Logistic Regression & Linear Classification

CMPUT 296: Basics of Machine Learning

Textbook §10.1-10.4

Logistics

- Thought Questions #3 **due today at 11:59pm**
  - Thought Questions #4 (chapters 10 and 11) due **Thursday, Nov 26/2020**
- Assignment #3 is available; due **Thursday, Dec 3/2020**
- **No class next week** (Fall Reading Break)

Recap: Bias vs. Variance

- Expected generalization error can be **decomposed** into **bias** and **variance**
  - Using a biased estimator can be better than an unbiased one if it **sufficiently reduces variance**
- Worked example: **linear regression**
  - **MLE estimator** is **unbiased** but can have **high variance**
  - **MAP estimator** is **biased** but has a **controllable maximum variance**
- This same principle applies to the choice of **hypothesis class**
  - Bigger hypothesis class can be less biased, but higher variance
- In all cases, exploiting **prior knowledge** is the key to controlling bias vs. variance

Recap:
Why is variance smaller than σ^2/λ?

Recall that

$$\mathrm{Var}\left(w_{\mathrm{MAP}}\right) = \sigma^2 \mathbb{E}\left[\frac{\sum_{i=1}^n X_i^2}{\left(\lambda + \sum_{i=1}^n X_i^2\right)^2}\right]$$

$$= \sigma^2 \mathbb{E}\left[\frac{S_n}{\lambda^2 + 2\lambda S_n + S_n^2}\right]$$

I claim that $\mathrm{Var}(w_{\mathrm{MAP}}) < \sigma^2/\lambda$

- If $S_n > \lambda$, then

$$\frac{S_n}{S_n^2 + z} < \frac{S_n}{S_n^2} = \frac{1}{S_n} < \frac{1}{\lambda}$$

- If $S_n < \lambda$, then

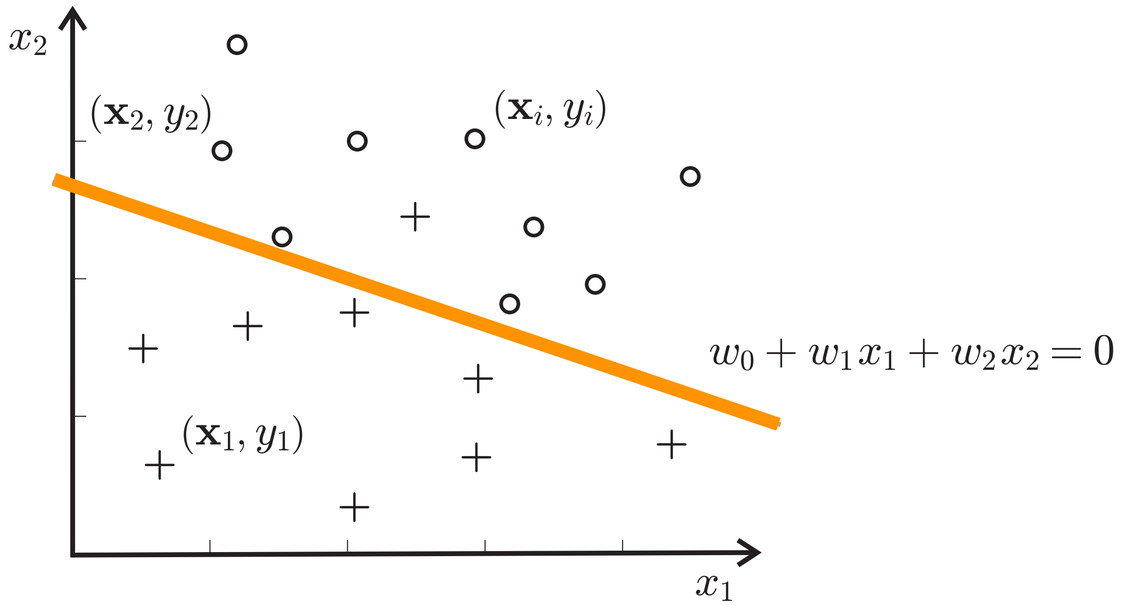
$$\frac{S_n}{\lambda^2 + z} < \frac{\lambda}{\lambda^2 + z} < \frac{\lambda}{\lambda^2} = \frac{1}{\lambda}$$

Outline

1. Recap & Logistics
2. Linear Classification and Logistic Regression
3. Solving Logistic Regression

Linear (Binary) Classifiers

- We've been using linear models for **regression** for the last several lectures
- You can also use them for **classification**:
  - Parameters $\mathbf{w}$ define a linear **decision boundary**
  - Observations on one side of decision boundary classified positive, other side negative
  - A dataset is **linearly separable** if there exists a linear decision boundary that perfectly classifies it



