Linear Regression

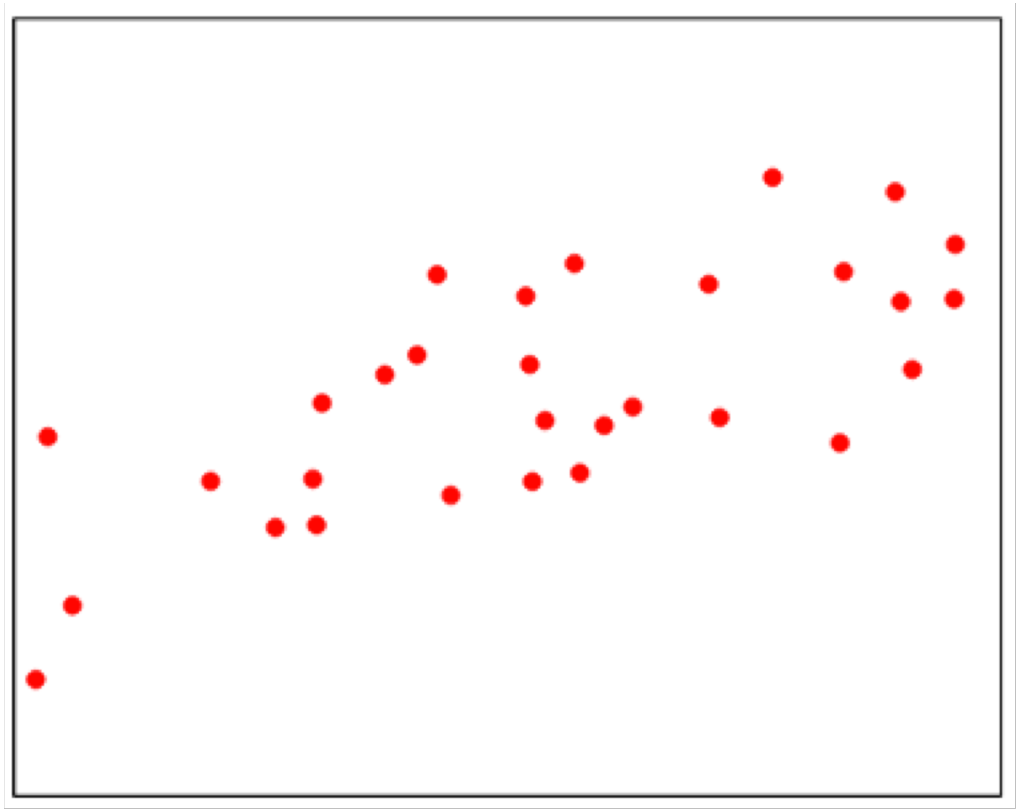
Based on slides by Doina Precup, Andrew Ng

Steps to solving a supervised learning problem

1. Decide what the input-output pairs are
2. Decide how to encode inputs and outputs

   thus deciding the input space and output space
3. Chose a class of hypotheses/mappings $\mathcal{H}$
4. Choose an error function (cost function) to define the best hypothesis
5. Choose an algorithm to search through the space of hypotheses efficiently

Regression



x	y
0.86	2.49
0.09	0.83
-0.85	-0.25
0.87	3.10
-0.44	0.87
-0.43	0.02
-1.10	-0.12
0.40	1.81
-0.96	-0.83
0.17	0.43

What hypothesis class should we pick?

Linear Regression

- y is a linear function of x:

$$h_{\mathbf{w}}(\mathbf{x}) = w_0 + w_1 x_1 (+ \cdots)$$

- or more generally:

$$h_{\mathbf{w}}(\mathbf{x}) = \sum_{i=0}^{n} w_i x_i = \mathbf{w}^T \mathbf{x}$$

Where is the bias?

How do we select the model parameters?

What cost function should we use?

- The cost function measures the difference between the predicted values and the true values (least mean square regression):

$$J(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{m}(h_{\mathbf{w}}(\mathbf{x}_i) - y_i)^2$$

so, choosing w to minimize the mean squared error $J(w)$

Is this familiar?

How can we minimize the cost function?

- Compute its gradient, setting it to zero and solving the equation
- Notations:

- Consider a function $f(u_1, u_2, \ldots, u_n) : \mathbb{R}^n \mapsto \mathbb{R}$ (for us, this will usually be an error function)
- The *partial derivative* w.r.t. u_i is denoted:

$$\frac{\partial}{\partial u_i} f(u_1, u_2, \ldots, u_n) : \mathbb{R}^n \mapsto \mathbb{R}$$

The partial derivative is the derivative along the u_i axis, keeping all other variables fixed.

