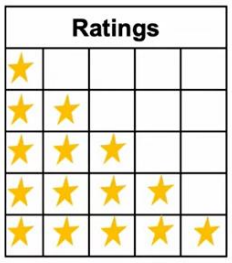
Predicting movie ratings

User rates movies using one to five stars

zero

Movie	Alice(1)	Bob(2)	Carol(3)	Dave(4)
Love at last	5	5	0	0
Romance forever	5	?	?	0
Cute puppies of love	?	4	0	?
Nonstop car chases	0	0	5	4
Swords vs. karate	0	0	5	?



n_u = no. of users

n_m = no. of movies

$r(i,j)=1$ if user j has rated movie i

$y^{(i,j)}$ = rating given by user j to movie i (defined only if $r(i,j)=1$)

$n_u = 4$ $r(1,1) = 1$

$n_m = 5$ $r(3,1) = 0$ $y^{(3,2)} = 4$

What if we have features of the movies?

$n_u = 4$
$n_m = 5$
$n = 2$

Movie	Alice(1)	Bob(2)	Carol(3)	Dave(4)	x_1 (romance)	x_2 (action)
Love at last	5	5	0	0	0.9	0
Romance forever	5	?	?	0	1.0	0.01
Cute puppies of love	?	4	0	?	0.99	0
Nonstop car chases	0	0	5	4	0.1	1.0
Swords vs. karate	0	0	5	?	0	0.9

$x^{(1)} = \begin{bmatrix} 0.9 \\ 0 \end{bmatrix}$

$x^{(3)} = \begin{bmatrix} 0.99 \\ 0 \end{bmatrix}$

For user 1: Predict rating for movie i as: $w^{(1)} \cdot x^{(i)} + b^{(1)}$ — just linear regression

$w^{(1)} = \begin{bmatrix} 5 \\ 0 \end{bmatrix}$ $b^{(1)} = 0$ $x^{(3)} = \begin{bmatrix} 0.99 \\ 0 \end{bmatrix}$

$w^{(1)} \cdot x^{(3)} + b^{(1)} = 4.95$

For user j: Predict user j's rating for movie i as $w^{(j)} \cdot x^{(i)} + b^{(j)}$

Cost function

Notation:

- $r(i,j) = 1$ if user j has rated movie i (0 otherwise)
- $y^{(i,j)}$ = rating given by user j on movie i (if defined)
- $w^{(j)}, b^{(j)}$ = parameters for user j
- $x^{(i)}$ = feature vector for movie i

For user j and movie i, predict rating: $w^{(j)} \cdot x^{(i)} + b^{(j)}$

$m^{(j)}$ = no. of movies rated by user j

To learn $w^{(j)}, b^{(j)}$

number of features

$$\min_{w^{(j)}, b^{(j)}} J\left(w^{(j)}, b^{(j)}\right) = \frac{1}{2m^{(j)}} \sum_{i:r(i,j)=1} \left(w^{(j)} \cdot x^{(i)} + b^{(j)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2m^{(j)}} \sum_{k=1}^{n} \left(w_k^{(j)}\right)^2$$

predicting user j's ratings for other movies.

Cost function

To learn parameters $w^{(j)}, b^{(j)}$ for user j :

$$J\left(w^{(j)}, b^{(j)}\right) = \frac{1}{2} \sum_{i:r(i,j)=1} \left(w^{(j)} \cdot x^{(i)} + b^{(j)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2} \sum_{k=1}^{n} \left(w_k^{(j)}\right)^2$$

To learn parameters $w^{(1)}, b^{(1)}, w^{(2)}, b^{(2)}, \cdots w^{(n_u)}, b^{(n_u)}$ for all users :

$$J\begin{pmatrix} w^{(1)}, \ldots, w^{(n_u)} \\ b^{(1)}, \ldots, b^{(n_u)} \end{pmatrix} = \frac{1}{2} \sum_{j=1}^{n_u} \sum_{i:r(i,j)=1} \left(w^{(j)} \cdot x^{(i)} + b^{(j)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2} \sum_{j=1}^{n_u} \sum_{k=1}^{n} \left(w_k^{(j)}\right)^2$$

learning all the parameters for all of the users.

Problem motivation

Movie	Alice (1)	Bob (2)	Carol (3)	Dave (4)	x_1 (romance)	x_2 (action)
Love at last	5	5	0	0	?	?
Romance forever	5	?	?	0 ←	?	? $x^{(2)}$
Cute puppies of love	?	4	0	? ←	?	? $x^{(3)}$
Nonstop car chases	0	0	5	4 ←	?	?
Swords vs. karate	0	0	5	? ←	?	?

