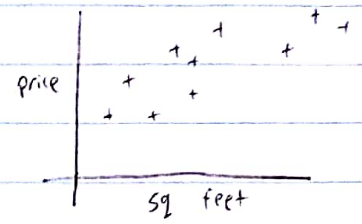
CS299 Lecture Notes 1

Supervised Learning



- input $x^{(i)}$ (also called input features)
- output $y^{(i)}$ (or target variable)
- A pair $(x^{(i)}, y^{(i)})$ is called a training example
- Training set $\{(x^{(i)}, y^{(i)}) : i = 1, \dots, m\}$
- Input and output space $\mathcal{X} = \mathcal{Y} (= \mathbb{R}$ in our case)
- We wish to learn a function $h : \mathcal{X} \to \mathcal{Y}$ so that h is a good predictor

Linear Regression

$$h(x) = \sum_{i=0}^{n} \theta_i x_i = \theta^T x$$

- by convention, $x_0 = 1$
- Given a training set, how do we pick h? i.e., the params θ
- We can try to make $h(x)$ as close as possible to y, at least for the training examples that we have. We define the cost function

$$J(\theta) = \frac{1}{2} \sum_{i=1}^{m} (h(x^{(i)}) - y^{(i)})^2$$
$$= \frac{1}{2} \| X\theta - y^{(i)} \|_2^2$$

where $X = \begin{pmatrix} - x^{(i)T} - \end{pmatrix}$

LMS Algorithm

- We can minimize $J(\theta)$ by gradient descent

m training examples
n features (n+1 if including the intercept x_0)

- start w/ some θ, and repeatedly update by:

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

For a single example:

$$\frac{\partial}{\partial \theta_j} J(\theta) = \frac{\partial}{\partial \theta_j} \frac{1}{2} \sum_{i=1}^{m} (h(x^{(i)}) - y^{(i)})^2 = \boxed{(h_\theta(x) - y)x_j}$$

for the whole training set:

$$\frac{\partial}{\partial \theta} J(\theta) = \frac{\partial}{\partial \theta} \frac{1}{2} \| A\theta - y \|^2$$

$$= 2 \cdot \frac{1}{2} A^T (A\theta - y)$$

$$= A^T A \theta - A^T y \overset{!}{=} 0$$

$$\boxed{\theta = (A^T A)^{-1} A^T y}$$

- this is called the LMS update rule, or Widrow-Hoff.
- update is proportional to the error term $(y^{(i)} - h_\theta(x^{(i)}))$
- Batch gradient descent: use whole dataset for each update
- Stochastic gradient descent: update parameters w.r.t a subset of examples.

- Matrix derivatives

$$\nabla_A \operatorname{tr}(AB) = B^T \qquad (A_{n \times m}, B_{m \times n})$$

$$\nabla_{A^T} f(A) = (\nabla_A f(A))^T$$

$$\nabla_A \operatorname{tr} ABA^T C = CAB + C^T A B^T$$

$$\nabla_A |A| = |A| (A^{-1})^T \qquad |A| \text{ - determinant}$$

Normal Equation

$$X = \begin{bmatrix} \text{---} (x^{(1)})^T \text{---} \\ \vdots \\ \text{---} (x^{(m)})^T \text{---} \end{bmatrix} \in \mathbb{R}^{m \times n}$$

$$y = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(m)} \end{bmatrix}$$

$$J(\theta) = \frac{1}{2}(X\theta - y)^T (X\theta - y) = \frac{1}{2} \| X\theta - y \|^2$$

Probabilistic interpretation

- What are the assumptions under which linear regression is a reasonable choice?
- $$y^{(i)} = \theta^T x^{(i)} + \varepsilon^{(i)}$$
  where $\varepsilon^{(i)}$ is an error term that captures unmodeled effects (i.e., missing features), or random noise.
- Let us further assume that the $\varepsilon^{(i)}$ are distributed IID according to a Gaussian dist. with $(0, \sigma^2)$. We can write
  $$\varepsilon^{(i)} \sim N(0, \sigma^2)$$
  and the density is given by
  $$p(\varepsilon^{(i)}) = \frac{1}{\sqrt{2\pi}\,\sigma} \exp\left(-\frac{(\varepsilon^{(i)})^2}{2\sigma^2}\right)$$
  $$\Rightarrow \quad p(y^{(i)} \mid x^{(i)}; \theta) = \frac{1}{\sqrt{2\pi}\,\sigma} \exp\left(-\frac{(y^{(i)} - \theta^T x^{(i)})^2}{2\sigma^2}\right)$$
- Denote
  $$L(\theta) = L(\theta; X, y) = p(y \mid X; \theta)$$
  that is, given a certain dataset X, we can calculate the likelihood of any prediction y.
- by the IID assumption
  $$L(\theta) = \prod_{i=1}^{m} p(y^{(i)} \mid x^{(i)}; \theta)$$
  $$= \prod \frac{1}{\sqrt{2\pi}\,\sigma} \exp\left(-\left(\frac{y^{(i)} - \theta^T x^{(i)}}{\sqrt{2}\,\sigma}\right)^2\right)$$
- We maximize L, or $\ell = \log(L)$

