CSCC11 Week 3 Notes

Bias - Variance Tradeoff:

- How we train a model:
  1. Collect data
  2. Train a model M
  3. Predict using a trained model

Note: Data collection is a sampling process. It can be sampled from a data generating distribution.

To determine the generalizability of M, we want to see how well M predicts on avg when we sample D from the distribution. We care about the expected squared error shown below.

- $E[(y-\hat{f})^2]$

$= E[(f+\varepsilon-\hat{f})^2]$ Recall: $y = f(x) + \varepsilon$

$= E[(f+\varepsilon-\hat{f}+E[\hat{f}]-E[\hat{f}])^2]$

$= E[(f-E[\hat{f}])^2] + E[\varepsilon^2] + E[(E[\hat{f}]-\hat{f})^2] + 2E[(f-E[\hat{f}])\varepsilon] + 2E[\varepsilon(E[\hat{f}]-\hat{f})] + 2E[(E[\hat{f}]-\hat{f})(f-E[\hat{f}])]$

$= (f-E[\hat{f}])^2 + E[\varepsilon^2] + E[(E[\hat{f}]-\hat{f})^2] + 2(f-E[\hat{f}])E[\varepsilon] + 2E[\varepsilon]E[E[\hat{f}]-\hat{f}] + 2E[E[\hat{f}]-\hat{f}](f-E[\hat{f}])$

$= (f-E[\hat{f}])^2 + E[\varepsilon^2] + E[(E[\hat{f}]-\hat{f})^2]$

$= \text{Bias}[\hat{f}]^2 + \text{Var}[\varepsilon] + \text{Var}[\hat{f}]$

$= \underbrace{\text{Bias}[\hat{f}]^2}_{\text{Bias}} + \underbrace{\sigma^2}_{\text{Baye's Error}} + \underbrace{\text{Var}[\hat{f}]}_{\text{Variance}}$

- Some properties:
  1. Let X be a discrete random variable with a finite number of outcomes $x_1, x_2, \dots, x_k$ occurring with probabilities $p_1, p_2, \dots, p_k$ respectively. The expected value of X, denoted as $E[X]$ is

$$E[x] = \sum_{i=1}^{k} x_i p_i$$

Note: Expected value is another term for the mean.

$$= x_1 p_1 + x_2 p_2 + \dots + x_k p_k$$

  2. Variance is the avg of how much each point differs from the mean.

$$Var[x] = E[x^2] - E[x]^2$$

  3. Standard deviation measures how far apart a group of numbers is from the mean. A low std dev indicates that the values tend to be close to the mean while a high std dev indicates that the values are spread out over a wider range. It is denoted as σ.

$$\sigma = \sqrt{Var[x]}$$

- The bias of a model is the error that comes from the potentially wrong prior assumptions in the model. These assumptions cause the model to miss important info about the relationship btwn the feature and targets for a ML problem.

- The variance of a model is the error that comes from the model's sensitivity to small variations in the training data

- Models with a high bias are often too simple and lead to underfitting.

  Models with a high variance are often too complex and lead to overfitting.

- Baye's Error / Irreducible Error is something we have no control over. It is the error that is introduced from the chosen framing of the problem and may be caused by unknown variables.

Ordinary Least Squares Regression Through Maximum Likelihood:

- Recall: $y = f(x) + \varepsilon$

  $\varepsilon \sim N(0, C)$ where C is the covariance matrix.

Suppose we're dealing with linear models.

$y = w^T x + \varepsilon$

Random Var   Random Var

$y \sim N(w^T x, C)$

The goal is to find w^* s.t. $\underbrace{p(y|w, x)}_{\text{Max likelihood function}}$ is maximized

- Given $\{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$, we have $P(y_1, y_2, \ldots, y_n | w, x_1, \ldots, x_n)$. Further, the data set is an i.i.d, so

Gaussian/Normal Distribution

$$P(y_1, \ldots, y_n | w, x_1, \ldots, x_n) = \prod_{i=1}^{n} P(y_i | w, x_i)$$

$$= \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi |C|^d}} \exp\left(-\frac{1}{2}(y_i - w^T x_i)^T C^{-1} (y_i - w^T x_i)\right)$$

Precison Matrix

Take the log likelihood

$$= \sum_{i=1}^{n} \log(P(y_i | w, x_i))$$

$$= \sum_{i=1}^{n} \log\left(\frac{1}{\sqrt{2\pi |C|^d}}\right) + \frac{-1}{2}(y_i - w^T x_i)^T C^{-1} (y_i - w^T x_i)$$

Classification:

- Supervised Learning: The label/output y is a category.
  - If there's 2 categories, it's called Binary Classification.
  - If there's > 2 categories, it's called Multiclass Classification.
  - With multi-label classification, multiple labels may be assigned to 1 instance.

Note: Binary classification means classifying the elements into 2 groups.

Multi-label classification means classifying the elements into more than 2 groups.

We'll be mainly focused on binary classification.