$w^{(1)} = \begin{bmatrix} 5 \\ 0 \end{bmatrix}$, $w^{(2)} = \begin{bmatrix} 5 \\ 0 \end{bmatrix}$, $w^{(3)} = \begin{bmatrix} 0 \\ 5 \end{bmatrix}$, $w^{(4)} = \begin{bmatrix} 0 \\ 5 \end{bmatrix}$

$b^{(1)} = 0$, $b^{(2)} = 0$, $b^{(3)} = 0$, $b^{(4)} = 0$ ←

using $w^{(j)} \cdot x^{(i)} + b^{(j)}$

$w^{(1)} \cdot x^{(1)} \approx 5$

$w^{(2)} \cdot x^{(1)} \approx 5$

$w^{(3)} \cdot x^{(1)} \approx 0$

$w^{(4)} \cdot x^{(1)} \approx 0$

→ $x^{(1)} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$

4:46 / 13:55

Cost function

Given $w^{(1)}, b^{(1)}, w^{(2)}, b^{(2)}, \cdots, w^{(n_u)}, b^{(n_u)}$

to learn $x^{(i)}$:

$$J(x^{(i)}) = \frac{1}{2} \sum_{j:r(i,j)=1} \left(w^{(j)} \cdot x^{(i)} + b^{(j)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2} \sum_{k=1}^{n} \left(x_k^{(i)}\right)^2$$

→ To learn $x^{(1)}, x^{(2)}, \cdots, x^{(n_m)}$:

$$J(x^{(1)}, x^{(2)}, \ldots, x^{(n_m)}) = \frac{1}{2} \sum_{i=1}^{n_m} \sum_{j:r(i,j)=1} \left(w^{(j)} \cdot x^{(i)} + b^{(j)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2} \sum_{i=1}^{n_m} \sum_{k=1}^{n} \left(x_k^{(i)}\right)^2$$

that will give us our collaborative filtering algorithm.

DeepLearning.AI Stanford ONLINE Andrew Ng

7:54 / 13:55

Collaborative filtering

	$j=1$ Alice	$j=2$ Bob	$j=3$ Carol
$i=1$ Movie1	5	5	?
$i=2$ Movie2	?	2	3

Cost function to learn $w^{(1)}, b^{(1)}, \cdots w^{(n_u)}, b^{(n_u)}$:

$$\min_{w^{(1)},b^{(1)},\cdots,w^{(n_u)},b^{(n_u)}} \frac{1}{2}\sum_{j=1}^{n_u}\sum_{i:r(i,j)=1}\left(w^{(j)}\cdot x^{(i)}+b^{(j)}-y^{(i,j)}\right)^2 + \frac{\lambda}{2}\sum_{j=1}^{n_u}\sum_{k=1}^{n}\left(w_k^{(j)}\right)^2$$

Cost function to learn $x^{(1)}, \cdots, x^{(n_m)}$:

$$\min_{x^{(1)},\cdots,x^{(n_m)}} \frac{1}{2}\sum_{i=1}^{n_m}\sum_{j:r(i,j)=1}\left(w^{(j)}\cdot x^{(i)}+b^{(j)}-y^{(i,j)}\right)^2 + \frac{\lambda}{2}\sum_{i=1}^{n_m}\sum_{k=1}^{n}\left(x_k^{(i)}\right)^2$$

Put them together:

$$\min_{\substack{w^{(1)},\ldots,w^{(n_u)}\\ b^{(1)},\ldots,b^{(n_u)}\\ x^{(1)},\ldots,x^{(n_m)}}} J(w,b,x) = \frac{1}{2}\sum_{(i,j):r(i,j)=1}\left(w^{(j)}\cdot x^{(i)}+b^{(j)}-y^{(i,j)}\right)^2 + \frac{\lambda}{2}\sum_{j=1}^{n_u}\sum_{k=1}^{n}\left(w_k^{(j)}\right)^2 + \frac{\lambda}{2}\sum_{i=1}^{n_m}\sum_{k=1}^{n}\left(x_k^{(i)}\right)^2$$

as a function of w, b, and x?