Learning Linear Classifiers

- Formally, a linear binary classifier is a **predictor** $f : \mathbb{R}^{d+1} \rightarrow \{0,1\}$ where

$$f(\mathbf{x}; \mathbf{w}) = \begin{cases} 1 & \text{if } \mathbf{w}^T\mathbf{x} > 0, \\ 0 & \text{otherwise.} \end{cases}$$

- **Question:** Why 0 instead of 1.7 or something?
- There are two main approaches to learning linear classifiers:
  1. Learn the decision boundary **directly**
  2. Learn a model of $p(y \mid \mathbf{x})$, and then predict 1 when $p(y \mid \mathbf{x}) > 0.5$

Logistic Regression

- **Logistic regression** is a way to model $p(y \mid \mathbf{x})$:

$$p(y = 1 \mid \mathbf{x}, \mathbf{w}) = \sigma(\mathbf{w}^T \mathbf{x})$$

  - where $\sigma : \mathbb{R} \to [0,1]$ is an increasing function that maps everything to $[0,1]$
  - Since $y \in \{0,1\}$, this is a **Bernoulli distribution**
  - Even though σ must be **nonlinear**, the resulting classifier will be **linear** (**why?**)
- **Question:** If we perfectly model $p(y \mid \mathbf{x})$, will our classifier always be correct?
- **Question:** Why is this called logistic **regression**? We are doing **classification**!

Sigmoid Functions

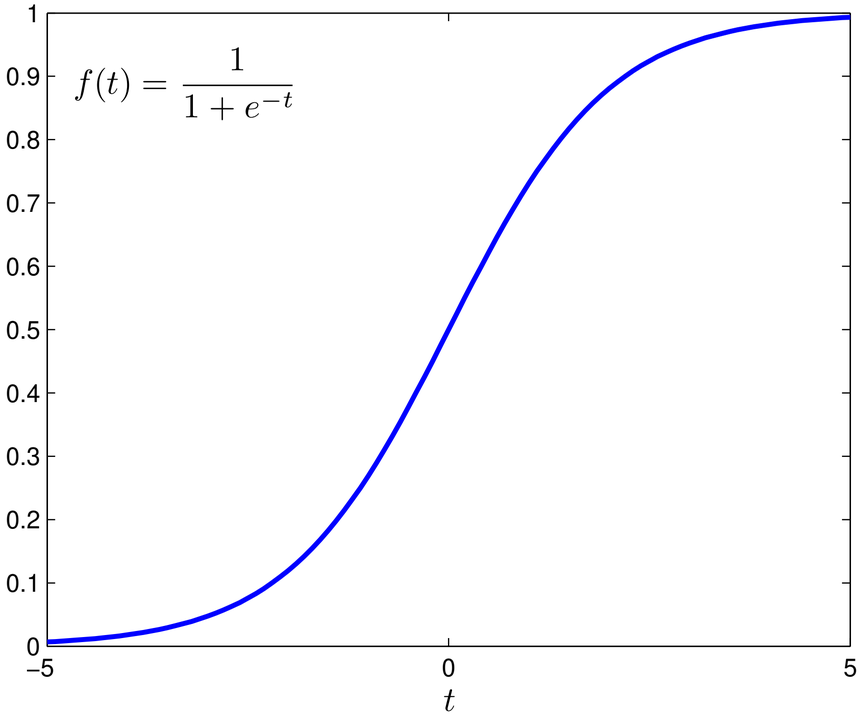
- A **sigmoid** (S-shaped) function is any function $\sigma : \mathbb{R} \to [0,1]$ that is:

  1. **Increasing:** $t_1 > t_2 \implies \sigma(t_1) > \sigma(t_2)$

  2. **Squashing:** $0 < \sigma(t) < 1 \quad \forall t \in \mathbb{R}$

- Logistic regression uses a specific sigmoid called the **logistic function**:

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



Classification with Conditional Probabilities

- **Question:** Suppose $p(y = 1 \mid \mathbf{x}) > 0.5$. Would we ever want to predict $\hat{y} = 0$?
- Suppose we have two classifiers with equivalent accuracies.
  - On our training dataset, they both predict the **same labels** for every datapoint.
  - **Classifier A** is based on a model that predicts $p(y = 1 \mid \mathbf{x}) \in \{0.4, 0.6\}$
  - **Classifier B** is based on a model that predicts $p(y = 1 \mid \mathbf{x}) \in \{0.1, 0.9\}$.
- **Question:** Which classifier is preferable?
- **Question:** How should we train our classifiers to get the better version?

