EECS 545 → Machine Learning

Lecture 7 → Regularization and Model Selection

- MLE → Maximum Likelihood Estimation
  Objective is to maximize $\log P(\mathcal{D}|w)$
  Example is Linear Regression
- MAP → Maximum a Posteriori (log-likelihood + log prior)
  Objective is to maximize $\log P(\mathcal{D}|w) + \log P(w)$
  Example is Regularized Linear Regression

- MLE discriminative model → $\log P(\mathcal{D}|w) = \log \prod_{n=1}^{N} P(y^{(n)}|\phi(x^{(n)}), w)$

$$= \sum_{n=1}^{N} P(y^{(n)}|\phi(x^{(n)}), w)$$

- MLE generative model → $\log P(\mathcal{D}|w) = \sum_{n=1}^{N} P(y^{(n)}, x^{(n)}|w)$
- Issue with MLE is that we run the risk of overfitting

- MAP assumes prior dist^n $P(w)$
- Point estimate using Bayes rule → $\arg\max_w P(w|\mathcal{D}) = \arg\max_w P(w|\mathcal{D})P(w)$

$$\log P(\mathcal{D}|w)P(w) = \log \prod_{n=1}^{N} P(y^{(n)}|\phi(x^{(n)}), w) + \log P(w)$$

$$= \sum_{n=1}^{N} P(y^{(n)}|\phi(x^{(n)}), w) + \log P(w)$$

- Typically L2-norm is used as prior → $P(w) = \mathcal{N}(0, \lambda^{-1}I)$

$$\log P(w) = -\frac{\lambda}{2}\|w\|^2 + \text{constant}$$

$$P(w) = \mathcal{N}(0, \lambda^{-1}I)$$
$$= \text{const} * \exp\left(-\frac{1}{2} w^T(\lambda^{-1}I)^{-1} w\right)$$
$$= \text{const} * \exp\left(-\frac{\lambda}{2} w^T w\right)$$
$$\log P(w) = \text{Const} - \frac{\lambda}{2} w^T w$$
$$= \text{const} - \frac{\lambda}{2}\|w\|^2$$

- MAP derivation → $\log P(\mathcal{D}|w)P(w) = \sum_{n=1}^{N} \log P(y^{(n)}|\phi(x^{(n)}), w) + \log P(w)$

$$= \sum_{n=1}^{N} \log\left(\sqrt{\frac{\beta}{2\pi}} \exp\left(-\frac{\beta}{2}\|y^{(n)} - w^T\phi(x^{(n)})\|^2\right)\right) + \text{const} - \frac{\lambda}{2}\|w\|^2$$

$$= \frac{N}{2}\log\beta - \frac{N}{2}\log 2\pi - \sum_{n=1}^{N}\frac{\beta}{2}\|y^{(n)} - w^T\phi(x^{(n)})\|^2 + \text{const} - \frac{\lambda}{2}\|w\|^2$$

- Minimize → $\tilde{E}[w] = \sum_{n=1}^{N}\frac{\beta}{2}\|y^{(n)} - w^T\phi(x^{(n)})\|^2 + \frac{\lambda}{2}\|w\|^2 + \text{const}$ wrt w which is minimized by,

$$w_{ML} = (\lambda I + \Phi^T\Phi)^{-1}\Phi^T y$$

- Overfitting → when the test accuracy is significantly higher than the training accuracy due to the model fitting too strongly to the training set and failing to generalize
- Underfitting → when the model fails to capture the relationship between the input and output, leading to high error on both the training set and the unseen data

$E_{RMS} = \sqrt{2E[w^*]/N}$

- L2 regularization controls the tradeoff between fitting error and complexity
- Small L2 regularization results in complex models, but with the risk of overfitting
- Large L2 regularization results in simple models, but with the risk of underfitting

- To avoid overfitting, ① more training data and ② regularization

Bias Variance Tradeoff

- We want to learn a model with,
  - ① Small bias → how well a model fits the data on average?
  - ② Small variance → how stable a model is wrt data samples?

- If we have multiple datasets, each of size N, then any particular dataset, D, will give a particular function $h(x; D)$
- We have,

$$E_D \int \{h(x;D) - E[y|x]\}^2 p(x)\, dx$$

$$= \underbrace{\int \{E_D[h(x;D)] - E[y|x]\}^2 p(x)\, dx}_{\text{bias}^2} + \underbrace{\int E_D[\{h(x;D)] - E_D[h(x;D)]\}^2] p(x)\, dx}_{\text{variance}}$$

- Expected loss → $E[L] = \int\int \{h(x) - y\}^2 p(x,y)\, dx\, dy$
  $= (\text{bias})^2 + \text{variance} + \text{noise}$

where $(\text{bias})^2 = \int \{E_D[h(x;D)] - E[y|x]\}^2 p(x) dx$
$\text{variance} = \int E_D[\{h(x;D)] - E_D[h(x;D)]\}^2] p(x) dx$
$\text{noise} = \int\int \{E[y|x] - y\}^2 p(x,y) dx\, dy$

- Over-regularized model (large λ) will have a high bias and low variance
- Under-regularized model (small λ) will have high variance and low bias

Model Selection

- Hold out cross validation → ① randomly split D into D_{train} and D_{val}
  70%:30% split is typical
  ② train each model M_i on D_{train} to get some hypothesis h_i
  ③ select and output hypothesis h_i that has the smallest error on the holdout validation set
- Disadvantages → ① lose data in D_{val} as we cannot train on this
  ② some data is only used for training, some is only used for validation

- K-fold cross validation → create a k-fold partition of the dataset
  for each of the k experiments, use k-1 folds for training and the remaining one for validating
  true error is estimated as the average error rate
- Special case, $k = N$ → leave one out cross validation (LOOCV)
  - expensive, but wastes least amount of training data for cross validation
  - might be useful when data size is not big
- Popular values of $k = 3, 5, 10$

- Training set → set of examples used for learning
  to fit the parameters to the classifier
- Validation set → set of examples used to tune hyperparameters of a classifier
  use to find the "optimal" hyperparameters
- Test set → set of examples used to assess performance of a fully trained classifier

- Test set is used for evaluation, not for tuning your parameters