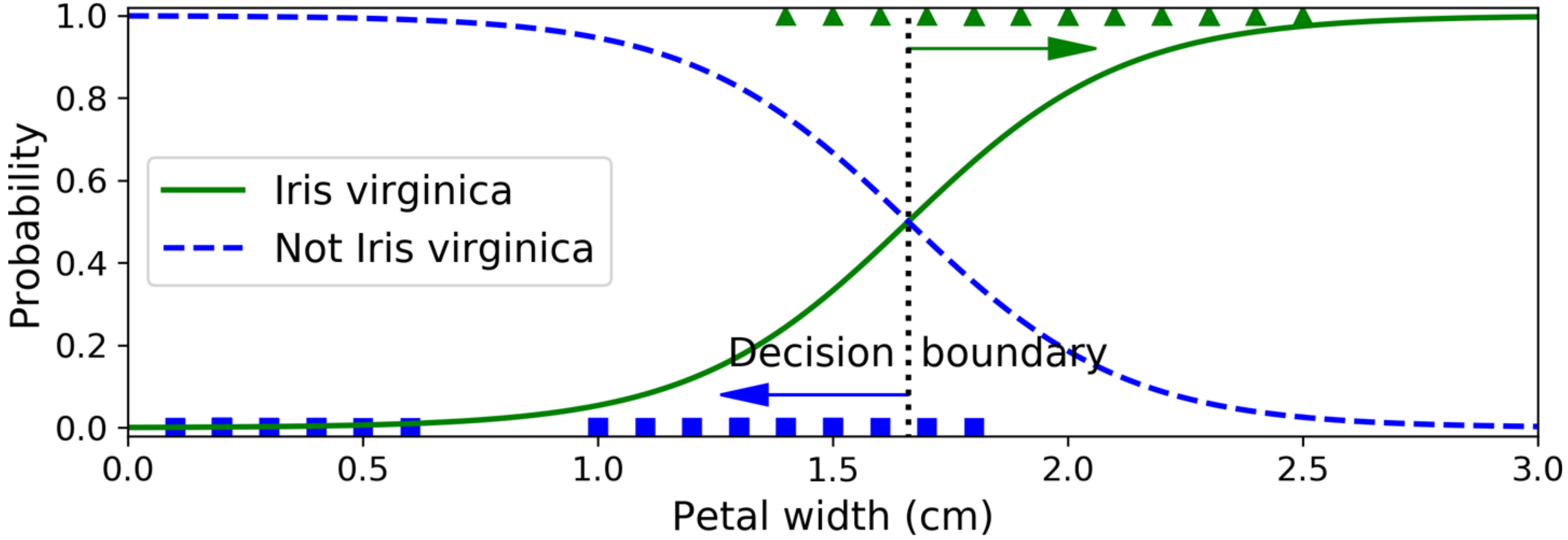




Sigmoid
Function

Logistic Regression

Decision boundary and probability:



