Model Selection and Regularization

CS534
General Model Selection Problem

• Assume that we have a set of models $M = \{M_1, M_2, \cdots, M_d\}$ that we are trying to select from. Some examples include:

• **Feature Selection**: each $M_i$ corresponds to using a different feature subset from a large set of potential features

• **Algorithm Selection**: each $M_i$ corresponds to an algorithm, e.g., Naïve Bayes, Logistic Regression, DT ...

• **Parameter selection**: each $M_i$ corresponds to a particular parameter choice, e.g., the choice of kernel and C for SVM
Approaches to Model Selection

- Holdout and Cross-validation methods
  - Experimentally determine when overfitting occurs
- Penalty methods
  - MAP Penalty
  - Minimum Description Length
  - Many others
- Ensembles
  - Instead of choosing, consider many possibilities and let them vote
  - Will talk about this later
Simple Holdout Method

1. Divide training set \( S \) into \( S_{\text{train}} \) and \( S_{\text{valid}} \)
2. Train each model \( M_i \) on \( S_{\text{train}} \) to get a hypothesis \( h_i \)
3. Choose and output \( h_i \) with the smallest error rate on \( S_{\text{valid}} \)

Could retrain the selected model on the whole dataset to get the final hypothesis \( h \) - this will improve the original \( h_i \) because of more training data
Notes on hold-out methods

• Hold-out method often used for choosing among nested hypotheses:
  – Deciding # of training epochs for Neural net
  – Deciding when to stop growing or pruning a decision tree
  – Deciding when to stop growing an ensemble

Example:
Selecting # of epochs for neural net
Issues

• It wastes part of the data
  – The model selection choice is still made using only part of the data
  – Still possible to overfit the validation data since it is a relatively small set of data

• To address these problems, we can use Cross-Validation
K-fold Cross-validation

- Partition (randomly) $S$ into $K$ disjoint subsets $S_1, \cdots, S_K$ (preferably in a class-balanced way)
- To evaluate model $M_j$:
  
  ```
  for i=1:K
    1. Train $M_j$ on $S \setminus S_i$ ($S$ removing $S_i$) $\rightarrow h_{ji}$
    2. Test $h_{ji}$ on $S_i$ $\rightarrow \epsilon_j(i)$
  End for
  
  $\epsilon_j = \frac{1}{K} \sum_i \epsilon_j(i)$
  ```
- Select model that minimizes the error:
  $$M^* = \arg\min_{M_j} \epsilon_j$$
- Train $M^*$ on $S$ and output resulting hypothesis
Comments on k-fold Cross-Validation

• Computationally more expensive than simple hold-out method but better use of data
  – Every data point in the training set is used in validating the model selection choices
• If the data is really scarce, we can use the extreme choice of $k = |S|$
  – Each validation set contains only one data point
  – Often referred to as Leave-one-out (LOO) cross-validation
Feature Selection

• A special case of model selection problem
• Problem: given a supervised learning problem in which the feature dimension is very high, but only a small subset of the features is relevant
• Goal: identify a small subset
• Why?
  – Smaller feature set size leads to less chance of overfitting
  – In some domains, user might like to know which features are important for predicting the target variable
    • E.g. Which set of biomarkers are indicative of higher risk of particular decease
Search Space for Feature Selection

• Given d features, there are $2^d$ possible subsets
• Too expensive to enumerate all possible models to evaluate and choose
• Practical solutions rely on heuristic search
Forward Search

1. Initialize $F = \emptyset$

2. Repeat {
   a) For $i = 1, \ldots, d$ if $i \not\in F$, let $F_i = F \cup \{i\}$, and use cross-validation to evaluate $F_i$
   b) Set $F$ to be the best feature subset found in step a)
}

3. Select the best feature subset that was evaluated during the entire search process
Backward Search

1. Initialize $\mathcal{F} = \{1, \ldots, d\}$
2. Repeat {
   a) For all $i \in \mathcal{F}$, let $\mathcal{F}_{-i} = \mathcal{F}/\{i\}$, and use cross-validation to evaluate $\mathcal{F}_{-i}$
   b) Set $\mathcal{F}$ to be the best feature subset found in step a
}
3. Select the best feature subset that was evaluated during the entire search process
Wrapper vs. Filter approaches

• Both forward and backward search methods are considered wrapper approaches
  – They wrap around a learning algorithm in order to find the subset that works the best with the given learning algorithm

