Logistic Regression and MaxEnt

April 9, 2020

- Generative models:

$$\mathbf{Pr}[y \mid \mathbf{x}] = \frac{\mathbf{Pr}[\mathbf{x} \mid y]\mathbf{Pr}[y]}{\mathbf{Pr}[\mathbf{x}]}$$

  - $y]$.
  - Example: Naive Bayes.
- Discriminative models.

  - models $\mathbf{Pr}[y \mid \mathbf{x}]$ directly as $g(\mathbf{x}$
  - Example: Decision tree, Logistic Regre
- Instance-based Learning.
  - Example: kNN classifier.

Figure: Linear Regres

Task

- Input: $(x^{(i)}, y^{(i)})$ pairs $(1 \leq i \leq n)$
- Preprocess: let $\mathbf{x}^{(\mathbf{i})} = \begin{bmatrix} 1 & x^{(i)} \end{bmatrix}^\top$
- Output: The best $\mathbf{w} = \begin{bmatrix} w_0 & w_1 \end{bmatrix}^\top$ such that $\hat{y} = \mathbf{w}^\top \mathbf{x}$ best explains the observations

The cr

-

Find **w** such that ℓ is minimized.

Minimizing a Function

Taylor Series of $f(x)$ at point a

$$f(x) = \sum_{n=0}^{+\infty} \frac{f^{(i)}(a)}{n!}(x-a)^n \quad (1)$$

$$\ldots)^2) \quad (2)$$

- ... a if it is close to x.
- If $f(x)$ has local minimum x^*, then
  - $f'(x^*) = 0$, and
  - $f''(x^*) > 0$.

  Minimum of the local minima is the global minimum if it is smaller than the function values at all the boundary points.
- Intuitively, $f(x)$ is almost $f(a) + \frac{f''(a)}{2}(x-a)^2$ if a is close to x^*.

$$\frac{\partial \ell}{\partial w_j} = \sum_{i=1}^{n} 2\epsilon_i \frac{\partial \epsilon_i}{\partial w_j} = \sum_{i=1}^{n} 2\epsilon_i \frac{\partial \mathbf{w}^\top \mathbf{x}^{(\mathbf{i})}}{\partial w_j}$$

By setting the above to 0, this essentially requires, f

$$\boxed{\sum_{i=1}^{n} \hat{y}^{(i)} x_j^{(i)}} = \boxed{\sum_{i=1}^{n} y^{(i)} x_j^{(i)}}$$

what the model predicts

what the data says

In the simple 1D case, we have only two parameters in $\mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \end{bmatrix}$

$$\sum_{i=1}^{n}(w_0 + w_1 x_1^{(i)})x_0^{(i)} = \sum_{i=1}^{n} y^{(i)} x_0^{(i)}$$

Since $x_0^{(i)} = 1$, they are essentially

$$\sum_{i=1}^{n}(w_0 + w_1 x_1^{(i)}) \cdot 1 =$$

$$\sum_{i=1}^{n}(w_0 + w_1 x_1^{(i)}) \cdot x_1^{(i)} = \sum_{i=1}^{n} y^{(i)} \cdot x_1^{(i)}$$

Example

Using the same example in https://en.wikipedia.org/wiki/Linear_least_squares_(mathematics)

$$\mathbf{X} = \begin{bmatrix} — & (x^{(1)})^\top & — \\ — & (x^{(2)}) & — \\ — & & — \\ — & & — \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ & \\ & \\ & \end{bmatrix} \quad \begin{bmatrix} 6 \\ 5 \\ 7 \\ 10 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 \\ 1 & 2 \\ 1 & 3 \\ 1 & 4 \end{bmatrix} \quad \begin{bmatrix} 6 \\ 5 \\ 7 \\ 10 \end{bmatrix} = \begin{bmatrix} [illegible] \\ [illegible] \\ 1 & 3 \\ 1 & 4 \end{bmatrix} \quad \begin{bmatrix} [illegible] \\ [illegible] \\ \hat{y}_3 \\ \hat{y}_4 \end{bmatrix}$$

- Easily generalizes to more than 2-dim:

$$\begin{bmatrix} 1 & x_1^{(1)} & \dots & x_m^{(1)} \\ 1 & \dots & \dots & \dots \end{bmatrix} \quad w_0 \quad \begin{bmatrix} y^{(1)} \\ \dots \\ y^{(i)} \\ \dots \\ y^{(n)} \end{bmatrix}$$