- The *gradient* $\nabla f(u_1, u_2, \ldots, u_n) : \mathbb{R}^n \mapsto \mathbb{R}^n$ is a function which outputs a vector containing the partial derivatives.
That is:

$$\nabla f = \left\langle \frac{\partial}{\partial u_1} f, \frac{\partial}{\partial u_2} f, \ldots, \frac{\partial}{\partial u_n} f \right\rangle$$

Computing Gradient

$$
\begin{aligned}
\frac{\partial}{\partial w_j} J(\mathbf{w}) &= \frac{\partial}{\partial w_j} \frac{1}{2} \sum_{i=1}^{m} (h_{\mathbf{w}}(\mathbf{x}_i) - y_i)^2 \\
&= \frac{1}{2} \cdot 2 \sum_{i=1}^{m} (h_{\mathbf{w}}(\mathbf{x}_i) - y_i) \frac{\partial}{\partial w_j} (h_{\mathbf{w}}(\mathbf{x}_i) - y_i) \\
&= \sum_{i=1}^{m} (h_{\mathbf{w}}(\mathbf{x}_i) - y_i) \frac{\partial}{\partial w_j} \left(\sum_{l=0}^{n} w_l x_{i,l} - y_i \right) \\
&= \sum_{i=1}^{m} (h_{\mathbf{w}}(\mathbf{x}_i) - y_i) x_{i,j}
\end{aligned}
$$

Matrix Notations

$$X = \begin{pmatrix} \mathbf{x}_1^\top \\ \mathbf{x}_2^\top \\ \vdots \\ \mathbf{x}_n^\top \end{pmatrix} = \begin{pmatrix} 1 & x_{11} & \cdots & x_{1p} \\ 1 & x_{21} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & \cdots & x_{np} \end{pmatrix} \qquad \mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}$$

The solution

- Recalling some multivariate calculus:

$$\begin{aligned}
\nabla_{\mathbf{w}} J &= \nabla_{\mathbf{w}} \frac{1}{2}(\mathbf{X}\mathbf{w} - \mathbf{y})^T(\mathbf{X}\mathbf{w} - \mathbf{y}) \\
&= \nabla_{\mathbf{w}} \frac{1}{2}(\mathbf{w}^T\mathbf{X}^T\mathbf{X}\mathbf{w} - \mathbf{y}^T\mathbf{X}\mathbf{w} - \mathbf{w}^T\mathbf{X}^T\mathbf{y} + \mathbf{y}^T\mathbf{y}) \\
&= \mathbf{X}^T\mathbf{X}\mathbf{w} - \mathbf{X}^T\mathbf{y}
\end{aligned}$$

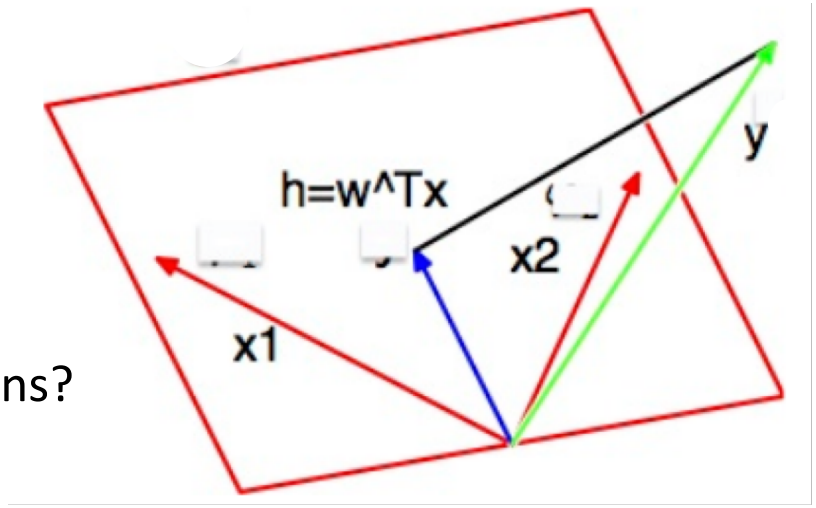
- Setting gradient equal to zero:

$$\begin{aligned}
\mathbf{X}^T\mathbf{X}\mathbf{w} - \mathbf{X}^T\mathbf{y} &= 0 \\
\Rightarrow \mathbf{X}^T\mathbf{X}\mathbf{w} &= \mathbf{X}^T\mathbf{y} \\
\Rightarrow \mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}
\end{aligned}$$

- The inverse exists if the columns of $\mathbf{X}$ are linearly independent.

Properties of the mean squared error

- Good intuition with inductive learning
- Nice math (closed-form solution (very rare in machine learning), unique global optimum)
- Geometric interpretation



Any other interpretations?

