Overview

- Linear Regression with One Variable
- Gradient Descent
- Linear Regression with Multiple Variables
- Gradient Descent with Multiple Variables
Example: Advertising Data

- Data taken from An Introduction to Statistical Learning with Applications in R (http://www-bcf.usc.edu/~gareth/ISL/data.html)
- Data consists of the advertising budgets for three media (TV, radio and newspapers) and the overall sales in 200 different markets.

<table>
<thead>
<tr>
<th>TV</th>
<th>Radio</th>
<th>Newspaper</th>
<th>Sales</th>
</tr>
</thead>
<tbody>
<tr>
<td>230.1</td>
<td>37.8</td>
<td>69.2</td>
<td>22.1</td>
</tr>
<tr>
<td>44.5</td>
<td>39.3</td>
<td>45.1</td>
<td>10.4</td>
</tr>
<tr>
<td>17.2</td>
<td>45.9</td>
<td>69.3</td>
<td>9.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
• Suppose we want to predict sales in a new area?
• Predict sales when the TV advertising budget is increased to 350?
• ... Draw a line that fits through the data points
Some Notation

Training data:

<table>
<thead>
<tr>
<th>TV ($x$)</th>
<th>Sales ($y$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>230.1</td>
<td>22.1</td>
</tr>
<tr>
<td>44.5</td>
<td>10.4</td>
</tr>
<tr>
<td>17.2</td>
<td>9.3</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

- $m$ = number of training examples
- $x$ = “input” variable/features
- $y$ = “output” variable/“target” variable
- $(x^{(i)}, y^{(i)})$ the $i$th training example
- $x^{(1)} = 230.1$, $y^{(1)} = 22.1$, $x^{(2)} = 44.5$, $y^{(2)} = 10.4$
Model Representation

- **Prediction:**
  \[ \hat{y} = h_\theta(x) = \theta_0 + \theta_1 x \]

- \( \theta_0, \theta_1 \) are (unknown) parameters

- sometimes abbreviate \( h_\theta(x) \) to \( h(x) \)

\[ \theta_0 = 15, \ \theta_1 = 0 \]

\[ \theta_0 = 0, \ \theta_1 = 0.1 \]

\[ \theta_0 = 15, \ \theta_1 = 0.1 \]
Cost Function: How to choose model parameters $\theta$?

- **Prediction:** $\hat{y} = h_\theta(x) = \theta_0 + \theta_1 x$
- **Idea:** Choose $\theta_0$ and $\theta_1$ so that $h_\theta(x^{(i)})$ is close to $y^{(i)}$ for each of our training examples $(x^{(i)}, y^{(i)}), i = 1, \ldots, m$.
- **Least squares case:** select the values for $\theta_0$ and $\theta_1$ that minimise cost function:

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

• Suppose our training data consists of just two observations: \((3, 1), (2, 1)\), and to keep things simple we know that \(\theta_0 = 0\).
• The cost function is
\[
\frac{1}{2} \sum_{j=1}^{2} (y^{(j)} + \theta_1 x^{(j)})^2 = \frac{1}{2} (1 - 3\theta_1)^2 + (1 - 2\theta_1)^2
\]
• What value of \(\theta_1\) minimises \((1 - 3\theta_1)^2 + (1 - 2\theta_1)^2\)?
Example: Advertising Data
Example: Advertising Data

- Least square linear fit
- Residuals are the difference between the value predicted by the fit and the measured value.
  - Do the residuals look “random” or do they have some “structure”? Is our model satisfactory?
  - We can use the residuals to estimate a confidence interval for the prediction made by our linear fit.
- We could use cross-validation/bootstrapping to estimate out confidence in the fit itself.
Summary

• Hypothesis: $h_\theta(x) = \theta_0 + \theta_1 x$
• Parameters: $\theta_0, \theta_1$
• Cost Function: $J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$
• Goal: Select $\theta_0$ and $\theta_1$ that minimise $J(\theta_0, \theta_1)$
Gradient Descent

Need to select $\theta_0$ and $\theta_1$ that minimise $J(\theta_0, \theta_1)$. Brute force search over pairs of values of $\theta_0$ and $\theta_1$ is inefficient, can we be smarter?

- Start with some $\theta_0$ and $\theta_1$
- Repeat:
  
  Update $\theta_0$ and $\theta_1$ to new value which makes $J(\theta_0, \theta_1)$ smaller

- When curve is “bowl shaped” or convex then this must eventually find the minimum.
Gradient Descent

- Start with some $\theta_0$ and $\theta_1$
- Repeat:
  - Update $\theta_0$ and $\theta_1$ to new value which makes $J(\theta_0, \theta_1)$ smaller
- When curve has several minima then we can’t be sure which we will converge to.
- Might converge to a local minimum, not the global minimum

![3D graphs showing gradient descent](image-url)
Gradient Descent

Repeat: Update $\theta_0$ and $\theta_1$ to new value which makes $J(\theta_0, \theta_1)$ smaller

- One option: carry out local search of $\theta_0$ and $\theta_1$ to find one that decreases $J$.
- Another option: gradient descent:

\[
\text{temp}0 := \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)
\]

\[
\text{temp}1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)
\]

$\theta_0 := \text{temp}0$, $\theta_1 := \text{temp}1$

- $\frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) \approx \frac{J(\theta_0 + \delta, \theta_1) - J(\theta_0, \theta_1)}{\delta}$ for $\delta$ sufficiently small.
- $J(\theta_0 + \delta, \theta_1) \approx J(\theta_0, \theta_1) + \delta \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$
- When $\delta = -\alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$ then
  $J(\theta_0 + \delta, \theta_1) \approx J(\theta_0, \theta_1) - \alpha \left( \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) \right)^2$
Gradient Descent