1

- How to perform polynomial regression for x?
  - $\hat{y} = w_0 + w_1 x + w_2 x^2 \dots + w_m$
  - Let $x_j^{(i)} = (x_1^{(i)})^j \implies$ Polynomi (`http://mathworld.wolfram.com/LeastSquaresFittingPolynomial.html`)

High-level idea:

- Observations (i.e., training data) are noisy
  - $P(y^{(i)} \mid \hat{y}^{(i)}) =$ ... $= f_i(\mathbf{w})$
- Any **w** is possible, but some **w** is most likely.
  -
  -
    - Maximum likelihood estimation (MLE)
    - If we also incorporate some prior on
      Maximum Posterior Estimation (
      Gaussian prior on **w**, this will add
      the objective function.
- Many models and their variants can be deemed as different ways of estimating $P(y^{(i)} \mid \hat{y}^{(i)})$

Find $\mathbf{w}$ such that $\|\mathbf{y} - \mathbf{X}\mathbf{w}\|_2$ is minimized.

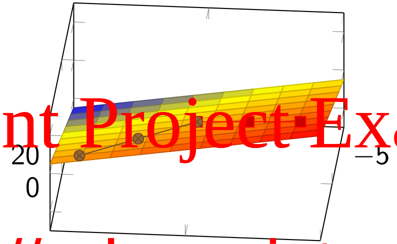
- What is $\mathbf{X}\mathbf{w}$ when $\mathbf{X}$ is fixed?
  - It is the hyperplane spanned by the d column vectors of $\mathbf{X}$.
-

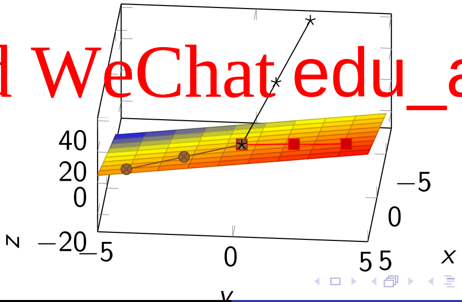
  i-th column of $\mathbf{X}$ as X_i)

$$\left.\begin{aligned} X_1^\top(\mathbf{y} - \mathbf{X}\mathbf{w}) &= 0 \\ X_2^\top(\mathbf{y} - \mathbf{X}\mathbf{w}) &= 0 \\ \ldots\ldots &= 0 \\ X_d^\top(\mathbf{y} - \mathbf{X}\mathbf{w}) &= 0 \end{aligned}\right\} \Longrightarrow \mathbf{X}^\top(\mathbf{y} - \mathbf{X}\mathbf{w}) = \mathbf{0}$$

- $\mathbf{w} = (\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y} = \mathbf{X}^+\mathbf{y}$ ($\mathbf{X}^+$: pseudo inverse of $\mathbf{X}$)

Illustration





Special case: $y^{(i)} \in \{0, 1\}$.

- Not appropriate to directly regress $y^{(i)}$.
- Rather, model $y^{(i)}$ as the observed outcome of a Bernoulli trial with an unknown parameter p_i
-
  - $_i$ to $p_{\mathbf{x}}$.
  -
  - $\mathbf{x}$
  - What can we say about $p_{\mathbf{x}+\epsilon}$ w
- Answer: we impose a linear relationship bet
  - What about a simple linear model
    (Note: all points share the same parameter **w**)
  - Problem: mismatch of the domains: vs
  - Solution: mean function / inverse of link function: $g^{-1} : \Re \to \text{params}$

Solution

- Solution: Link function $g(\text{parameters}) \to \Re$

$$g(p) = \text{logit}(p) \triangleq \log \frac{p}{1-p} = \mathbf{w}^\top \mathbf{x} \qquad (3)$$

- [illegible]

$$\frac{[illegible]}{1 + e^{\mathbf{w}^\top \mathbf{x}}} \qquad \frac{[illegible]}{1 + e^{[illegible]\,\mathbf{w}^\top \mathbf{x}}}$$

Where $\sigma(z) = \frac{1}{1+\exp(-z)}$.
Recall that $p_{\mathbf{x}} = \mathbf{E}[y = 1 \mid \mathbf{x}]$.

- Decision boundary is $p \geq 0.5$.
  - Equivalent to whether $\mathbf{w}^\top \mathbf{x} \geq 0$. Hence, LR is a linear classifier.

