

30-Aug-2020: Welcome Video, and What is Machine Learning

- Machine learning: algorithms; supervised, unsupervised, reinforcement, recommender. In this course, also will learn best practices.

31-Aug-2020: Supervised Learning, and Unsupervised Learning

- Supervised learning: right answers are given
- Regression: predicts continuous variable output; Classification: predicts discrete values
- Classification can have $1, \dots, N, \dots, \infty$ attributes. E.g. benignness/malignancy based on age, or age and tumor size, etc.
- Unsupervised learning a.k.a. clustering: Right answers aren't given. For example, news that links to different sources for the same topic.
- Cocktail party algorithm: separates two voices in a conversation, with two microphone recordings. Singular value decomposition is key to this algorithm.
- When learning machine learning, use Octave

1-Sep-2020: Model Representation, and Cost Function

- Training set notation: m is number of training examples, x are input examples, and y are the output variables. Together, (x, y) form a training example. Also denoted $(x^{(i)}, y^{(i)})$.
- In a linear regression, $h_{\theta}(x) = \theta_0 + \theta_1 x \equiv h(x)$.
- Cost function is

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^m \left(h_{\theta}(x^{(i)}) - y^{(i)} \right)^2$$

- Want to minimize J w.r.t. θ_0 and θ_1 .

4-Sep-2020: Cost Function, Intuition I&II; Gradient Descent

- Intuition I; Let $\theta_0 = 0$, then $\min_{\theta_1} J(\theta_1)$ is what we want
- Ex: $h_{\theta}(x) = \theta_1 x$ and let $(x, y) = \{(1, 1), (2, 2), (3, 3)\}$.

$$J(\theta_1) = \frac{1}{2m} \sum_{i=1}^m \left(h_{\theta}(x^{(i)}) - y^{(i)} \right)^2$$

$$\rightarrow \text{If } \theta_1 = 0, h_{\theta}(x) \equiv 0$$

$$\begin{aligned} J(0) &= \frac{1}{2 \times 3} (1 + 4 + 9) \\ &= \frac{14}{6} \end{aligned}$$

- $J(\theta_1)$ is parabolic
- We want $\min_{\theta} J(\theta)$; here, $\theta_1 = 1$ satisfies this criterion
- Intuition II; Let θ_0, θ_1 be free in $J(\theta_0, \theta_1)$ and $h_{\theta}(x)$.
- $J(\theta_0, \theta_1)$ is a paraboloid
- Gradient Descent; Use gradient descent to find (θ_0, θ_1) that minimizes $J(\theta_0, \theta_1)$.
- Differing starting guesses can give different local minima.
- Gradient descent algorithm:

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \quad \text{for } j = 1, 2$$

- Simultaneously update θ_0, θ_1 , α is called the learning rate.
- Ex: $\theta_0 = 1, \theta_2 = 2$ and $\theta_j := \theta_j + \sqrt{\theta_0 \theta_1}$.

$$\begin{aligned} \theta_0 &:= \theta_0 + \sqrt{\theta_0 \theta_1} \\ &= 1 + \sqrt{1 \times 2} \\ &= 1 + \sqrt{2} \end{aligned}$$

$$\begin{aligned} \theta_1 &= \theta_2 + \sqrt{\theta_0 \theta_1} \\ &= 2 + \sqrt{1 \times 2} \quad \text{note here that we used the old value of } \theta_0 \\ &= 2 + \sqrt{2} \end{aligned}$$

5-Sep-2020: Gradient Descent Intuition, Gradient Descent for Linear Regression

- Gradient Descent Intuition: For simplicity, assume $\theta_0 = 0$
- One variable: $\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1} J(\theta_1)$; Newton-Raphson
- If α is too small, convergence may be very slow. If too large, it may miss the minimum.
- If θ_1 is already at a local minimum, g.d. leaves θ_1 unchanged since the derivative is zero.
- Gradient Descent for Linear Regression: We need derivatives

$$\frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^m (\theta_0 + \theta_1 x^{(i)} - y^{(i)})$$
$$\frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^m (\theta_0 + \theta_1 x^{(i)} - y^{(i)}) \times x^{(i)}$$

- So, gradient descent finds the new θ variables as

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^m (\theta_0 + \theta_1 x^{(i)} - y^{(i)})$$
$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^m (\theta_0 + \theta_1 x^{(i)} - y^{(i)}) \times x^{(i)}$$