- **Dashed line is where model estimates 50% probability**

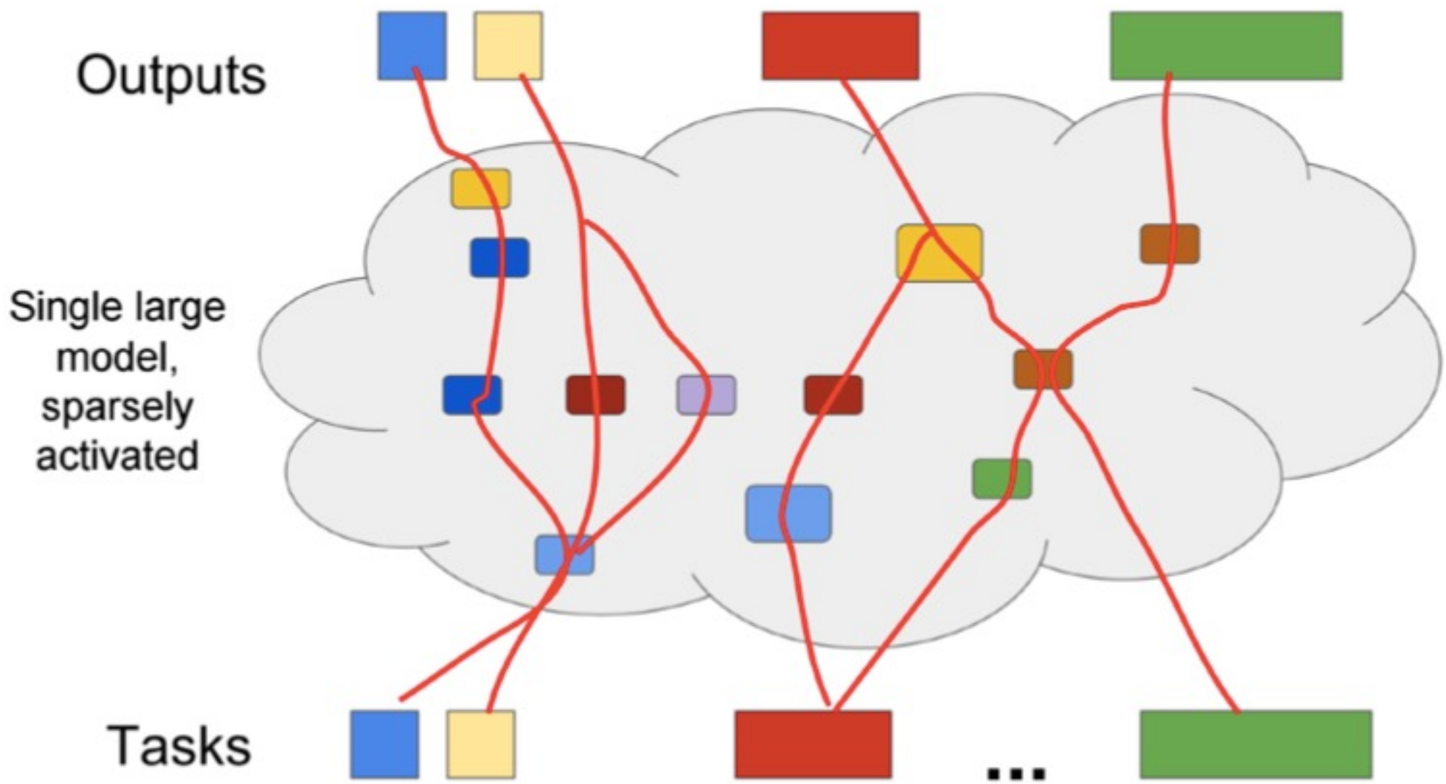
Multi-Class Classification

Extension of binary classification to multiple classes

- **Strategies:**
  - **1 vs. 1**
  - **1 vs. Rest**

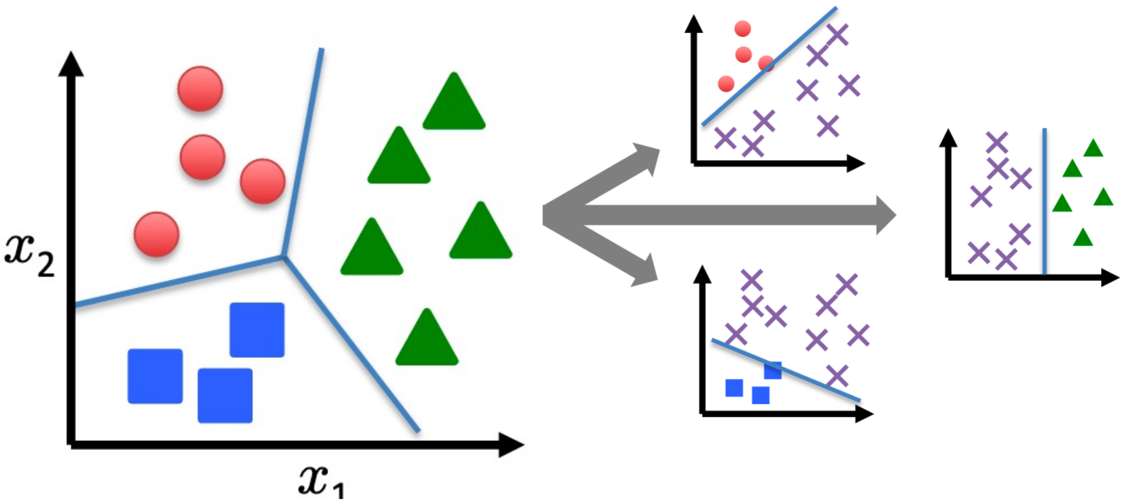
  Not all classifiers can do it

Multi-Task Model



Multi-Class Logistic Regression

Can use Logistic Regression for multi-class:



Split into N 1-vs-rest problems and train logistic regression for each class