$$\ell(\theta) = \log L(\theta)$$
$$= \log \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y^{(i)} - \theta^T x^{(i)})^2}{2\sigma^2}\right)$$
$$= \sum_{i=1}^{m} \log \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y^{(i)} - \theta^T x^{(i)})^2}{2\sigma^2}\right)$$
$$= \sum_{i=1}^{m} \log \frac{1}{\sqrt{2\pi}\sigma} + \log \exp\left(-\frac{(y^{(i)} - \theta^T x^{(i)})^2}{2\sigma^2}\right)$$
$$= m \log \frac{1}{\sqrt{2\pi}\sigma} - \sum_{i=1}^{m} \frac{(y^{(i)} - \theta^T x^{(i)})^2}{2\sigma^2}$$

- So, to maximize $\ell(\theta)$, we ~~can~~ have to minimize $\sum_{i=1}^{m} (y^{(i)} - \theta^T x^{(i)})^2$
- So, under the assumption $y^{(i)} = x^{(i)} + \varepsilon^{(i)}$, doing least squares regression corresponds to finding the maximum likelihood estimate of θ.

Locally Weighted Linear Regression

- underfitting : the data has structure not captured by the model
- over fitting : the model has structure not inherent to the data.

- fit θ to minimize $\sum_i w^{(i)} (y^{(i)} - \theta^T x^{(i)})^2$
output $\theta^T x$

where $w^{(i)} = \exp\left(-\frac{(x^{(i)} - x)^2}{2\tau^2}\right)$

which gives higher weight to pts close to the query pt.

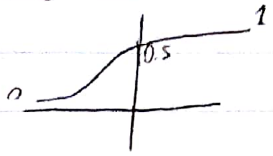
- parametric
- Non-parametric : amount of stuff we need to keep grows

Logistic Regression

- Suppose now that $y^{(i)} \in \{0,1\}$
- No longer makes sense to use linear regression
- Instead, we will choose

$$h_\theta(x) = g(\theta^T x) = \frac{1}{1+e^{-\theta^T x}}$$

$$g(z) = \frac{1}{1+e^{-z}}$$



(called the sigmoid or logistic function)

- $g'(z) = g(z)(1-g(z))$
- The starting assumption (probabilistic) is that

$$p(y=1 \mid x;\theta) = h_\theta(x)$$

$$p(y=0 \mid x;\theta) = \cancel{h_\theta}\ 1 - h_\theta(x)$$

which can be written more compactly as

$$p(y \mid x;\theta) = (h_\theta(x))^y (1-h_\theta(x))^{1-y}$$

- The likelihood is now

$$L(\theta) =$$

$$= p(\vec{y} \mid X;\theta)$$

$$= \prod_{i=1}^{m} (y^{(i)} \mid x^{(i)};\theta)$$

$$= \prod (h_\theta(x^{(i)}))^{y^{(i)}} (1-h_\theta(x^{(i)}))^{1-y^{(i)}}$$

~~max~~ Taking the log likelihood and maximizing it with SGD, we ~~the~~ get the update step:

$$\theta_j = \theta_j + \alpha(y^{(i)} - h_\theta(x^{(i)}))x_j^{(i)}$$

which is suspiciously similar to LMS ... (spoiler: GLM)

Digression: The perceptron Learning Algorithm

- Consider modifying the sigmoid function to "force" it to output 0 or 1.

$$g(z) = \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{if } z < 0 \end{cases}$$

- Using $h_\theta(x) = g(\theta^T x)$ and the update rule

$$\theta_j := \theta_j + \alpha\left(y^{(i)} - h_\theta(x^{(i)})\right) x_j^{(i)}$$

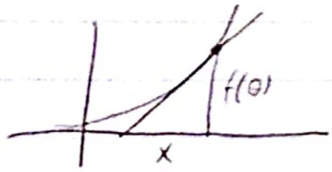
We get what is called the perceptron algorithm.

