» Examples of Predicting A Number

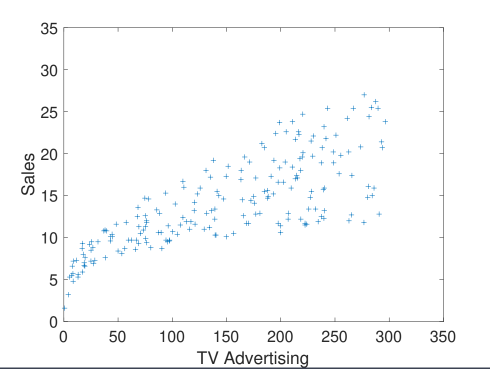
* Predict price of a house given its location, floor area, #rooms etc
* Predict income of a person given their age, gender, handset model, web pages browsed
* Predict temperature outside tomorrow given weather forecast and past measurements at your location
* Predict whether distance between mobile handsets given Bluetooth recieved signal strength, handset models, type of location (house, bus, office, supermarket etc), whether handsets in a pocket/bag or not

» Example: Advertising Data

- Data https://www.statlearning.com/s/Advertising.csv
- Data consists of the advertising budgets for three media (TV, radio and newspapers) and the overall sales in 200 different markets.

TV	Radio	Newspaper	Sales
230.1	37.8	69.2	22.1
44.5	39.3	45.1	10.4
17.2	45.9	69.3	9.3
⋮	⋮	⋮	⋮

» Example: Advertising Data



* 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

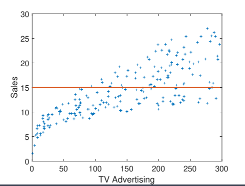
Training data:

TV (x)	Sales (y)
230.1	22.1
44.5	10.4
17.2	9.3
$\vdots$	$\vdots$

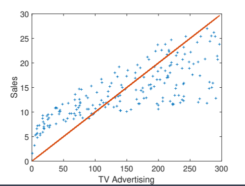
* m=number of training examples
* x="input" variable/features
* y="output" variable/"target" variable
* $(x^{(i)}, y^{(i)})$ the ith training example
* $x^{(1)} = 230.1$, $y^{(1)} = 22.1$, $x^{(2)} = 44.5$, $y^{(2)} = 10.4$

» Model

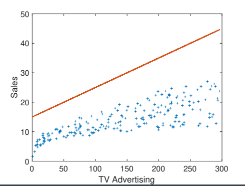
- Prediction: $\hat{y} = h_\theta(x) = \theta_0 + \theta_1 x$
- θ_0, θ_1 are (unknown) parameters
- often abbreviate $h_\theta(x)$ to $h(x)$



$\theta_0 = 15, \theta_1 = 0$



$\theta_0 = 0, \theta_1 = 0.1$



$\theta_0 = 15, \theta_1 = 0.1$

» Cost Function: How to choose model parameters θ?

* Prediction: $\hat{y} = h_\theta(x) = \theta_0 + \theta_1 x$
* Idea: Choose θ_0 and θ_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 θ_0 and θ_1 that minimise cost function:

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

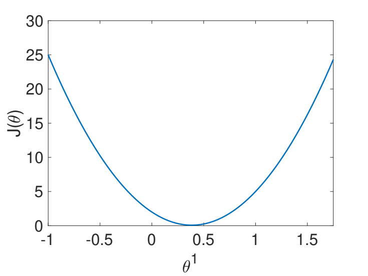
Note: cost function is a sum over prediction error at each training point so can also write as

$$J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^{m} l_i(\theta_0, \theta_1)$$