- This is called “batch gradient descent”; batch implies looking at all the training examples. This is represented by the $\sum_{i=1}^m$.
- Quiz Linear Regression with One Variable: 2) $m = \Delta y / \Delta x = (1 - 0.5) / (2 - 1) = 0.5 \implies y = 0.5x + b$; y-intercept is clearly zero since (0,0) is a data point.
- 3) $h_\theta(x)$; $\theta_0 = -1$, $\theta_1 = 2$; $h_\theta(6) = -1 + 2 \times 6 = 11$

9-Sep-2020: Linear Algebra Review

- Matrices and Vectors: Nothing new; in this course, index from 1.
- Addition and Scalar Multiplication: Nothing new
- Matrix Vector Multiplication: Nothing new;
- Ex: Let house sizes be {2104, 1416, 1534, 852}. Let the hypothesis be $h_\theta(x) = -40 + 0.25x$.

$$\begin{bmatrix} 1 & 2104 \\ 1 & 1416 \\ 1 & 1534 \\ 2 & 852 \end{bmatrix} \begin{bmatrix} -40 \\ 0.25 \end{bmatrix} = \begin{bmatrix} -40 \times 1 + 0.25 \times 2104 \\ -40 \times 1 + 0.25 \times 1416 \\ -40 \times 1 + 0.25 \times 1534 \\ -40 \times 1 + 0.25 \times 852 \end{bmatrix} = \begin{bmatrix} h_\theta(2104) \\ h_\theta(1416) \\ h_\theta(1534) \\ h_\theta(852) \end{bmatrix}$$

This essentially says data matrix \times parameters = prediction

- Best to do this with built-in linear algebra function in Octave/Python. You can do it manually in a for-loop, but it'll be really slow.
- Matrix Multiplication: Take the same example. Now we have three hypotheses:

$$h_\theta(x) = -40 + 0.25x$$

$$h_\theta(x) = 200 + 0.1x$$

$$h_\theta(x) = -150 + 0.4x$$

In matrix form, this becomes

$$\begin{bmatrix} 1 & 2104 \\ 1 & 1416 \\ 1 & 1534 \\ 2 & 852 \end{bmatrix} \begin{bmatrix} -40 & 200 & -150 \\ 0.25 & 0.1 & 0.4 \end{bmatrix} = \begin{bmatrix} 486 & 410 & 692 \\ 314 & 342 & 416 \\ 344 & 353 & 464 \\ 173 & 285 & 191 \end{bmatrix}$$

- Matrix Multiplication Properties: Not commutative. $AB \neq BA$. But it's associative. $ABC = (AB)C = A(BC)$.
- Identity matrix is I such that $AI = IA = A$. $I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ in 2D.

- Inverse of A is A^{-1} such that $AA^{-1} = A^{-1}A = I$.
- Transpose of A is A^T . If $B = A^T$, then $B_{ij} = A_{ji}$.
- Quiz: 4) $u = \begin{bmatrix} 3 \\ -5 \\ 4 \end{bmatrix}$, $v = \begin{bmatrix} 1 \\ 2 \\ 5 \end{bmatrix}$, then $u^T v = \begin{bmatrix} 3 & -5 & 4 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 5 \end{bmatrix} = -3 + (-10) + 20 = 13$.

10-Sep-2020: Multiple Features

- Introduce other features: e.g. house price not just a function of square footage; Now, house price vs. sq. footage, age, number of bedrooms, etc.
- n is the number of features, $x_j^{(i)}$ represents the value of the j^{th} feature for the i^{th} training example; $x^{(i)}$ is a vector of all the features.
- Hypothesis: $h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_n x_n$. Let $x_0^{(i)} = 1$. Then, we can write this in matrix form as

$$x = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix}, \quad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_n \end{bmatrix} \implies h_\theta(x) = \theta^T x$$

11-Sep-2020: G.D. for Multiple Variables, G.D. in Practice I - Feature Scaling, G.D. in Practice II - Learning Rate, Features and Polynomial Regression, Normal Equation

- Gradient Descent for Multiple Variables: For $n \geq 1$, gradient descent is

$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^m \left(h_\theta(x^{(i)}) - y^{(i)} \right) x_j^{(i)} \quad \text{for } j = 0, \dots, n.$$

- G.D. in Practice I - Feature Scaling: ensure features have similar scales. E.g.: Houses in the data set have 1-5 bedrooms, and are between 0-2000 sq. ft. Scale these features to the order of 1. So, divide bedrooms by 5 so it's 0-1, and divide square footage by 2000, so it's 0-2.
- Feature should be $-1 \leq x_i \leq 1$.
- Mean renormalization; Subtract off the mean, and then scale. E.g. $x_1 = (\text{sq. footage} - 1000)/2000$ and $x_2 = (\text{bedrooms} - 2)/5$. More formally,

$$x_i \rightarrow \frac{x_i - \mu_i}{s_i} \quad (\text{mean renormalization}),$$

where x_i is the feature, μ_i is the mean value of the i^{th} feature, and s_i is the range, or standard deviation, of the i^{th} feature.

