Discriminative methods

- Brief overview of machine learning
- Linear regression
- Logistic regression
- Decision trees
- Boosting idea
- SVM

Inductive learning method

Construct/adjust $h$ to agree with $f$ on training set
($h$ is consistent if it agrees with $f$ on all examples)

E.g., curve fitting:
Inductive learning method

Construct/adjust $h$ to agree with $f$ on training set
($h$ is consistent if it agrees with $f$ on all examples)

E.g., curve fitting:

\[ f(x) \]

\[ x \]
Inductive learning method

Construct/adjust $h$ to agree with $f$ on training set
($h$ is consistent if it agrees with $f$ on all examples)

E.g., curve fitting:

Linear regression

- Fit function to the data so you can predict future values
- Given data $(x_1, y_1), (x_2, y_2), \ldots (x_n, y_n)$
- Find such hypothesis $f$, such that $y = f(x)$ has small error of future data
- Choose the form of function and estimate the data using linear least squares techniques (in closed form, or gradient descent)
Logistic Regression

- Similar as linear regression, but now y’s are only +/- or 0/1 denoting positive or negative examples of a class
- We know that our function should have values 0 or 1
- Transform to continuous setting – construct such as $h$ where function would have values only between 0-1
  \[ h_{\theta}(x) = g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}} \]
- $g(.)$ is a sigmoid, logistic function
  \[ g(z) = \frac{1}{1 + e^{-z}} \]
- Property of the derivative
  \[ g'(z) = g(z)(1 - g(z)) \]

Logistic Regression

- How do we find parameter $\theta$
- Find such parameters so as to maximize likelihood of the data assume that
  \[ P(y = 1|x; \theta) = h_{\theta}(x) \]
  \[ P(y = 0|x; \theta) = 1 - h_{\theta}(x) \]
- More compactly
  \[ p(y|x; \theta) = (h_{\theta})^y (1 - h_{\theta})^{1-y} \]
- Resulting gradient ascent rule
  \[ \theta_j = \theta_j + \alpha (y^{(i)} - h_{\theta}(x^{(i)}))x^{(i)} \]
- Closely related to perceptron learning algorithm where values are forced to be 0-1 - no clear probabilistic interpretation
Ensemble learning

- Idea: instead of keeping simple hypothesis – keep multiple of them – ensemble of hypotheses
- Increased expressive power
- Enables us to combine several simpler hypotheses to get better prediction

---

**Ensemble learning**

- **Boosting**
  - Idea: weighted training set – each training example has an associated weight $w \geq 0$
  - The higher the weight, higher importance of the example in the learning stage
  - Start with all examples with weight 1.
  - Generate hypothesis $h_1$
  - $h_1$ classifies some examples correctly some incorrectly
  - Next hypothesis should do better on the missclassified examples – i.e. increase their weights
  - Given new weighted training set generate $h_2$ etc
  - We will see examples of these later
Linear classifiers

- Find linear function (hyperplane) to separate positive and negative examples – many such hyperplanes

\[ x_i \text{ positive: } x_i \cdot w + b \geq 0 \]
\[ x_i \text{ negative: } x_i \cdot w + b < 0 \]

Which hyperplane is best?

Support vector machines

- Find hyperplane that maximizes the margin between the positive and negative examples

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998
Support vector machines

- Find hyperplane that maximizes the margin between the positive and negative examples

\[ x_i, \text{positive} (y_i = 1) : \quad x_i \cdot w + b \geq 1 \]
\[ x_i, \text{negative} (y_i = -1) : \quad x_i \cdot w + b \leq -1 \]

For support, vectors, \( x_i \cdot w + b = \pm 1 \)

Distance between point and hyperplane:
\[ \left| x_i \cdot w + b \right| \]
\[ \| w \| \]

Therefore, the margin is \( 2 / ||w|| \)

C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, 1998

Summary: Discriminative methods

- Nearest-neighbor and k-nearest-neighbor classifiers
  - L1 distance, \( \chi^2 \) distance, quadratic distance, Earth Mover’s Distance

- Support vector machines
  - Linear classifiers
  - Margin maximization
  - The kernel trick
  - Kernel functions: histogram intersection, generalized Gaussian, pyramid match
  - Multi-class

- Of course, there are many other classifiers out there
  - Neural networks, boosting, decision trees, …