Probabilistic Interpretation

- Assume the outputs and inputs are related via:

$$y^{(i)} = \theta^T x^{(i)} + \epsilon^{(i)}$$

error term to capture either unmodeled effects (e.g., missing features), or random noise

- $\epsilon^{(i)}$ are distributed independently and identically according to the Gaussian distribution with mean zero and some variance σ^2: $\epsilon^{(i)} \sim \mathcal{N}(0, \sigma^2)$:

$$p(\epsilon^{(i)}) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(\epsilon^{(i)})^2}{2\sigma^2}\right)$$

- So:

$$p(y^{(i)}|x^{(i)};\theta) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y^{(i)} - \theta^T x^{(i)})^2}{2\sigma^2}\right)$$

Not condition on θ, i.e., $y^{(i)} \mid x^{(i)}; \theta \sim \mathcal{N}(\theta^T x^{(i)}, \sigma^2)$

Probabilistic Interpretation

- Given X and θ, what is the distribution of y?
- The (conditional) likelihood function of data:

$$L(\theta) = L(\theta; X, y) = p(y|X; \theta)$$

- As the noise $\epsilon^{(i)}$ is independent (so as $y^{(i)}$ given $x^{(i)}$):

$$\begin{aligned} L(\theta) &= \prod_{i=1}^{m} p(y^{(i)} \mid x^{(i)}; \theta) \\ &= \prod_{i=1}^{m} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y^{(i)} - \theta^T x^{(i)})^2}{2\sigma^2}\right) \end{aligned}$$

The maximum likelihood principle

- Choosing the parameters to make the data as high probability as possible, i.e., choose θ to maximize $J(\theta)$.
- Taking the log:

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

- So, maximizing $J(\theta)$ gives the same answer as minimizing:

$$\frac{1}{2} \sum_{i=1}^{m} (y^{(i)} - \theta^T x^{(i)})^2$$

- Minimizing $-\log(L(\theta))$ give rises to the so-called "negative log-likelihood"

But does maximum likelihood make sense?

- With inductive bias learning: to learn parameters, minimize

$$\varepsilon(h) = \frac{1}{n}\sum_{i=1}^{n} L(h_w\,(x^{(i)}), y^{(i)}))$$

Using the negative log-likelihood as the loss function:

$$L(h_w(x^{(i)}), y^{(i)})) = -\log\, p(y^{(i)}|x^{(i)}; w)$$

But does maximum likelihood make sense?

- With inductive bias learning: to learn parameters, minimize

$$\varepsilon(h) = \frac{1}{n}\sum_{i=1}^{n} L(h_w\,(x^{(i)}), y^{(i)}))$$

Using the negative log-likelihood as the loss function:

$$L(h_w(x^{(i)}), y^{(i)})) = -\log\, p(y^{(i)}|x^{(i)}; w)$$

- With Bayesian inference

Bayes theorem in learning

Let h be a hypothesis and D be the set of training data.
Using Bayes theorem, we have:

$$P(h|D) = \frac{P(D|h)P(h)}{P(D)},$$

where:

- $P(h)$ is the *prior probability of hypothesis h*
- $P(D) = \int_h P(D|h)P(h)$ is the probability of training data D (normalization, independent of h)
- $P(h|D)$ is the probability of h given D
- $P(D|h)$ is the probability of D given h (*likelihood of the data*)

Choosing hypotheses

- What is the most probable hypothesis given the training data?
- *Maximum a posteriori (MAP)* hypothesis h_{MAP}:

$$
\begin{aligned}
h_{MAP} &= \arg\max_{h \in \mathcal{H}} P(h|D) \\
&= \arg\max_{h \in \mathcal{H}} \frac{P(D|h)P(h)}{P(D)} \text{(using Bayes theorem)} \\
&= \arg\max_{h \in \mathcal{H}} P(D|h)P(h)
\end{aligned}
$$

Last step is because $P(D)$ is independent of h (so constant for the maximization)

- This is the Bayesian answer

Maximum likelihood estimation

$$h_{MAP} = \arg\max_{h \in \mathcal{H}} P(D|h)P(h)$$

- If we assume $P(h_i) = P(h_j)$ (all hypotheses are equally likely a priori) then we can further simplify, and choose the *maximum likelihood (ML) hypothesis*:

$$h_{ML} = \arg\max_{h \in \mathcal{H}} P(D|h) = \arg\max_{h \in \mathcal{H}} L(h)$$

- Standard assumption: the training examples are *independently identically distributed (i.i.d.)*
- This alows us to simplify $P(D|h)$:

$$P(D|h) = \prod_{i=1}^{m} P(\langle \mathbf{x_i}, y_i \rangle | h) = \prod_{i=1}^{m} P(y_i|\mathbf{x}_i; h)P(\mathbf{x}_i)$$

The log trick

- We want to maximize:

$$L(h) = \prod_{i=1}^{m} P(y_i|\mathbf{x}_i; h)P(\mathbf{x}_i)$$

This is a product, and products are hard to maximize!

- Instead, we will maximize $\log L(h)$! (the log-likelihood function)

$$\log L(h) = \sum_{i=1}^{m} \log P(y_i|\mathbf{x}_i; h) + \sum_{i=1}^{m} \log P(\mathbf{x}_i)$$

- The second sum depends on D, but not on h, so it can be ignored in the search for a good hypothesis

Is the analytical solution always possible for linear regression?

$$\mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

- Problems occur if $X^T X$ is not invertible
- Possible solutions:
  - Transform the data (the kernel methods)
    - Apply a transformation of the inputs from X to some other space X', then do linear regression in the transformed space
- Use a different hypothesis class (e.g., non-linear functions)
- Gradient Descent