Learning the Parameter $\mathbf{w}$

- Consider a training data point $\mathbf{x^{(i)}}$.
  - Recall that the conditional probability ($\mathbf{Pr}[y^{(i)} = 1 \mid \mathbf{x^{(i)}}]$) computed by the model is denoted by the shorthand notation p (which is a function of $\mathbf{w}$ and $\mathbf{x^{(i)}}$).
  - The likelihood of $\mathbf{x^{(i)}}$ is p , if $y^{(i)} = 1$, or equivalently,

-

$$L(\mathbf{w}) = \prod_{i=1}^{n} p(\mathbf{x^{(i)}})^{y^{(i)}} ($$

- Log-likelihood is (assume $\log \triangleq \ln$)

$$\ell(\mathbf{w}) = \sum_{i=1}^{n} y^{(i)} \log p(\mathbf{x^{(i)}}) + (1 - y^{(i)}) \log (1 - p(\mathbf{x^{(i)}})) \quad (5)$$

- To maximize ℓ, notice that it is concave. So take its partial derivatives

$$\frac{\partial \ell(\mathbf{w})}{\partial \mathbf{w}_j} = \sum_{i=1}^{n} \left(y^{(i)} \frac{1}{p(\mathbf{x^{(i)}})} \frac{\partial p(\mathbf{x^{(i)}})}{\partial \mathbf{w}_j} + (1 - y^{(i)}) \frac{1}{1 - p(\mathbf{x^{(i)}})} \frac{\partial (1 - p(\mathbf{x^{(i)}}))}{\partial \mathbf{w}_j} \right)$$

- and set them to 0 essentially means, for all

$$\sum_{i=1}^{n} \hat{y}^{(i)} \cdot \mathbf{x^{(i)}}_j = \boxed{\sum_{i=1}^{n} p(\mathbf{x^{(i)}})\mathbf{x^{(i)}}_j} = \boxed{\sum_{i=1}^{n} y^{(i)} \cdot \mathbf{x^{(i)}}_j}$$

what the model predicts

what the data says

- Consider one dimensional $\mathbf{x}$. The above condition is simplified to

$$n \quad (i) \ (i) \qquad n \quad (i) \ (i)$$

-

  training data in class $Y = 1$.

- The LHS says: if we use our learned model to ass probability (of belonging to the class training data, the LHS is the expected sum of

- If this is still abstract, think of an example.

- **w** is intialized to some random value (e.g., **0**).
- Since the gradient gives the *steepest* direction to increase a function's value, we move a small step towards that direction, i.e.,

$$i=1$$

where α (*learning rate*) is usually a sma

decreasing over the epochs.

- Stochastic version: using the gradient on a randomly selected training instance, i.e.,

$$w_j \leftarrow w_j + \alpha(y^{(i)} - p(\mathbf{x^{(i)}}))\mathbf{x^{(i)}}_j$$

Newton's Method

- Gradient Ascent moves to the "right" direction a tiny step a time. Can we find a good step size?

- Consider 1D case: **minimize** $f(x)$ and the current point is a.
  - $f(x) = f(a) + f'(a)(x - a) + \frac{f''(a)}{2}(x - a)^2$ for x near a.
  -

$$\Leftrightarrow \quad f'(a) \cdot 1 + \frac{f''(a)}{2} \cdot 2(x-a) \qquad 0$$

$$\Leftrightarrow \quad x = a - \frac{f'(a)}{f''(a)}$$

- Can be applied to multiple dimension cases too $\Rightarrow$ need to use ∇ (gradient) and Hess (Hessian).

- Regularization is another method to deal with overfitting.
  - It is designed to penalize large values of the model parameters.
  - Hence it *encourages* simpler models, which are less likely to overfit.

- Instead of optimizing for $\ell(\mathbf{w})$, we optimize $\ell(\mathbf{w}) + \lambda R(\mathbf{w})$.
  -

    -
    - Grid search: `http://scikit-learn.org/stab` `ml`
    - There are alternative methods.

  - $R(\mathbf{w})$ quantifies the "size" of the model pa choices are:
    - L_2 regularization (Ridge LR) $R(\mathbf{w}) = \|w\|_2^2$
    - L_1 regularization (Lasso LR) $R(\mathbf{w}) = \|w\|_1$
    - L_1 regularization is more likely to result in sparse models.

- LR can be generalized to multiple classes $\Longrightarrow$ MaxEnt.

$$\ldots{}_c^{\top}\mathbf{x})$$

  - Z is the normalization constant.
  - Let $\mathbf{c}^*$ be the last class in C, the
- Derive LR from MaxEnt. How?

- Both belong to *exponential* or *log*

Further Reading

- Andrew Ng's note:
  1.pdf

- Tom Mitchell's book chapter: ht
  //www.cs.cmu.edu/~tom/mlboo