- G.D. in Practice II - Learning Rate: We can plot $J(\theta)$ as a function of iterations, N ; it should be a decreasing function.
- If $J(\theta)$ vs N diverges, you need a smaller learning rate, α .
- If $J(\theta)$ vs N falls, rises, falls, rises, etc., then use a smaller α .
- Features and Polynomial Regression: In the housing example, hypothesis could be $h_\theta(x) = \theta_0 + \theta_1 \times \text{length} + \theta_2 \times \text{depth}$. Maybe you think the relevant figure is $\text{area} = \text{length} \times \text{depth} \equiv x$. The hypothesis is $h_\theta(x) = \theta_0 + \theta_1 \times x$.
- Polynomial regression; e.g.

$$\begin{aligned} &\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 \\ &\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 \end{aligned}$$

where $x_1 = x = \text{area}$, $x_2 = x^2 = \text{area}^2$, $x_3 = x^3 = \text{area}^3$. In polynomial regression, feature scaling becomes very important.

- Don't just have to have integer powers: e.g. $h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^{1/2}$
- Normal Equation: Instead of using gradient descent to find $\min_\theta J$, use normal equation to do it analytically.
- Intuition; in 1-D, if $J(\theta) = a\theta^2 + b\theta + c$, you can find $dJ/d\theta = 0$ to get the extremum. In N -D, set $\partial_{\theta_j} J = 0$ for $j = 1, \dots, N$.

- Say you have m training examples, each with n features. Let

$$\begin{aligned} X_{ij} &= x_j^{(i)} \\ Y_i &= y^{(i)} \\ \theta &= (X^T X)^{-1} X^T Y \end{aligned}$$

- If the training examples are $(x^{(1)}, y^{(1)}), \dots, (x^{(m)}, y^{(m)})$, then

$$x^{(i)} = \begin{bmatrix} x_0^{(i)} \\ \vdots \\ x_n^{(i)} \end{bmatrix}, X = \begin{bmatrix} \vec{x}^{(1)T} \\ \vdots \\ \vec{x}^{(m)T} \end{bmatrix}, Y = \begin{bmatrix} y^{(1)} \\ \vdots \\ y^{(m)} \end{bmatrix}, \theta = (X^T X)^{-1} X^T Y,$$

where $x_0^{(i)} = 1$.

- With normal equation method, features don't have to be scaled.
- Normal equation method is slow if n is very large; Computing $(X^T X)^{-1}$ is costly. Inverting an $N \times N$ matrix costs $O(N^3)$.

12-Sep-2020: Normal Equation and Non-Invertibility

- What if $X^T X$ is singular? Octave's `pinv` (pseudo-inverse) takes care of that
- Causes; redundancy: e.g. area in ft^2 and in m^2 ; too few equations: $m < n$, fewer training examples compared to features, i.e. too few equations, too many unknowns.
- Quiz: 1) Midterm exam average $\mu_1 = (89 + 72 + 94 - 69)/4 = 81$; range is $s_1 = 94 - 69 = 25$, thus $x_1^{(3)} \rightarrow (x^{(3)} - \mu_1)/s_1 = (94 - 81)/25 = 0.52$
- 3) $X = \begin{bmatrix} x_0^{(1)} & \dots & x_3^{(1)} \\ \vdots & & \\ x_0^{(14)} & \dots & x_3^{(14)} \end{bmatrix}$ is 14×4 .