Maximum Likelihood for Classification with Conditional Probabilities

$$
\begin{aligned}
\mathbf{w}_{\text{MLE}} &= \arg \max_{\mathbf{w} \in \mathbb{R}^{d+1}} p(\mathscr{D} \mid \mathbf{w}) \\
&= \arg \max_{\mathbf{w} \in \mathbb{R}^{d+1}} \prod_{i=1}^{n} p(y_i \mid \mathbf{x}_i, \mathbf{w}) \\
&= \arg \max_{\mathbf{w} \in \mathbb{R}^{d+1}} \ln \left(\textstyle\prod_{i=1}^{n} p(y_i \mid \mathbf{x}_i, \mathbf{w}) \right) \\
&= \arg \max_{\mathbf{w} \in \mathbb{R}^{d+1}} \sum_{i=1}^{n} \ln p(y_i \mid \mathbf{x}_i, \mathbf{w}) \\
&= \arg \min_{\mathbf{w} \in \mathbb{R}^{d+1}} \sum_{i=1}^{n} - \ln p(y_i \mid \mathbf{x}_i, \mathbf{w}) \\
&= \arg \min_{\mathbf{w} \in \mathbb{R}^{d+1}} \sum_{i=1}^{n} c_i(\mathbf{w}) \\
&= \arg \min_{\mathbf{w} \in \mathbb{R}^{d+1}} \frac{1}{n} \sum_{i=1}^{n} c_i(\mathbf{w})
\end{aligned}
$$

where

$$c_i(\mathbf{w}) = - \ln p(y_i \mid \mathbf{x}_i, \mathbf{w})$$

Maximum Likelihood for Logistic Regression

$$
\begin{aligned}
c_i(\mathbf{w}) &= -\ln p(y_i \mid \mathbf{x}_i, \mathbf{w}) \\
&= -\ln \left(\sigma(\mathbf{w}^T \mathbf{x}_i)^{y_i} (1 - \sigma(\mathbf{w}^T \mathbf{x}_i))^{1-y_i} \right) \\
&= -\ln \left(\sigma(\mathbf{w}^T \mathbf{x}_i)^{y_i} \right) - \ln \left((1 - \sigma(\mathbf{w}^T \mathbf{x}_i))^{1-y_i} \right) \\
&= -y_i \ln \sigma(\mathbf{w}^T \mathbf{x}_i) - (1 - y_i) \ln (1 - \sigma(\mathbf{w}^T \mathbf{x}_i)) \\
&= -y_i \ln \left(\frac{1}{1 + \exp(-\mathbf{w}^T \mathbf{x}_i)} \right) - (1 - y_i) \ln \left(1 - \frac{1}{1 + \exp(-\mathbf{w}^T \mathbf{x}_i)} \right)
\end{aligned}
$$

Logistic Regression: Gradient

$$c_i(\mathbf{w}) = -y_i \ln \sigma(\mathbf{w}^T \mathbf{x}_i) - (1 - y_i) \ln(1 - \sigma(\mathbf{w}^T \mathbf{x}_i))$$

To find the gradient $\nabla c_i(\mathbf{w})$, find partial derivative of each term for each component of $\mathbf{w}$:

$$\frac{\partial c_i(\mathbf{w})}{\partial w_j} = -\frac{\partial}{\partial w_j} y_i \ln \sigma(\mathbf{w}^T \mathbf{x}) - \frac{\partial}{\partial w_j} (1 - y_i) \ln \left(1 - \sigma(\mathbf{w}^T \mathbf{x})\right)$$

Logistic Regression: Gradient (First Term)

$$\frac{\partial}{\partial w_j} y_i \ln \sigma(\mathbf{w}^T\mathbf{x}) = y_i \frac{\partial}{\partial w_j} \ln \sigma(\mathbf{w}^T\mathbf{x})$$

$$= y_i \frac{\partial \ln \sigma(\mathbf{w}^T\mathbf{x})}{\partial \sigma(\mathbf{w}^T\mathbf{x})} \frac{\partial \sigma(\mathbf{w}^T\mathbf{x})}{\partial \mathbf{w}^T\mathbf{x}} \frac{\mathbf{w}^T\mathbf{x}}{\partial w_j}$$

$$= y_i \frac{1}{\sigma(\mathbf{w}^T\mathbf{x})} \boxed{\frac{\partial \sigma(\mathbf{w}^T\mathbf{x})}{\partial \mathbf{w}^T\mathbf{x}}} \frac{\mathbf{w}^T\mathbf{x}}{\partial w_j}$$

$$= y_i \frac{1}{\sigma(\mathbf{w}^T\mathbf{x})} \sigma(\mathbf{w}^T\mathbf{x}) \left(1 - \sigma(\mathbf{w}^T\mathbf{x})\right) \frac{\mathbf{w}^T\mathbf{x}}{\partial w_j}$$

$$= y_i \boxed{\frac{1}{\sigma(\mathbf{w}^T\mathbf{x})} \sigma(\mathbf{w}^T\mathbf{x})} \left(1 - \sigma(\mathbf{w}^T\mathbf{x})\right) x_{ij}$$

$$= 1$$

$$= y_i \left(1 - \sigma(\mathbf{w}^T\mathbf{x})\right) x_{ij} \quad \blacksquare$$