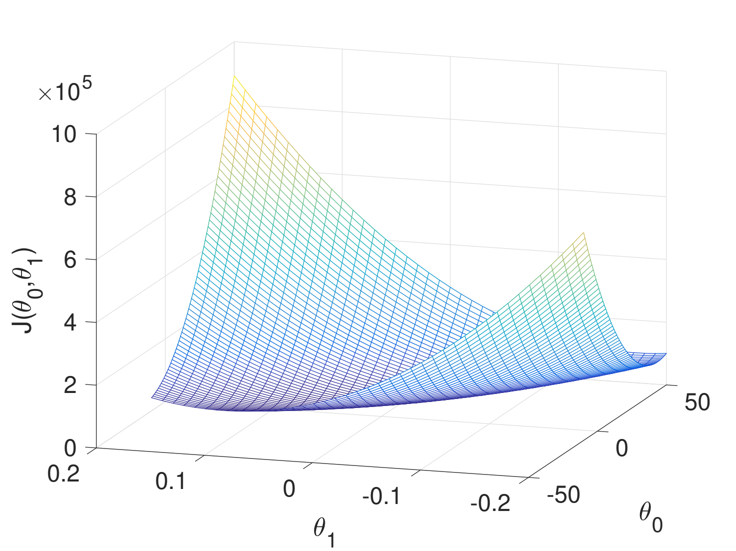
where $l_i(\theta_0, \theta_1) = (h_\theta(x^{(i)}) - y^{(i)})^2$.

» 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 θ_1 minimises $(1 - 3\theta_1)^2 + (1 - 2\theta_1)^2$?

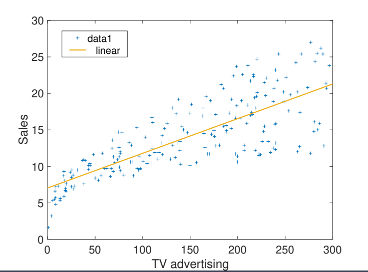


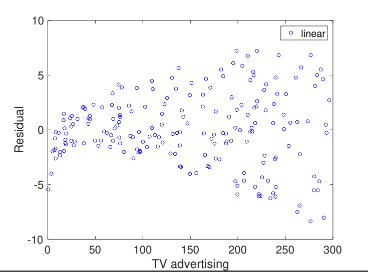
» Example: Advertising Data



» 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.





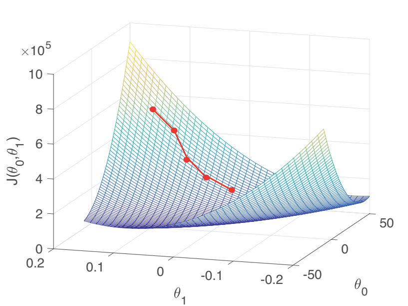
» Summary: Linear Regression With One Feature

* Feature: x
* Linear Model: $h_\theta(x) = \theta_0 + \theta_1 x$
* Parameters: θ_0, θ_1
* Cost Function: $J(\theta_0, \theta_1) = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$
* Optimisation: Select θ_0 and θ_1 that minimise $J(\theta_0, \theta_1)$ *"least squares" – why?*

» Gradient Descent

Need to select θ_0 and θ_1 that minimise $J(\theta_0, \theta_1)$. Brute force search over pairs of values of θ_0 and θ_1 is inefficient, can we be smarter ?

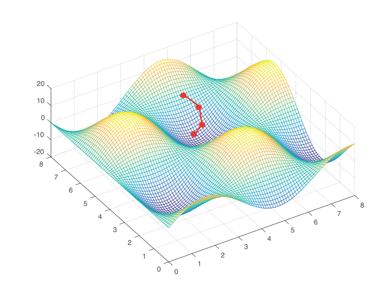
* Start with some θ_0 and θ_1
* Repeat:
  Update θ_0 and θ_1 to new value which makes $J(\theta_0, \theta_1)$ smaller

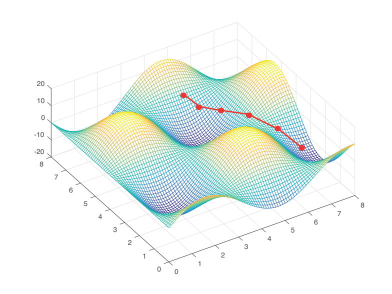


* When curve is "bowl shaped" or convex then this must eventually find the minimum.

» Gradient Descent

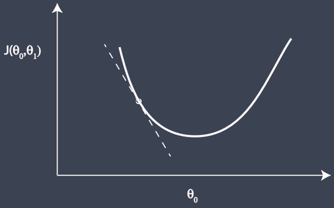
* Start with some θ_0 and θ_1
* Repeat:
  Update θ_0 and θ_1 to new value which makes $J(\theta_0, \theta_1)$ smaller
* When curve has several minima then we can't be sure which we will converge to.
* Might converge to a local minimum, not the global minimum



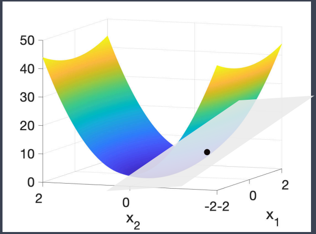


» Derivatives

- When x is scalar the *derivative* of a function $f(\cdot)$ at point x is the *slope* of the line that just touches the function at x e.g.