13-Sep-2020: Octave Quiz

- Quiz: 1) A is 3×2 , B is 2×3 . Thus, AB and $A+B^T$ are valid
- 4) u, v are 7×1 . Calculate $u \cdot v$. This can be done via $u^T v$. In Octave, this is `sum(v.*w)` and `v'*w`

17-Sep-2020: Classification, Hypothesis Representation, Decision Boundary, Cost Function, Simplified Cost Function and Gradient Descent

- Classification: $y \in \{0, 1\}$ (binary), $y \in \{0, 1, 2, \dots, N\}$ (multiclass)
- Could fit a linear $h_\theta(x) = \theta^T x$ and classify using a threshold of 0.5. Not good, though. Too sensitive to outliers. Also, $h_\theta(x)$ can be negative.
- Hypothesis Representation:

$$\begin{aligned} h_\theta(x) &= g(\theta^T x); \\ g(z) &= \frac{1}{1 + e^{-z}} \quad \text{sigmoid or logistic function; Fermi-Dirac distribution;} \\ \implies h_\theta(x) &= \frac{1}{1 + \exp(-\theta^T x)}. \end{aligned}$$

- $h_\theta(x)$ represents the probability that $y = 1$ on an input x . E.g. $\begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} 1 \\ \text{tumor size} \end{bmatrix}$. If $g(x) = 70\%$, then there is a 70% chance that the tumor is malignant.
- $h_\theta(x) = P(y = 1 | x; \theta)$ means "the probability that $y = 1$ given x , parametrized by θ "
- Probabilities sum to 1. $P(y = 1 | x; \theta) + P(y = 0 | x; \theta) = 1$.
- Decision Boundary: Can say if $h_\theta(x) \geq 0.5 \implies y = 1$, $h_\theta(x) < 0.5 \implies y = 0$.
- $g(z) \geq 0.5 \implies z \geq 0$. So, $h_\theta(x) = g(\theta^T x) \geq 0.5 \implies \theta^T x \geq 0$. Converse is true for < 0.5 .
- Decision Boundary; Say $g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$. WLOG, let $\theta_1 = \theta_2 = 1$. On an x_2 - x_1 diagram, this parametrizes a straight line. $x_2 = -x_1 + \theta_0$. The decision boundary is the set of points (x_2, x_1) s.t. $h_\theta(x) = 0.5$.

- Example; $\theta_0 = 5, \theta_1 = -1, \theta_2 = 0 \implies h_\theta(x) = g(5 - x_1)$. Decision boundary is implied by $x_1 = 5$. Where is $h \geq 0.5$? $5 - x \geq 0 \implies x_1 \leq 5$. This region corresponds to $y = 1$.
- Non-Linear Decision Boundaries; $g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2)$.

- Cost Function: Training set $\{(x^{(1)}, y^{(1)}), \dots, (x^{(m)}, y^{(m)})\}$, $x^{(i)} = \begin{bmatrix} x_0^{(i)} \\ \vdots \\ x_n^{(i)} \end{bmatrix}$. m training examples, each with n features. $x_0^{(i)} \equiv 1$.

- Recall, for linear regression, $J(\theta) = \frac{1}{m} \sum_{i=1}^m \frac{1}{2} (h_\theta(x^{(i)}) - y^{(i)})^2$. Define

$$\text{cost}(h_\theta(x), y) = \frac{1}{2} (h_\theta(x) - y)^2, \quad \text{for linear regression only}$$

so $J(\theta) = \frac{1}{m} \sum_{i=1}^m \text{cost}(h_\theta(x^{(i)}), y^{(i)})$.

- For log. regression, $J(\theta)$ is not convex, i.e. it has many local minima. Need a new cost function.
- For log. regression, $\text{cost}(h_\theta(x), y) = \begin{cases} -\log(h_\theta(x)) & \text{if } y = 1 \\ -\log(1 - h_\theta(x)) & \text{if } y = 0 \end{cases}$
- If $y = 1$, and $h_\theta(x) = 1$, the cost=0. As $h_\theta(x) \rightarrow 0$, cost $\rightarrow \infty$. Converse is also true for $y = 0$.
- Simplified Cost Function For Gradient Descent: Write out cost function in one equation

$$\text{cost}(h_\theta(x), y) = -y \log(h_\theta(x)) - (1 - y) \log(1 - h_\theta(x))$$

Plug into J .

$$J(\theta) = \frac{1}{m} \sum_{i=1}^m \left(-y^{(i)} \log(h_\theta(x^{(i)})) - (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)})) \right)$$

Comes from “max likelihood estimation.” Then find θ via $\min_\theta J(\theta)$. Use θ to make predictions $h_\theta(x) = 1/(1 + \exp(-\theta^T x))$.

- Use g.d.: $\theta_j := \theta_j - \alpha \partial_{\theta_j} J(\theta)$. Computing the derivatives, we have

$$\theta_j := \theta_j - \alpha \sum_{i=1}^m \left(h_\theta(x^{(i)}) - y^{(i)} \right) x_j^{(i)},$$

which is the same as for linear regression, though h has a different meaning.