- Selecting step size $\alpha$ too small will mean it takes a long time to converge to minimum
- But selecting $\alpha$ too large can lead to us overshooting the minimum
- We need to adjust $\alpha$ so that algorithm converges in a reasonable time.
Gradient Descent

For $J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$ with $h_\theta(x) = \theta_0 + \theta_1 x$:

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

So gradient descent algorithm is:

- repeat:

  temp0 := $\theta_0 - \frac{2\alpha}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})$
  temp1 := $\theta_1 - \frac{2\alpha}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})x^{(i)}$

  $\theta_0 :=$ temp0, $\theta_1 :=$ temp1
Linear Algebra Review

It's assumed you know basic linear algebra for this module. There is lots of revision material online e.g.

- https://youtu.be/6AP4lvfKmwg (coursera linear algebra review)
- https://www.khanacademy.org/math/linear-algebra

Basic notation:

- Vector $\mathbf{x} = \begin{bmatrix} 230.1 \\ 37.8 \end{bmatrix}$, element $x_1 = 230.1$
- Matrix $A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$, element $A_{11} = 1$
- Transpose $\mathbf{x}^T = [230.1 \ 37.8]$
- Inner product $\mathbf{x}^T \mathbf{y} = \sum_{i=1}^{n} x_i y_i$ for two vectors with $n$ elements
- Product of a matrix and a vector $A\mathbf{x}$, product of two matrices $AB$. 
Linear Regression with Multiple Variables

Advertising example:

<table>
<thead>
<tr>
<th></th>
<th>TV</th>
<th>Radio</th>
<th>Newspaper</th>
<th>Sales</th>
</tr>
</thead>
<tbody>
<tr>
<td>x₁</td>
<td>230.1</td>
<td>37.8</td>
<td>69.2</td>
<td>22.1</td>
</tr>
<tr>
<td>x₂</td>
<td>44.5</td>
<td>39.3</td>
<td>45.1</td>
<td>10.4</td>
</tr>
<tr>
<td>x₃</td>
<td>17.2</td>
<td>45.9</td>
<td>69.3</td>
<td>9.3</td>
</tr>
</tbody>
</table>

- \( n = \text{number of features} \) (3 in this example)
- \((x^{(i)}, y^{(i)})\) the \(i\)th training example e.g.
  
  \[ x^{(1)} = [230.1, 37.8, 69.2]^T = \begin{bmatrix} 230.1 \\ 37.8 \\ 69.2 \end{bmatrix} \]

- \( x_{j}^{(i)} \) is feature \(j\) in the \(i\)th training example, e.g. \(x_{2}^{(1)} = 37.8\)
Linear Regression with Multiple Variables

Hypothesis: \( h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 \)
e.g. \( h_\theta(x) = 15 + 0.1 \ x_1 - 5 \ x_2 + 10 \ x_3 \)

- For convenience, define \( x_0 = 1 \)
i.e. \( x_0^{(1)} = 1, \ x_0^{(2)} = 1 \) etc

- Feature vector \( x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \)

- Parameter vector \( \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix} \)

- \( h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n = \theta^T x \)
Linear Regression with Multiple Variables

• Hypothesis: \( h_\theta(x) = \theta^T x \) (with \( \theta, x \) now \( n+1 \)-dimensional vectors)

• Cost Function: \( J(\theta_0, \theta_1, \ldots, \theta_n) = J(\theta) = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2 \)

• Goal: Select \( \theta \) that minimises \( J(\theta) \)

As before, can find \( \theta \) using:

• Start with some \( \theta \)

• Repeat:
  Update vector \( \theta \) to new value which makes \( J(\theta) \) smaller

  e.g using gradient descent:

  • Start with some \( \theta \)
  • Repeat:
    for \( j=0 \) to \( n \) \{ tempj := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta) \}
    for \( j=0 \) to \( n \) \{ \theta_j := tempj \}
Gradient Descent with Multiple Variables

For \( J(\theta) = \sum_{i=1}^{m} (h_{\theta}(x^{(i)}) - y^{(i)})^2 \) with \( h_{\theta}(x) = \theta_0 + \theta_1 x_1 + \cdots + \theta_n x_n \): 

- \( \frac{\partial}{\partial \theta_0} J(\theta) = \frac{2}{m} \sum_{i=1}^{m} (h_{\theta}(x^{(i)}) - y^{(i)}) \)
- \( \frac{\partial}{\partial \theta_1} J(\theta) = \frac{2}{m} \sum_{i=1}^{m} (h_{\theta}(x^{(i)}) - y^{(i)}) x_1^{(i)} \)
- \( \frac{\partial}{\partial \theta_j} J(\theta) = \frac{2}{m} \sum_{i=1}^{m} (h_{\theta}(x^{(i)}) - y^{(i)}) x_j^{(i)} \)

So gradient descent algorithm is:

- Start with some \( \theta \)
- Repeat:
  
  for \( j=0 \) to \( n \) \{ \text{temp}_j := \theta_j - \frac{2\alpha}{m} \sum_{i=1}^{m} (h_{\theta}(x^{(i)}) - y^{(i)}) x_j^{(i)} \} 
  
  for \( j=0 \) to \( n \) \{ \theta_j := \text{temp}_j \}
Example: Advertising Data

- How is the impact of the advertising spend on TV and radio related, if at all?
- Perhaps a quadratic fit would be better? If so, what does that imply for how we allocate our advertising budget?