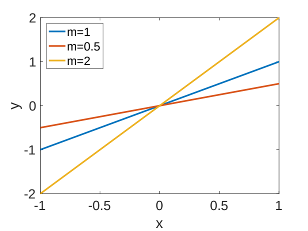
- When $x = [x_1, x_2]$ then the derivative is the *slope* of the plane that just touches the function at x e.g.

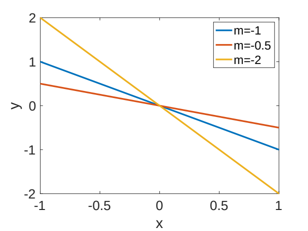


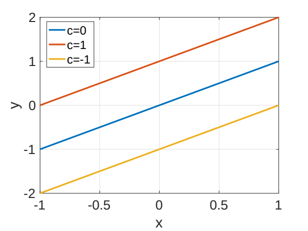
- And similarly when x has > 2 elements, but can't draw it.

» Derivatives

* Equation of a line is $y = mx + c$. m is slope, c is intercept (when $x = 0$ then $y = c$).

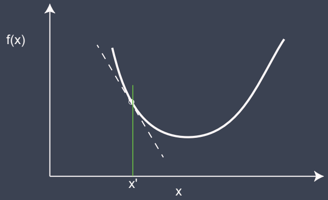






» Derivatives

- *Derivative* of a function $f(\cdot)$ at point x' is the *slope* of the line that just touches the function at x'



- Equation of a line is $mx + c$, slope m and intercept c.
- Line touches $f(\cdot)$ at point x', so $mx' + c = f(x')$ i.e. $c = f(x') - mx'$
- Slope m of line is derivative, i.e. $m = \frac{df}{dx}(x')$ (this is just notation[1], the important point is that we can calculate $\frac{df}{dx}(x')$ and so m using standard tools).
- Putting this together, equation of line is

$$\frac{df}{dx}(x')(x - x') + f(x')$$

and so

$$f(x) \approx f(x') + \frac{df}{dx}(x')(x - x')$$

[1] sometimes $f'(x)$ is used instead of $\frac{df}{dx}(x)$

» Derivatives

- Equation of a plane is $y = m_1 x_1 + m_2 x_2 + c$
- Note: we now have two slopes m_1, m_2 and intercept c
- Notation: For plane just touching $f(\cdot)$ at point x'
  - $m_1 = \frac{\partial f}{\partial x_1}(x')$, $m_2 = \frac{\partial f}{\partial x_2}(x')$
  - $\frac{\partial f}{\partial x_1}(x')$ is the *partial derivative* of $f(\cdot)$ wrt x_1 at point x'
  - $\frac{\partial f}{\partial x_2}(x')$ is the *partial derivative* of $f(\cdot)$ wrt x_2 at point x'
  - $\nabla f(x') = [\frac{\partial f}{\partial x_1}(x'), \frac{\partial f}{\partial x_2}(x'), \dots, \frac{\partial f}{\partial x_n}(x')]$, the vector of partial derivatives. And sometimes $\nabla_{x_1} f(x')$ is used for $\frac{\partial f}{\partial x_1}(x')$ etc.
- This plane is an approximation to function $f(\cdot)$ near point x' i.e.

$$f(x) \approx f(x') + \frac{\partial f}{\partial x_1}(x')(x_1 - x'_1) + \frac{\partial f}{\partial x_2}(x')(x_2 - x'_2)$$

- If we choose $x_1 = x' - \alpha \frac{\partial f}{\partial x_1}$ and $x_2 = x' - \alpha \frac{\partial f}{\partial x_2}$ then moving from point x' to x tends to decrease function $f(\cdot)$ i.e. $f(x) \lessapprox f(x')$

» Gradient Descent

Repeat:

$$\delta_0 := -\alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1),\ \delta_1 := -\alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$$

$$\theta_0 := \theta_0 + \delta_0,\ \theta_1 := \theta_1 + \delta_1$$

α is called the *step size* or *learning rate*, its value needs to be selected appropriately (not too large, not too small).