DeepLearning.AI Stanford ONLINE Andrew Ng

10:41 / 13:55

Gradient Descent

collaborative filtering

Linear regression (course 1)

repeat {

$$w_i = w_i - \alpha\frac{\partial}{\partial w_i}J(w,b)$$
$$b = b - \alpha\frac{\partial}{\partial b}J(w,b)$$

$$w_i^{(j)} = w_i^{(j)} - \alpha\frac{\partial}{\partial w_i^{(j)}}J(w,b,x)$$
$$b^{(j)} = b^{(j)} - \alpha\frac{\partial}{\partial b^{(j)}}J(w,b,x)$$
$$x_k^{(i)} = x_k^{(i)} - \alpha\frac{\partial}{\partial x_k^{(i)}}J(w,b,x)$$

}

parameters w, b, x x is also a parameter

w and b, as well as x.

DeepLearning.AI Stanford ONLINE Andrew Ng

12:43 / 13:55

Binary labels

Movie	Alice(1)	Bob(2)	Carol(3)	Dave(4)
Love at last	1	1	0	0
Romance forever	1	?	?	0
Cute puppies of love	?	1	0	?
Nonstop car chases	0	0	1	1
Swords vs. karate	0	0	1	?

we can then decide how much we should recommend these items to them.

DeepLearning.AI Stanford ONLINE Andrew Ng

1:46 / 8:27

From regression to binary classification

- Previously:
  - Predict $y^{(i,j)}$ as $w^{(j)} \cdot x^{(i)} + b^{(j)}$
- For binary labels:
  - Predict that the probability of $y^{(i,j)} = 1$ is given by $g(w^{(j)} \cdot x^{(i)} + b^{(j)})$

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

1 that is of the user having engaged with or like the item using this model.

DeepLearning.AI Stanford ONLINE Andrew Ng

5:13 / 8:27

Cost function for binary application

Previous cost function:

$$\frac{1}{2}\sum_{(i,j):r(i,j)=1}\left(w^{(j)}\cdot x^{(i)}+b^{(j)}-y^{(i,j)}\right)^2+\frac{\lambda}{2}\sum_{i=1}^{n_m}\sum_{k=1}^{n}\left(x_k^{(i)}\right)^2+\frac{\lambda}{2}\sum_{j=1}^{n_u}\sum_{k=1}^{n}\left(w_k^{(j)}\right)^2$$

$f(x)$

Loss for binary labels $y^{(i,j)}$: $f_{(w,b,x)}(x)=g(w^{(j)}\cdot x^{(i)}+b^{(j)})$

$$L\left(f_{(w,b,x)}(x),y^{(i,j)}\right)=-y^{(i,j)}\log\left(f_{(w,b,x)}(x)\right)-\left(1-y^{(i,j)}\right)\log\left(1-f_{(w,b,x)}(x)\right)$$

Loss for single example

$$J(w,b,x)=\sum_{(i,j):r(i,j)=1}L\left(f_{(w,b,x)}(x),y^{(i,j)}\right)$$

$g(w^{(j)}\cdot x^{(i)}+b^{(j)})$

cost for all examples

DeepLearning.AI Stanford ONLINE Andrew Ng

7:46 / 8:27

Users who have not rated any movies

Movie	Alice(1)	Bob (2)	Carol (3)	Dave (4)	Eve (5)	
Love at last	5	5	0	0	?	$r^{(1)}$
Romance forever	5	?	?	0	?	$r^{(2)}$
Cute puppies of love	?	4	0	?	?	$r^{(3)}$
Nonstop car chases	0	0	5	4	?	$r^{(4)}$
Swords vs. karate	0	0	5	?	?	$r^{(5)}$

$$\begin{bmatrix}5&5&0&0&?\\5&?&?&0&?\\?&4&0&?&?\\0&0&5&4&?\\0&0&5&0&?\end{bmatrix}$$

$$\min_{\substack{w^{(1)},\ldots w^{(n_u)}\\ b^{(1)},\ldots b^{(n_u)}\\ x^{(1)},\ldots x^{(n_m)}}}\frac{1}{2}\sum_{(i,j):r(i,j)=1}\left(w^{(j)}\cdot x^{(i)}+b^{(j)}-y^{(i,j)}\right)^2+\frac{\lambda}{2}\sum_{j=1}^{n_u}\sum_{k=1}^{n}\left(w_k^{(j)}\right)^2+\frac{\lambda}{2}\sum_{i=1}^{n_m}\sum_{k=1}^{n}\left(x_k^{(i)}\right)^2$$

sustained and more compact way.