• Alternatively, filter approaches heuristically select the features without considering the learning algorithm
  – Mutual information is one such measure frequently used by filter methods
    • Compute the mutual information between each feature with the class label, and rank them from high to low
    • Choose the top k features in the ranked order
    • How to decide k? Cross-Validation
Penalty (Regularization) Methods

• Basic idea: include a penalty term in the objective function to penalize complex hypothesis

• We have seen examples of this:
  – Regularized linear regression
    \[ J(w) = \sum_i (y_i - w^T x_i)^2 + \lambda |w|^2 \]
  – Regularized logistic regression
    \[ J(w) = \log likelihood(w) - \lambda |w|^2 \]

• A common approach for deriving such regularization method is Maximum A Posterior (MAP) estimation
Bayesian VS Frequentist

• When it comes to parameter estimation, there are two different statistical views
  – Frequentist: parameter is deterministic, it takes an unknown value
  – Bayesian: parameter is a random variable with an unknown distribution

• We can express our belief about the parameter using priors
• After observing the data, we can update our belief to obtain the posterior distribution of the parameter

\[
p(\theta|D) = \frac{p(\theta)p(D|\theta)}{p(D)} = \frac{\int p(D|\theta)p(\theta)d\theta}{\int p(D|\theta)p(\theta)d\theta}
\]
Maximum A Posterior (MAP) as a penalty method

\[ \hat{\theta}_{map} = \arg\max_{\theta} p(\theta|D) \]

\[ = \arg\max_{\theta} p(D|\theta)p(\theta) \]

\[ = \arg\max_{\theta} \log p(D|\theta) + \log p(\theta) \]

penalty
MAP Estimation for Logistic Regression

\[ p(y = 1 | \mathbf{x}; \mathbf{w}) = p_1(\mathbf{x}) = \frac{1}{1 + e^{-w \cdot \mathbf{x}}} \]
\[ p(y = 0 | \mathbf{x}; \mathbf{w}) = 1 - p_1(\mathbf{x}) \]

- \( h \) describes conditional distribution of \( y | \mathbf{x} \)
- Parameters: \( \mathbf{w} \)
- Learning goal is to find \( h \) (i.e. \( \mathbf{w} \)) to maximize \( P(h | S) \):

\[
\arg \max_{\mathbf{w}} P(\mathbf{w}|\mathbf{D}) = \arg \max_{\mathbf{w}} P(\mathbf{D}|\mathbf{w}) P(\mathbf{w}) = \arg \max_{\mathbf{w}} \log P(\mathbf{D}|\mathbf{w}) + \log P(\mathbf{w})
\]

- Our prior belief \( p(\mathbf{w}_i) \): \( N(0, \sigma^2) \) for \( i = 1, \cdots, d \)
  - Large weight values correspond to more complex hypotheses, so this prior prefer simpler hypothesis (\( \mu = 0 \))
Logistic Regression: MAP

\[
\text{argmax}_w \sum_{j} \log p(y_j | x_j, w) + \log \prod_i N(w_i; 0, \sigma^2)
\]

\[
= \text{argmax}_w \sum_{j} \log p(y_j | x_j, w) + \sum_i \log \left( \frac{1}{\sqrt{2\pi\sigma}} \exp \left( -\frac{w_i^2}{2\sigma^2} \right) \right)
\]

\[
= \text{argmax}_w \sum_{j} \log p(y_j | x_j, w) + \sum_i -\frac{w_i^2}{2\sigma^2}
\]

\[
= \text{argmax}_w \sum_{j} \log p(y_j | x_j, w) - \frac{\lambda}{2} \sum_i w_i^2
\]

Old delta:

\[
\nabla L(w) = \sum_{i=1}^{N} (y^i - \hat{y}^i)x^i
\]

\[
\nabla L(w) = \sum_{i=1}^{N} (y^i - \hat{y}^i)x^i - \lambda w
\]
Impact of $\lambda$

- $\lambda$ is inversely proportional to the variance of our prior belief $\lambda = \frac{1}{\sigma^2}$

- Use cross-validation to choose
Summary

• Minimizing training set error will not necessarily minimize test set error – overfitting

• Hold-Out and Cross Validation
  – Empirical methods for estimating the true error
  – Hold-out less computation, but only use part of the data can overfit to validation set
  – LOO is the most accurate estimate one can get, but expensive
  – K-fold cross validation is much more practical

• Penalty method adds a penalty term to the regular objective function
  – MAP estimation is often used to derive penalty methods
  – Often require parameter tuning – use cross validation