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:

$$\delta_0 := -\frac{2\alpha}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})$$
$$\delta_1 := -\frac{2\alpha}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x^{(i)}$$
$$\theta_0 := \theta_0 + \delta_0,\ \theta_1 := \theta_1 + \delta_1$$

» Practicalities: Normalising Data

- When using gradient descent (and also more generally) its a good idea to *normalise* your data i.e. scale and shift the inputs and outputs so that they lie roughly between 0 $\rightarrow$1 or -1 $\rightarrow$ 1
- Its ok if data range spans 1 $\rightarrow$ 100, problem is when range is v large e.g 1 $\rightarrow 10^6 \rightarrow$ large ranges (i) mess up numerics, (ii) larger valued data tends to dominate cost function and training focusses on that data

	Age	Income
Person 1	18	5,000 €
Person 2	20	25,000 €
Person 3	25	40,000 €
Person 4	30	50,000 €
Person 5	30	55,000 €
Person 6	28	60,000 €
Person 7	46	65,000 €
Person 8	58	70,000 €
Person 9	62	80,000 €
Person 10	61	100,000 €
Person 11	19	150,000 €

Min	18	5000
Max	62	150000
Mean	36.09	63636
Std Dev	16.54	36748

Normalized (min-max)

	Age	Income
Person 1	0.00	0.00
Person 2	0.05	0.14
Person 3	0.16	0.24
Person 4	0.27	0.31
Person 5	0.27	0.34
Person 6	0.23	0.38
Person 7	0.64	0.41
Person 8	0.91	0.45
Person 9	1.00	0.52
Person 10	0.98	0.66
Person 11	0.02	1.00

Standardize (z Score)

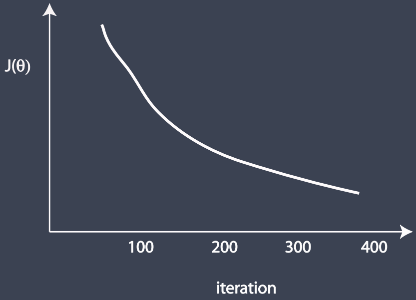
	Age	Income
Person 1	-1.09	-1.60
Person 2	-0.97	-1.05
Person 3	-0.67	-0.64
Person 4	-0.37	-0.37
Person 5	-0.37	-0.24
Person 6	-0.49	-0.10
Person 7	0.60	0.04
Person 8	1.32	0.17
Person 9	1.57	0.45
Person 10	1.51	0.99

» Practicalities: Normalising Data

* Commonly replace x_j with $\frac{x_j-\mu_j}{\sigma_j}$ where:
  * Shift $\mu_j = \frac{1}{m}\sum_{i=1}^{n} x_j^{(i)}$ tries to make features have approximately zero mean (do not apply to $x_0 = 1$ though)
  * Scaling factor $\sigma_j = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(x_j^{(i)} - \mu)^2}$ tries to make features mostly lie between -1 and 1

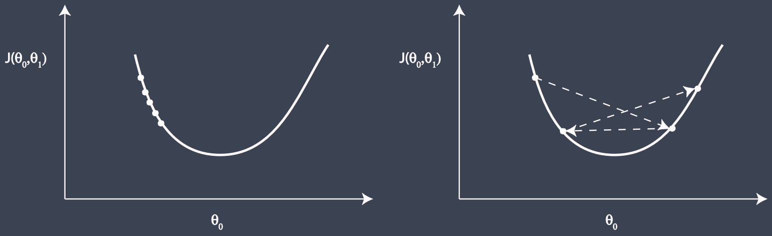
  or:
  * Shift $\mu_j = min(x_j^{(i)})$ (do not apply to $x_0 = 1$ though)
  * Scaling factor $\sigma_j = max(x_j) - min(x_j)$
* E.g. in advertising data TV budget values lie in range 0.7 to 296.4, so rescaling as $\frac{TV-0.7}{296-0.7}$ gives a feature with values in interval $0 \leq x_1 \leq 1$

» Practicalities: When to stop?