Exercise:

$$\frac{\partial \sigma(z)}{\partial z} = \sigma(z)\left(1 - \sigma(z)\right)$$

Logistic Regression: Datapoint Gradient

Similarly, second term is

$$\frac{\partial}{\partial w_j}(1-y_i)\ln\left(1-\sigma(\mathbf{w}^T\mathbf{x}_i)\right) = (y_i-1)\sigma(\mathbf{w}^T\mathbf{x}_i)x_{ij}$$

So

$$\frac{\partial}{\partial w_j}c_i(\mathbf{w}) = -\left(y_i\left(1-\sigma(\mathbf{w}^T\mathbf{x}_i)\right)x_{ij} + (y_i-1)\sigma(\mathbf{w}^T\mathbf{x}_i)x_{ij}\right)$$

$$= -\left(y_i x_{ij} - y_i x_{ij}\sigma(\mathbf{w}^T\mathbf{x}) + y_i x_{ij}\sigma(\mathbf{w}^T\mathbf{x}_i) - x_{ij}\sigma(\mathbf{w}^T\mathbf{x}_i)\right)$$

$$= -\left(y_i x_{ij} - x_{ij}\sigma(\mathbf{w}^T\mathbf{x}_i)\right)$$

$$= -\left(x_{ij}(y_i - \sigma(\mathbf{w}^T\mathbf{x}_i))\right) \qquad = x_{ij}\left(\sigma(\mathbf{w}^T\mathbf{x}_i) - y_i\right) \quad \blacksquare$$

Logistic Regression: Full Gradient

$$\nabla c(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n} \nabla c_i(\mathbf{w}) \quad = \frac{1}{n}\sum_{i=1}^{n} \begin{bmatrix} \frac{\partial c_i(\mathbf{w})}{\partial w_1} \\ \vdots \\ \frac{\partial c_i(\mathbf{w})}{\partial w_n} \end{bmatrix} \quad = \frac{1}{n}\sum_{i=1}^{n} \begin{bmatrix} x_{i1}\left(\sigma(\mathbf{w}^T\mathbf{x}_i) - y_i\right) \\ \vdots \\ x_{in}\left(\sigma(\mathbf{w}^T\mathbf{x}_i) - y_i\right) \end{bmatrix}$$

$$= \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i\left(\sigma(\mathbf{w}^T\mathbf{x}_i) - y_i\right)$$

Solving Logistic Regression

- Unfortunately, there is **no closed-form solution** to $\nabla c(\mathbf{w}) = 0$
- **Question:** What can we do instead?
- **Gradient descent** with gradient update:

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \frac{\eta_t}{n} \sum_{i=1}^{n} \mathbf{x}_i \left(\sigma(\mathbf{w}^T \mathbf{x}_i) - y_i \right)$$

- **Stochastic gradient descent** with gradient update (for randomly-chosen i):

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \mathbf{x}_i \left(\sigma(\mathbf{w}^T \mathbf{x}_i) - y_i \right)$$

Why not Ordinary Least Squares?

- Instead of the maximum likelihood solution, we could have directly minimized squared error:

$$\arg \min_{\mathbf{w} \in \mathbb{R}^{d+1}} \frac{1}{n} \sum_{i=1}^{n} \left(\sigma(\mathbf{w}^T \mathbf{x}_i) - y_i \right)^2$$

- Our log-likelihood target is **convex**
  (i.e., second derivative is everywhere positive semidefinite)
  - This means that every **local minimum** is also a **global minimum**
- This direct squared-error optimization target is **non-convex**
  - That means that it might have **many local minima**, with no way to tell which is global
  - In fact, it turns out that this target can have **exponentially many** local minima
- This is another example of the benefit of thinking carefully about **which target** to optimize

Summary

- **Linear binary classification:** Learn a linear **decision boundary**
  - All observations on one "side" of boundary are classified as 0, all observations on the other "side" are classified as 1

$$\text{i.e., } f(\mathbf{x}; \mathbf{w}) = \begin{cases} 1 & \text{if } \mathbf{w}^T\mathbf{x} > 0 \\ 0 & \text{if } \mathbf{w}^T\mathbf{x} \leq 0 \end{cases}$$

  - Can learn boundary **directly**, or predict based on **model** $p(y \mid \mathbf{x}, \mathbf{w})$
- **Logistic regression:** Learn a model $p(y = 1 \mid \mathbf{x}, \mathbf{w}) = \sigma(\mathbf{w}^T\mathbf{x})$
  - No closed-form solution for MLE; must learn numerically (e.g., SGD)
  - MLE problem is **convex**; local optimum is also a global optimum
  - Learning decision boundary directly is **non-convex**; can have **exponentially many** local optima