- Gradient descent and predict most probable label

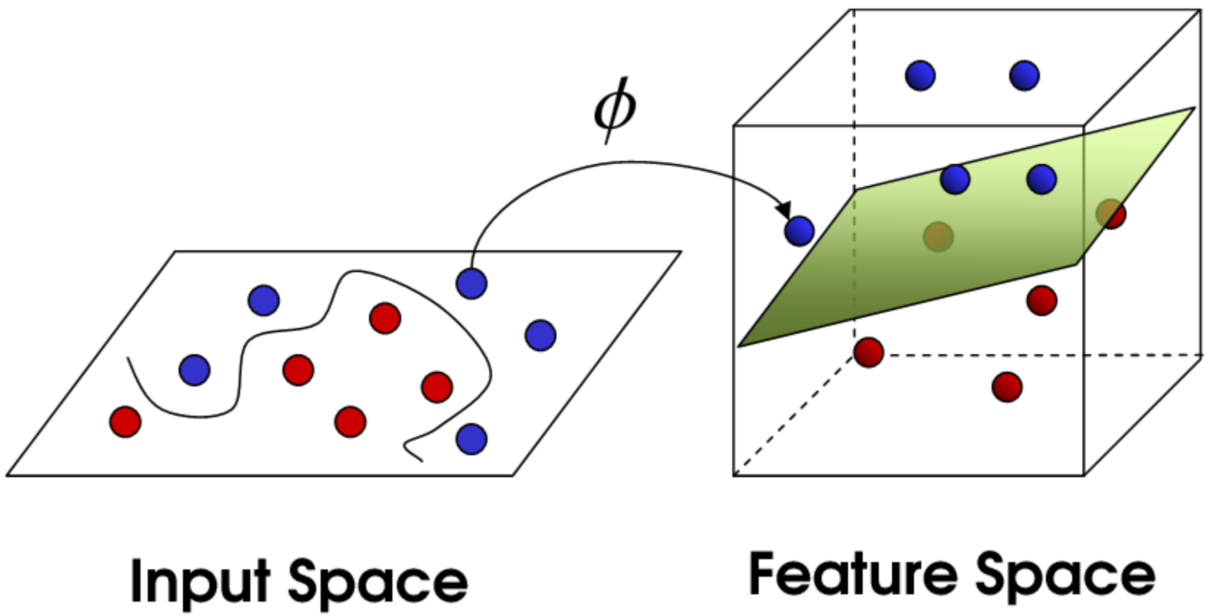
Softmax Regression

- For each class compute a score for every instance
- Apply softmax function to the scores to estimate class probability:

$$\hat{p}_k = \sigma(\mathbf{s}(\mathbf{x}))_k = \frac{\exp\left(s_k(\mathbf{x})\right)}{\sum_{j=1}^{K} \exp\left(s_j(\mathbf{x})\right)}$$

- i.e. compute exponential of every score then normalize by the sum of exponentials