- In the 60s, this was argued to be a rough model of how a single neuron in the brain worked.
- Good starting pt for theory analysis
- Hard to derive as a max likelihood sort of all estimation alg

Another Algorithm for Maximizing $\ell(\theta)$

- Consider logistic regression again.
- Newton's method uses a linear approx. of a function $f: \mathbb{R} \to \mathbb{R}$ to find a θ s.t. $f(\theta) = 0$

$$\theta := \theta - \frac{f(\theta)}{f'(\theta)}$$



$$\frac{f(\theta)}{x} = f'(\theta)$$

$$x = \frac{f(\theta)}{f'(\theta)}$$

- What if we want to use it to maximize $\ell(\theta)$?
We can try to find $\ell'(\theta) = 0$ using Newton's method:

$$\theta := \theta - \frac{\ell'(\theta)}{\ell''(\theta)}$$

- The generalization of Newton's method to the multidimensional setting is given by

$$\boxed{\theta := \theta - H^{-1} \nabla_\theta \ell(\theta)}$$

Where H is the hessian, $H_{ij} = \frac{\partial^2 \ell(\theta)}{\partial\theta_i \partial\theta_j}$

- Newton's method usually takes fewer iterations to converge, but each iteration is more expensive because of H^{-1}.

Generalized Linear Models

- Both the regression ($y|x;\theta \sim N(\mu, \sigma^2)$) and the classification ($y|x;\theta \sim \text{Bernoulli}(\phi)$) examples are a special case of a GLM.

The Exponential Family

- We start by defining the exponential family of distributions

$$p(y;\eta) = b(y)\exp(\eta^T T(y) - a(\eta))$$

η - natural parameter (also called canonical param.)

$T(y)$ - sufficient statistic (often $T(y) = y$)

$a(\eta)$ - log partition function. $e^{-a(\eta)}$ plays the role of normalization, making sure the dist. sums/integrates to 1. (did he mean $b(y)$?)

T, a, b defines a family of distributions that is parametrized by η.

- The Bernoulli and Gaussian dists. are expo. family dist.
- Bernoulli:

$$\begin{aligned} p(y;\phi) &= \phi^y(1-\phi)^{1-y} \\ &= \exp(\log \phi^y(1-\phi)^{1-y}) \\ &= \exp(y\log\phi + (1-y)\log(1-\phi)) \\ &= \exp(\underbrace{y}_{T(y)}\underbrace{\log\tfrac{\phi}{1-\phi}}_{\eta} + \underbrace{\log(1-\phi)}_{a}) \end{aligned}$$

$$T(y) = y$$

$$\boxed{\phi = \frac{1}{1+e^{-\eta}}}$$

$$\begin{aligned} a(\eta) &= -\log(1-\phi) \\ &= -\log\left(1-\frac{1}{1+e^{-\eta}}\right) = -\log\left(\frac{e^{-\eta}}{1+e^{-\eta}}\right) \\ &= \log\left(\frac{1+e^{-\eta}}{e^{-\eta}}\right) = \log(1+e^{\eta}) \end{aligned}$$

$$b(y) = 1$$

- Gaussian: for convenience, choose $\sigma^2 = 1$ (it had no affect)

$$p(y;\mu) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}(y-\mu)^2\right)$$

$$= \underbrace{\frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}y^2\right)}_{b} \exp\left(\underbrace{\mu}_{\eta}\underbrace{y}_{T} - \underbrace{\frac{1}{2}\mu^2}_{a}\right)$$

$$\eta = \mu$$
$$T(y) = y$$
$$a(\eta) = \frac{1}{2}\mu^2 = \frac{1}{2}\eta^2$$
$$b(y) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}y^2\right)$$

Constructing GLMs

- We want to predict Y given X.
- To derive a GLM, we make the following assumptions:
  1. $Y \mid X; \theta \sim$ Exponential Family (η)
  2. Given X, our goal is to predict the expected value of $T(y)$ (given X). Usually, $T(y) = y$, which means we want h to satisfy $h_\theta(x) = E[Y \mid X]$. For example, in logistic regression $h_\theta(x) = p(y=1 \mid x) = E[y \mid x; \theta]$.
  3. The natural param. and X are related by $\eta = \theta^T x$ (this is more of a design choice)
- These assumptions allow us to derive a very elegant class of learning algs. In particular, we can derive logistic regression and ordinary least squares.

Ordinary Least Squares

- Y is continuous
- We model $Y \mid x; \theta \sim N(\mu, \sigma^2)$
- So, we let the Exponential Family (η) be the normal dist. as we saw, $\eta = \mu$. So, we have:

$$h_\theta(x) = E[y \mid x; \theta] \quad \text{(assumption 2)}$$
$$= \mu$$
$$= \eta = \theta^T x \quad \text{(assumption 1, 3)}$$

Logistic Regression

- $y \in \{0,1\}$
- $\phi = \frac{1}{1+e^{-\eta}}$
- $y|x;\theta \sim \text{Bernoulli}(\phi) \Rightarrow E[y|x;\theta] = \phi$
- We get

$$\begin{aligned} h_\theta(x) &= E[y|x;\theta] \\ &= \phi \\ &= \frac{1}{1+e^{-\eta}} \\ &= \frac{1}{1+e^{-\theta^T x}} \end{aligned}$$

- So, assuming a Bernoulli dist. ~~and an exponential~~ and the GLM assumptions together give rise to the sigmoid hypothesis.