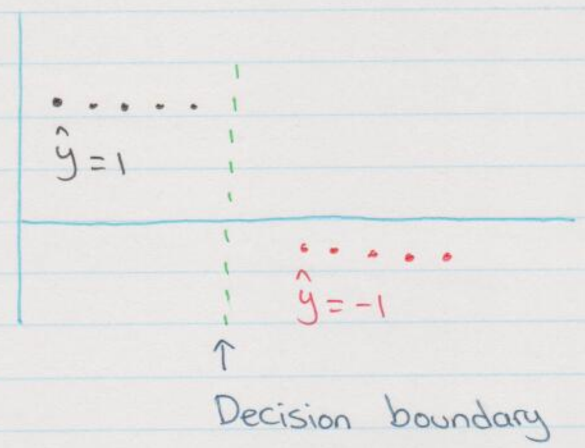
- Let's try linear regression for the first model.

$y = w^T x$

- A reasonable decision rule is:

$$\hat{y} = \begin{cases} 1, & \text{if } f(x) \geq 0 \\ -1, & \text{if otherwise} \end{cases}$$

← This is equivalent to $\hat{y} = \text{sgn}(w^T x)$



- The decision boundary separates the 2 groups. It is also called a hyperplane.

- In 1D, the decision boundary is a threshold.
  In 2D, it is a line.
  In 3D, it is a plane.

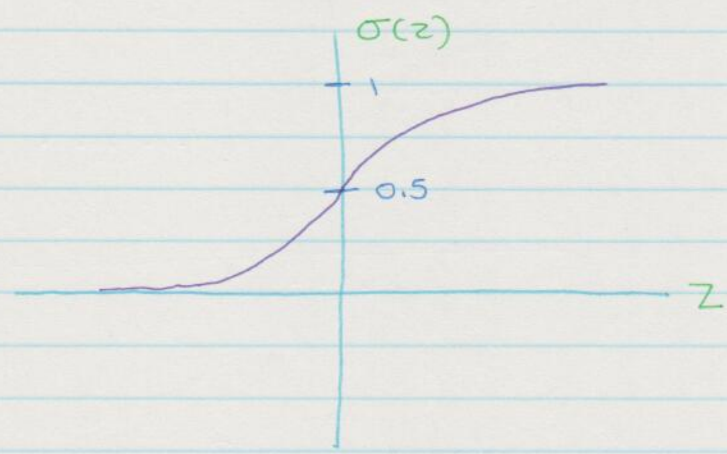
- For our loss function, we can no longer use least squares. We have to use something else.
  Something else we can use is Zero/One Loss function.

$$L_{0\text{-}1}(f(x), y) = \begin{cases} 0, & \text{if } f(x) = y \\ 1, & \text{if } f(x) \neq y \end{cases}$$

However, this isn't great either.

- Instead, we can use this: the sigmoid/logistic function, denoted as σ.

$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$



- Now, let's define the class probability input:

$$P(C_1|x) = \frac{1}{1+\exp(-(w^Tx))}$$
$$= \sigma(w^Tx)$$

$$P(C_2|x) = 1 - \sigma(w^Tx)$$

- The decision boundary is given by:

$$P(C_1|x) = P(C_2|x)$$
$$\Rightarrow \frac{P(C_1|x)}{P(C_2|x)} = 1$$

This is where we're not sure if the point belongs in $P(C_1|x)$ or $P(C_2|x)$.

If $\frac{P(C_1|x)}{P(C_2|x)} > 1$, then choose C_1.

If $\frac{P(C_1|x)}{P(C_2|x)} < 1$, then choose C_2.

- $$\frac{P(C_1|x)}{P(C_2|x)} = \frac{\sigma(w^Tx)}{1-\sigma(w^Tx)}$$

$$= \frac{\frac{1}{1+\exp(-w^Tx)}}{1 - \frac{1}{1+\exp(-w^Tx)}}$$

$$= \frac{\frac{1}{\cancel{1+\exp(-w^Tx)}}}{\frac{\cancel{1}+\exp(-w^Tx)-\cancel{1}}{\cancel{1+\exp(-w^Tx)}}}$$

$$= \frac{1}{\exp(-w^Tx)} = \exp(w^Tx)$$

Continuing from the prev page, I'll take the ln of both sides.

$$\ln\left(\frac{P(C_1|x)}{P(C_2|x)}\right) = w^T x = \underbrace{\ln(1)}_{} = 0$$

$\Rightarrow$ Linear Decision Boundary

This is because we earlier said that our decision boundary is $\frac{P(C_1|x)}{P(C_2|x)} = 1$

- Given $P(C_1|x) = \frac{1}{1+\exp(-w^T x)}$, w is the parameter. Hence, we want to find w^* that gives us the best performance.

- Given the training set $\{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$, we want to model $P(y|x)$ using $P(\underbrace{y_1, \dots, y_N | x_1, \dots, x_N}_{\text{Training set}}, w)$ s.t. the likelihood is maximized.