Support Vector Machines



Support Vector Machines

Can be useful for:

- **Linear Classification**
- **Non-Linear Classification**
- **Regression**
- **Anomaly Detection**

- **Idea:**
  - Fit/create the widest possible road between the classes – "**large margin classification**"

Support Vector Machines

non-separable data in *d*-dimensions may be better separated if mapped into a higher dimensional space

a hyper-plane is used to partition the high dimensional space

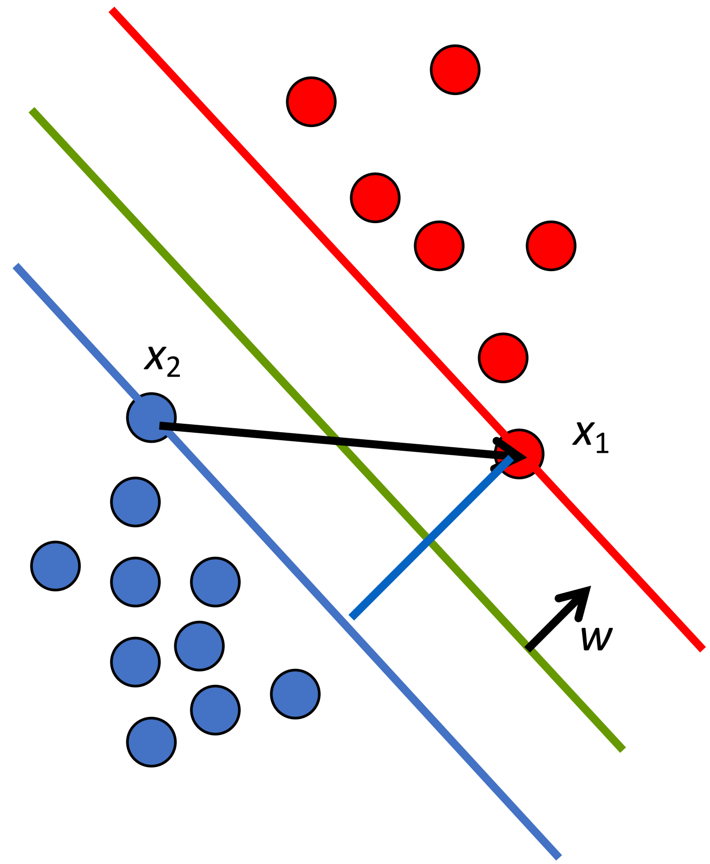
$$h : \Re^d \rightarrow \Re^\infty$$

$$f(x) = w \cdot h(x) + c$$

Support Vector Machines

green plane: $\hat{y} = 0$
red plane: $\hat{y} = +1$
blue plane: $\hat{y} = -1$

The distance between **red** and **blue** planes is called the **margin**

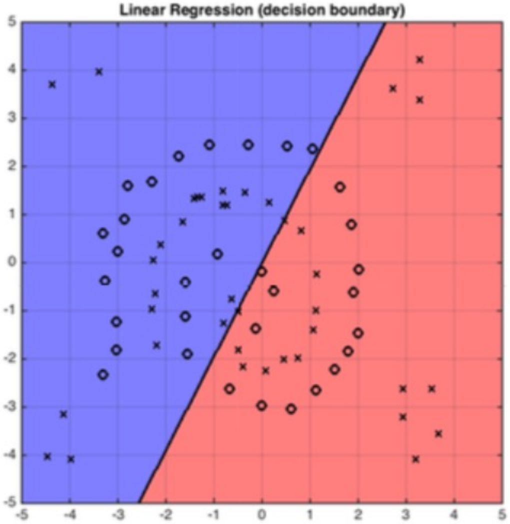


Maximizing margin is equivalent to minimizing $||w||^2$

Kernels

Linear classifiers lead to **linear decision boundaries**

- What if the boundary is more complicated?