3:07 / 8:45

Mean Normalization

$$\begin{bmatrix} 5 & 5 & 0 & 0 & ? \\ 5 & ? & ? & 0 & ? \\ ? & 4 & 0 & ? & ? \\ 0 & 0 & 5 & 4 & ? \\ 0 & 0 & 5 & 0 & ? \end{bmatrix} \quad \begin{matrix} 2.5 \\ 2.5 \\ 2 \\ 2.25 \\ 1.25 \end{matrix}$$

$$\mu = \begin{bmatrix} 2.5 \\ 2.5 \\ 2 \\ 2.25 \\ 1.25 \end{bmatrix}$$

$$\begin{bmatrix} 2.5 & 2.5 & -2.5 & -2.5 & ? \\ 2.5 & ? & ? & -2.5 & ? \\ ? & 2 & -2 & ? & ? \\ -2.25 & -2.25 & 2.75 & 1.75 & ? \\ -1.25 & -1.25 & 3.75 & -1.25 & ? \end{bmatrix}$$

For user j, on movie i predict:

$$w^{(j)} \cdot x^{(i)} + b^{(j)} + \mu_i$$

$y^{(i,j)}$

$w^{(j)}, b^{(j)}, x^{(i)}$

User 5 (Eve):

$w^{(5)} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ $b^{(5)} = 0$

$w^{(5)} \cdot x^{(1)} + b^{(5)} + \mu_1 = 2.5$

0

8:09 / 8:45

Custom Training Loop

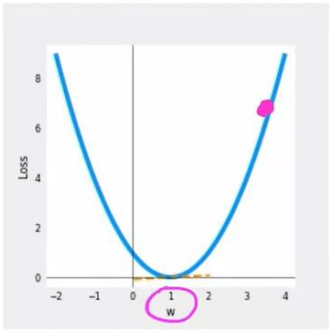
$$J = (\underbrace{wx}_{f(x)} - \underbrace{1}_{y})^2$$

Gradient descent algorithm

Repeat until convergence

$$w = w - \alpha \frac{\partial}{\partial w} J(w,b)$$

Fix b = 0 for this example




```
w = tf.Variable(3.0)
x = 1.0
y = 1.0 # target value
alpha = 0.01

iterations = 30
for iter in range(iterations):
    # Use TensorFlow's Gradient tape to record the steps
    # used to compute the cost J, to enable auto differentiation.
    with tf.GradientTape() as tape:
        fwb = w*x
        costJ = (fwb - y)**2

    # Use the gradient tape to calculate the gradients
    # of the cost with respect to the parameter w.
    [dJdw] = tape.gradient( costJ, [w] )

    # Run one step of gradient descent by updating
    # the value of w to reduce the cost.
    w.assign_add(-alpha * dJdw)
```

Tf.variables are the parameters we want to optimize

Auto Diff

Auto Grad

f(x)

$\frac{\partial}{\partial w} J(w)$

tf.variables require special function to modify

DeepLearning.AI Stanford ONLINE Andrew Ng

7:09 / 11:37

Implementation in TensorFlow

Gradient descent algorithm

Repeat until convergence

$$w = w - \alpha \frac{\partial}{\partial w} J(w,b,x)$$
$$b = b - \alpha \frac{\partial}{\partial b} J(w,b,x)$$
$$X = X - \alpha \frac{\partial}{\partial x} J(w,b,x)$$

```
# Instantiate an optimizer.
optimizer = keras.optimizers.Adam(learning_rate=1e-1)

iterations = 200
for iter in range(iterations):
    # Use TensorFlow's GradientTape
    # to record the operations used to compute the cost
    with tf.GradientTape() as tape:

        # Compute the cost (forward pass is included in cost)
        cost_value = cofiCostFuncV(X, W, b, Ynorm, R,
                                   num_users, num_movies, lambda)
    # Use the gradient tape to automatically retrieve
    # the gradients of the trainable variables with respect to
          the loss
    grads = tape.gradient( cost_value, [X,W,b] )

    # Run one step of gradient descent by updating
    # the value of the variables to minimize the loss.
    optimizer.apply_gradients( zip(grads, [X,W,b]) )
```

Dataset credit: Harper and Konstan. 2015. The MovieLens Datasets: History and Context

DeepLearning.AI Stanford ONLINE Andrew Ng

9:57 / 11:37

Finding related items

The features $x^{(i)}$ of item i are quite hard to interpret.

To find other items related to it,
find item k with $x^{(k)}$ similar to $x^{(i)}$

i.e. with smallest distance

$$\sum_{l=1}^{n} \left(x_l^{(k)} - x_l^{(i)} \right)^2$$

$$\left\| x^{(k)} - x^{(i)} \right\|^2$$

an even more powerful recommended system as well.