- "Debugging": How to make sure gradient descent is working correctly $\rightarrow J(\theta)$ should decrease after every iteration
- Stopping criteria: stop when decreases by less than e.g. 10^{-2} or after a fixed number of iterations e.g. 200, whichever comes first.

» Practicalities: Selecting Step Size



- Selecting step size α too small will mean it takes a long time to converge to minimum
- But selecting α too large can lead to us overshooting the minimum
- We need to adjust α so that algorithm converges in a reasonable time.
- There are also many automated approaches for adjusting α at each iteration. E.g. using *line search* (at each gradient descent iteration try several values of α until find one that causes descent).

» Python sklearn

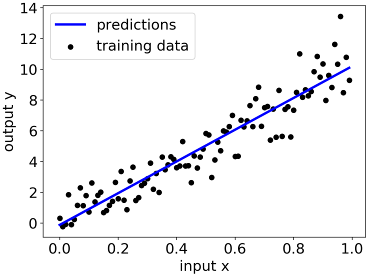
* We'll use python and usually sklearn https://scikit-learn.org/stable/index.html in examples and assignments
    * Sometimes you'll be asked to implement things from scratch rather than calling the sklearn function → to help you understand what's happening "under the hood"
* Linear regression:

```
import numpy as np
Xtrain = np.arange(0,1,0.01).reshape(-1, 1)
ytrain = 10*Xtrain + np.random.normal(0.0,1.0,100).reshape(-1, 1)
from sklearn.linear_model import LinearRegressio
model = LinearRegression().fit(Xtrain.reshape(-1, 1), ytrain.reshape(-1, 1))
print(model.intercept_, model.coef_)
```

Typical output:
0.0175381 9.77234168

Plotting model predictions:

```
import matplotlib.pyplot as plt
plt.rc('font', size=18)
plt.rcParams['figure.constrained_layout.use'] = True
plt.scatter(Xtrain, ytrain, color='black')
plt.plot(Xtrain, ypred, color='blue', linewidth=3)
plt.xlabel("input x"); plt.ylabel("output y")
plt.legend(["predictions","training data"])
plt.show()
```



» Notes On Presenting Plots

In your assignments and individual project reports:

* Always label axes in plots
* Make sure text and numbers are legible and plot is easy to read (use colours, adjust line width/marker size etc). If really illegible, you should expect to lose marks.
* Always clearly explain what data a plot shows - giving code is *not* enough, you must explain in english.
  If you don't do this you should expect to lose marks.

» Linear Regression with Multiple Variables

Advertising example:

TV x_1	Radio x_2	Newspaper x_3	Sales y
230.1	37.8	69.2	22.1
44.5	39.3	45.1	10.4
17.2	45.9	69.3	9.3
$\vdots$	$\vdots$	$\vdots$	$\vdots$

* n=number of features (3 in this example)
* $(x^{(i)}, y^{(i)})$ the ith 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 ith training example, e.g. $x_2^{(1)} = 37.8$

» Linear Regression with Multiple Variables

Model: $h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3$

e.g. $\underbrace{h_\theta(x)}_{Sales} = 15 + 0.1 \underbrace{x_1}_{TV} - 5 \underbrace{x_2}_{Radio} + 10 \underbrace{x_3}_{Newspaper}$

- For convenience, define $x_0 = 1$
- Feature vector $x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$
- Parameter vector $\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix}$
- $h_\theta(x) = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n = \theta^T x$

» Linear Regression with Multiple Variables

- Model: $h_\theta(x) = \theta^T x$
  (with θ, x now $n+1$-dimensional vectors)
- Cost Function: $J(\theta_0, \theta_1, \ldots, \theta_n) = J(\theta) = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2$
- Optimisation: Select θ that minimises $J(\theta)$
- As before, can find θ using e.g using gradient descent:
  - Start with some θ
  - Repeat:

    for j=0 to n $\{\delta_j := -\alpha \frac{\partial}{\partial \theta_j} J(\theta)\}$

    for j=0 to n $\{\theta_j := \theta_j + \delta_j\}$

» 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 θ
* Repeat:

  for j=0 to n $\{\delta_j := -\frac{2\alpha}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}\}$
  for j=0 to n $\{\theta_j := \theta_j + \delta_j\}$

» 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 ?