Kernels

Possible solution:

- apply **feature transformations** to **input feature** vectors

  Advantage: problem stays convex and well-behaved

  - can use gradient descent

  Disadvantage: transformation usually called $\phi(x)$ is very high dimensional to capture non-linear interactions of the input features

  - typically, too slow

Kernel Trick

Kernel Functions: enable algorithms to operate in **high-dimensional feature** space

Kernel Trick: avoid the computational bottleneck by computing **weights** and **inner products** instead of **data points** themselves

- For example, in gradient descent you only need the **inner products** of all pairs of data vectors to learn a hyper-plane classifier

Kernel Functions

Vectors: a and b

Linear Kernel: good starting point

$$K(a,b)=a^{T}b$$

Polynomial Kernel: useful for non-linear data

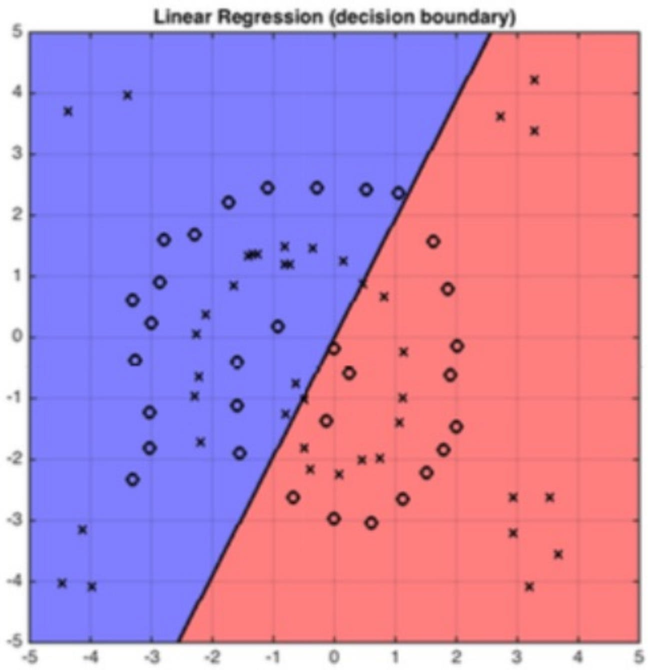
$$K(a,b) = (\gamma a^{T}b+r)^{d}$$

Gaussian Radial Basis Function Kernel works well for many cases:

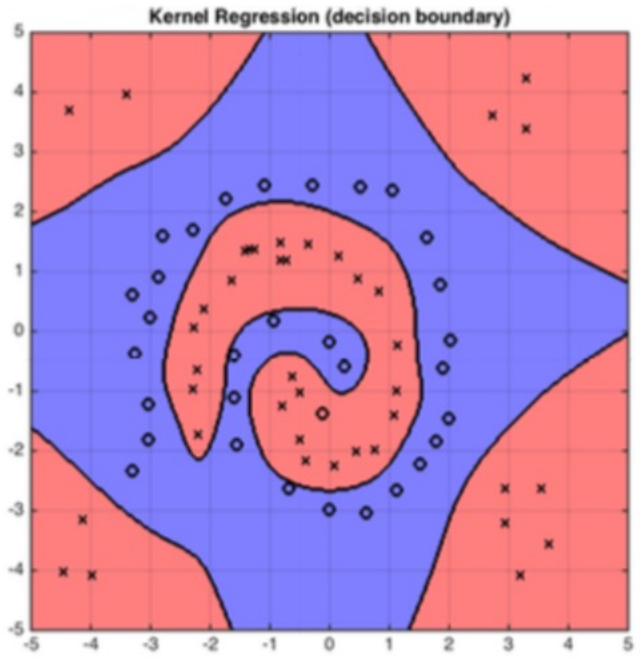
$$K(a,b)=\exp(-\gamma\|a-b\|^{2})$$

Others: exponential, laplacian, sigmoid

Kernel Functions



Linear



Gaussian RBF