I.e. $\arg\max_{w} (P(y_1, \dots, y_N | x_1, \dots, x_N, w))$

- Recall: $(x_i, y_i) \overset{iid}{\sim} D$, so we can write $\arg\max_{w} (P(y_1, \dots, y_N | x_1, \dots, x_N, w))$ as

$$\arg\max_{w} \prod_{i=1}^{N} P(y_i | x_i, w)$$

Note: Since $y_i \in \{0,1\}$, we can assume the output follows a Bernoulli Distribution

$$\Rightarrow \arg\max_{w} \prod_{i=1}^{N} P(C_1 | x_i, w)^{y_i} (1-(P(C_1|x_i, w)))^{1-y_i}$$

$$\Rightarrow \arg\max_{w} \prod_{i=1}^{N} P(C=1 \mid X_i, w)^{y_i} \cdot P(C=2 \mid X_i, w)^{1-y_i}$$

Then, if we take the negative log we get:

$$L(w) = \arg\min_{w} -\sum_{i=1}^{N} \left(\begin{array}{l} (y_i \log(P(C=1 \mid X_i, w))) + \\ (1-y_i) \log(P(C=2 \mid X_i, w)) \end{array} \right)$$

$$\Rightarrow \arg\min_{w} -\sum_{i=1}^{N} \left(\begin{array}{l} y_i \log(P(C=1 \mid X_i, w)) + \\ (1-y_i)(\log(1-P(C=1 \mid X_i, w))) \end{array} \right)$$

Now, if we let $P_i = P(C=1 \mid X_i, w)$, we get:

$$L(w) = \arg\min_{w} -\sum_{i=1}^{N} \underbrace{(y_i \log P_i + (1-y_i)(\log(1-P_i)))}_{\text{This term is called Cross-Entropy}}$$

Cross-entropy is a convex function.
Cross-entropy loss has no closed form soln, so we need to use gradient descent.

- Recall: $w_i^{(t+1)} = w_i^{(t)} - \underbrace{\alpha}_{\text{Step size}} \left(\dfrac{\partial L(w)}{\partial w_i} \right)$

  is the formula for gradient descent.

- We can stop the procedure when:
  1. $w^{(t+1)} \approx w^{(t)}$
  2. Reached max iterations
  3. Validation loss is going up

- $$\frac{\partial P_i}{\partial w_i} = \frac{\partial P_i}{\partial \sigma(w^T x_i)} \cdot \frac{\partial \sigma(w^T x_i)}{\partial w_i}$$

$$= \sigma(w^T x_i)(1-\sigma(w^T x_i))$$

- $$\frac{\partial L(w)}{\partial w} = -\sum_{i=1}^{N} \left(y_i (1-P_i) x_i - (1-y_i) P_i x_i \right)$$

$$= -\sum_{i=1}^{N} (y_i - P_i) x_i$$

$$= -\sum_{i=1}^{N} \left(y_i - \sigma(w^T x) \right) x_i$$

- Regularized Logistic Regression:

$$L(w) = -\sum_{i=1}^{N} \left(y_i \log P_i + (1-y_i) \log(1-P_i) \right) +$$

$$\underbrace{\lambda \|w\|_2^2}_{L2} \qquad \text{Note: } \|w\|_2^2 < c$$

- The new gradient descent is:

$$w_i^{(t+1)} = w_i^{(t)} - \alpha \left(\frac{\partial L(w)}{\partial w_i} \right) - \alpha \lambda w_i^{(t)}$$

- How to tune λ:
  1. Partition training data into training set and validation set.
  2. Use training set to learn the weights $(w_1, \ldots, w_n)$.
  3. Use validation set to estimate the tuning parameter. We typically choose the tuning parameter when the model performs the best on the validation set.
  4. Note: Never use test data for tuning the hyper-parameters.

Normal Distribution:

- Also called Gaussian Distribution.
- The general form of its probability density function (pdf) is:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right)$$

- The general notation of a normal distribution is: $N(\mu, \sigma^2)$.

- Recall: On page 3, we said: $y \sim N(w^T x, C)$. Hence, subbing $w^T x$ for μ and C for σ^2, we get:

$$\frac{1}{\sqrt{2\pi|C|^d}} \exp\left(-\frac{1}{2}(y_i - w^T x_i)^T C^{-1} (y_i - w^T x_i)\right)$$

Bernoulli Distribution:

- Can be thought of as a model for the set of possible outcomes of a single experiment that asks a yes/No question.

- PDF: $\begin{cases} q = 1-p, & \text{if } k=0 \\ p, & \text{if } k=1 \end{cases}$

or as

$$p^k (1-p)^{1-k}$$

More Bias-Variance Notes:

- A model with a high bias fails to properly fit the training data. (Underfitting)

- A model with a high variance fits the training data so well, it memorizes it and fails to correctly apply what is has learned to new real-world data. (Overfitting)

- The optimal model has enough bias to avoid memorizing the training data and enough variance to actually fit the patterns in the training data.