Regularized Linear Models

CS6780 – Advanced Machine Learning
Spring 2019

Thorsten Joachims
Cornell University

Reading: Murphy 8.1-8.3, Murphy 7.5
Discriminative ERM Learning

• Modeling Step:
  • Select classification rules $H$ to consider (hypothesis space)

• Training Principle:
  • Given training sample $(\vec{x}_1, y_1), \ldots, (\vec{x}_n, y_n)$
  • Find $h$ from $H$ with lowest training error $\rightarrow$ Empirical Risk Minimization
  • Argument: generalization error bounds $\rightarrow$ low training error leads to low prediction error, if overfitting is controlled.

• Examples: SVM, decision trees, Perceptron
Bayes Decision Rule

• Assumption:
  – learning task \( P(X,Y)=P(Y|X)\ P(X) \) is known

• Question:
  – Given instance \( x \), how should it be classified to minimize prediction error?

• Bayes Decision Rule (for zero/one loss):

\[
h_{\text{bayes}}(\tilde{x}) = \underset{y \in Y}{\text{argmax}} [P(Y = y | X = \tilde{x})]
\]

\[= \underset{y \in Y}{\text{argmax}} [P(Y = y, X = \tilde{x})]
\]
Generative vs. Conditional vs. ERM

- **Empirical Risk Minimization**
  - Find \( h = \arg\min_{h \in H} \text{Err}_S(h) \) s.t. overfitting control
  - Pro: directly estimate decision rule
  - Con: need to commit to loss, input, and output before training

- **Discriminative Conditional Model**
  - Find \( P(Y|X) \), then derive \( h(x) \) via Bayes rule
  - Pro: not yet committed to loss during training
  - Con: need to commit to input and output before training; learning conditional distribution is harder than learning decision rule

- **Generative Model**
  - Find \( P(X,Y) \), then derive \( h(x) \) via Bayes rule
  - Pro: not yet committed to loss, input, or output during training; often computationally easy
  - Con: Needs to model dependencies in \( X \)
Logistic Regression

• Data:
  \[ S = ((x_1, y_1), \ldots, (x_n, y_n)), \quad x \in \mathbb{R}^N \text{ and } y \in \{-1, +1\} \]

• Model:
  \[ P(y|x, w) = Ber(y|\text{sigm}(w \cdot x)) \]

• Training objective:
  \[
  \hat{w} = \arg\min_w \sum_{i=1}^n \log(1 + \exp(-y_i w \cdot x_i))
  \]

• Algorithm:
  – Stochastic gradient descent, Newton, etc.
Regularized Logistic Regression

• Data:
  \[ S = ((x_1, y_1) \ldots (x_n, y_n)), \ x \in \mathbb{R}^N \text{ and } y \in \{-1, +1\} \]

• Model:
  \[ P(y|x, w) = Ber(y|\text{sigm}(w \cdot x)), \ P(w) = N(w|0, \Sigma) \]

• Training objective:
  \[
  \hat{w} = \arg\min_w \left\{ \frac{1}{2} w \cdot w + C \sum_{i=1}^{n} \log(1 + \exp(-y_i w \cdot x_i)) \right\}
  \]

• Algorithm:
  – Stochastic gradient descent, Newton, etc.
Softmax vs. Hinge Loss

Plot via www.desmos.com
Ridge Regression

• Data:
  - $S = ((x_1, y_1) \ldots (x_n, y_n))$, $x \in \mathbb{R}^N$ and $y \in \mathbb{R}$

• Model:
  - $P(y|x, w) = N(y|w \cdot x, \Sigma)$, $P(w) = N(w|0, \Sigma)$

• Training objective:
  \[
  \hat{w} = \arg\min_w \frac{1}{2} w \cdot w + C \sum_{i=1}^{n} (w \cdot x_i - y_i)^2
  \]

• Algorithm:
  - $\hat{w} = (\text{diag}(C) + X^T X)^{-1} X^T y$
Discriminative Training of Linear Rules

- Soft-Margin SVM
  - $R(w) = \frac{1}{2}w \cdot w$
  - $\Delta(\tilde{y}, y_i) = \max(0, 1 - y_i \tilde{y})$

- Perceptron
  - $R(w) = 0$
  - $\Delta(\tilde{y}, y_i) = 0$

- Linear Regression
  - $R(w) = 0$
  - $\Delta(\tilde{y}, y_i) = (y_i - \tilde{y})^2$

- Ridge Regression
  - $R(w) = \frac{1}{2}w \cdot w$
  - $\Delta(\tilde{y}, y_i) = (y_i - \tilde{y})^2$

- Lasso
  - $R(w) = \frac{1}{2} \sum |w_i|$
  - $\Delta(\tilde{y}, y_i) = (y_i - \tilde{y})^2$

- Regularized Logistic Regression / Conditional Random Field
  - $R(w) = \frac{1}{2}w \cdot w$
  - $\Delta(\tilde{y}, y_i) = \log(1 + e^{-y_i \tilde{y}})$

\[
\min_{w, b} R(w) + C \frac{1}{n} \sum_{i=1}^{n} \Delta(w \cdot x_i + b, y_i)
\]