DeepLearning.AI Stanford ONLINE Andrew Ng

3:22 / 6:32

Limitations of Collaborative Filtering

Cold start problem. How to

- rank new items that few users have rated?
- show something reasonable to new users who have rated few items?

Use side information about items or users:

- Item: Genre, movie stars, studio, ….
- User: Demographics (age, gender, location), expressed preferences, …

you guess what might the user be interested in,

DeepLearning.AI Stanford ONLINE Andrew Ng

Collaborative filtering vs Content-based filtering

Collaborative filtering:
Recommend items to you based on ratings of users who gave similar ratings as you

Content-based filtering:
Recommend items to you based on features of user and item to find good match

$r(i,j)=1$ if user j has rated item i

$y^{(i,j)}$ rating given by user j on item i (if defined)

content-based filtering is that

DeepLearning.AI Stanford ONLINE Andrew Ng

1:41 / 9:45

Examples of user and item features

User features:

- Age
- Gender (1 hot)
- Country (1 hot, 200)
- Movies watched (1000)
- Average rating per genre
- ...

$x_u^{(j)}$ for user j

Vector size could be different

Movie features:

- Year
- Genre/Genres
- Reviews
- Average rating
- ...

$x_m^{(i)}$ for movie i

could be just 50 numbers.

DeepLearning.AI Stanford ONLINE Andrew Ng

5:42 / 9:45

Content-based filtering: Learning to match

Predict rating of user *j* on movie *i* as

$w^{(j)} \cdot x^{(i)} + b^{(j)}$

$v_u^{(j)} \cdot v_m^{(i)}$

computed from $x_u^{(j)}$

computed from $x_m^{(i)}$

$x_u^{(j)}$ → $v_u^{(j)}$ = user's preferences = [4.9 (likes), 0.1 (likes), ..., 3.0]