- Canonical response function $g(\eta) = E[T(y);\eta]$
- Canonical inverse function g^{-1}

Softmax Regression

- $y \in \{1, 2, \dots, k\}$
- probability of each class is parametrized by $\phi_1, \dots, \phi_{k-1}$
- $\phi_k = 1 - \sum_{i=1}^{k-1} \phi_i$ (fully specified by $\phi_1, \dots, \phi_{k-1}$

- $p(y=i \mid \phi) = \phi_i$
- To express as an exponential family, define $T(y) \in \mathbb{R}^{k-1}$

$$T(1) = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \dots, \quad T(k-1) = \begin{bmatrix} 0 \\ \vdots \\ 1 \end{bmatrix}, \quad T(k) = 0$$

- Unlike prev examples, $T(y)$ is a vector.
- $(T(y))_i = 1\{y=i\}$
- $E[(T(y))_i] = p(y=i) = \phi$
- The multinomial is a member of the exponential family. We have

$$p(y;\phi) = \phi_1^{1\{y=1\}} \phi_2^{1\{y=2\}} \cdots \phi_k^{1\{y=k\}}$$

$$= \phi_1^{1\{y=1\}} \phi_2^{1\{y=2\}} \cdots \phi_k^{1-\sum_{i=1}^{k-1} 1\{y=i\}}$$

$$= \phi_1^{(T(y))_1} \phi_2^{(T(y))_2} \cdots \phi_k^{1-\sum_{i=1}^{k-1} (T(y))_i}$$

$$= \exp\big((T(y))_1 \log\phi_1 + (T(y))_2 \log\phi_2 + \cdots (1-\textstyle\sum_{i=1}^{k-1} (T(y))_i) \log\phi_k\big)$$

$$= \exp\left((T(y))_1 \log\frac{\phi_1}{\phi_k} + \cdots + (T(y))_{k-1} \log\frac{\phi_{k-1}}{\phi_k} + \log\phi_k\right)$$

$$= b(y)\exp(\eta^T T(y) - a(\eta))$$

where

$$\eta = \begin{bmatrix} \log(\phi_1/\phi_k) \\ \log(\phi_2/\phi_k) \\ \vdots \\ \log(\phi_{k-1}/\phi_k) \end{bmatrix} \quad (\eta_k = 0 \text{ for convenience})$$

$$a(\eta) = -\log\phi_k$$
$$b(y) = 1$$

- This completes our formulation of the multinomial as an exponential family distribution.
- The link function is given by $\eta_i = \log\frac{\phi_i}{\phi_k}$
- To invert the link func:

$$e^{\eta_i} = \frac{\phi_i}{\phi_k}$$

$$\phi_k \sum_{i=1}^{k} e^{\eta_i} = \sum_{i=1}^{k} \phi_i = 1$$

$$\Rightarrow \boxed{\phi_k = \frac{1}{\sum_{i=1}^{k} e^{\eta_i}}}$$

$$\Rightarrow \boxed{\phi_i = \frac{e^{\eta_i}}{\sum_{j=1}^{k} e^{\eta_j}}} \quad \text{softmax function}$$

- by assumption 3, $\eta_i = \theta_i^T x$ for $i = 1, \ldots, k-1$ where $\theta_1, \ldots, \theta_{k-1} \in \mathbb{R}^{n+1}$ are the params of our model
- for convenience, we also define $\theta_k = 0$, so that $\eta_k = \theta_k^T x = 0$

- Hence, our model assumes that the conditional dist. of y given x is given by

$$p(y=i \mid x; \theta) = \phi_i$$

$$= \frac{e^{\eta_i}}{\sum_{j=1}^{k} e^{\eta_j}}$$

$$= \frac{\exp(\theta_i^T x)}{\sum_{j=1}^{k} \exp(\theta_j^T x)}$$

- Our hypothesis will output

$$h_\theta(x) = E[T(y) \mid x; \theta] = E\left[\begin{bmatrix} 1\{y=1\} \\ 1\{y=2\} \\ \vdots \\ 1\{y=k-1\} \end{bmatrix} \middle| x; \theta\right] = \begin{bmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_{k-1} \end{bmatrix} = \begin{bmatrix} \cdots \end{bmatrix}$$

- Finally, we fit the parameters by maximizing

$$\ell(\theta) = \sum_{i=1}^{m} \log p(y^{(i)} \mid x^{(i)}; \theta)$$

$$= \sum_{i=1}^{m} \log \prod_{l=1}^{k} \left(\frac{\exp(\theta_l^T x^{(i)})}{\sum_{j=1}^{k} \exp(\theta_j^T x^{(i)})} \right)^{1\{y^{(i)}=l\}}$$

(using gradient ascent on newton's method)