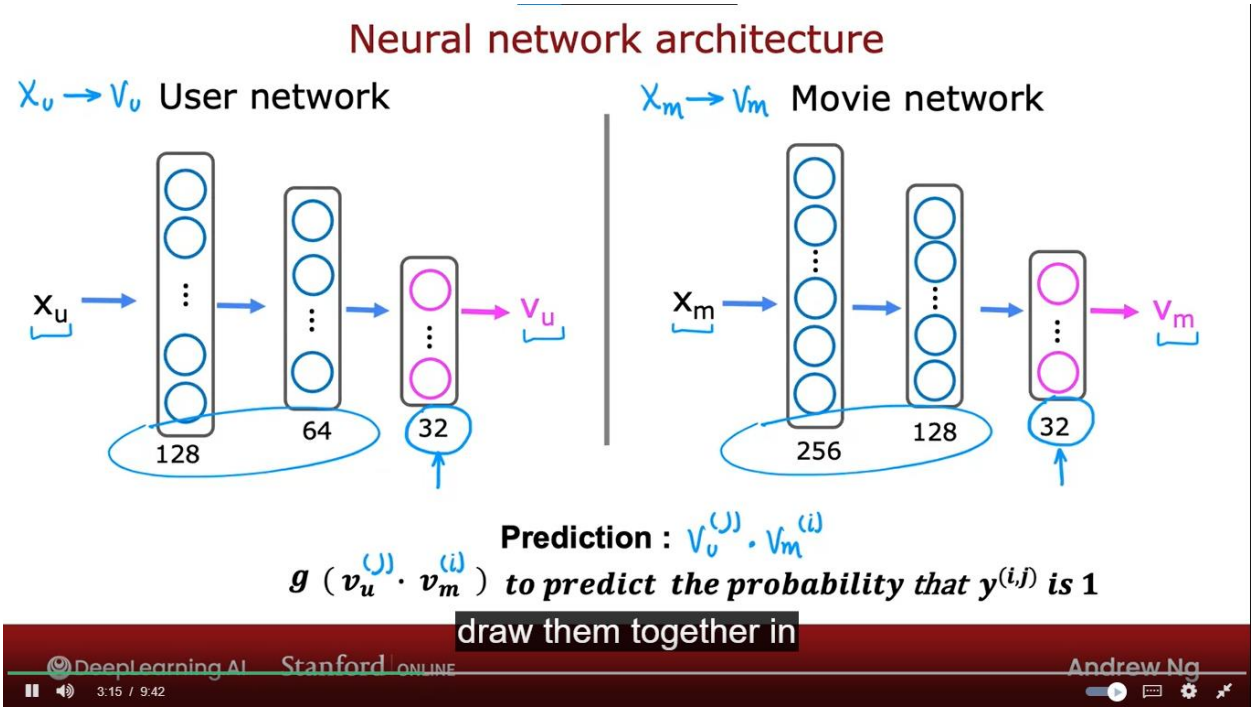
32

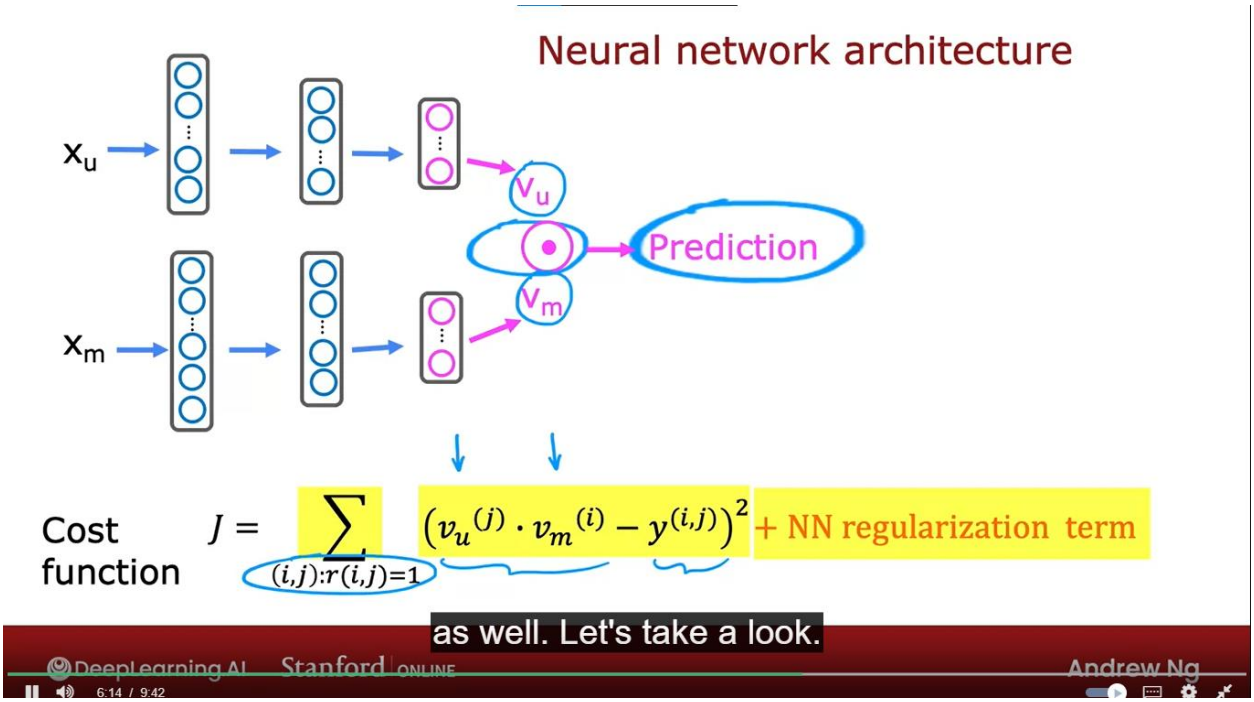
$x_m^{(i)}$ → $v_m^{(i)}$ = movie features = [4.5 (romance), 0.2 (action), ..., 3.5]

maybe both of these are say 32 numbers.

DeepLearning.AI Stanford ONLINE Andrew Ng

9:07 / 9:45


Neural network architecture
$x_u \to v_u$ User network
$x_m \to v_m$ Movie network
x_u
128
64
32
v_u
x_m
256
128
32
v_m
Prediction : $v_u^{(j)} \cdot v_m^{(i)}$
$g(v_u^{(j)} \cdot v_m^{(i)})$ to predict the probability that $y^{(i,j)}$ is 1
draw them together in
DeepLearning.AI Stanford ONLINE
Andrew Ng
3:15 / 9:42


Neural network architecture
x_u
x_m
v_u
v_m
Prediction
Cost function
$J = \sum_{(i,j):r(i,j)=1} \left(v_u^{(j)} \cdot v_m^{(i)} - y^{(i,j)}\right)^2$ + NN regularization term
as well. Let's take a look.
DeepLearning.AI Stanford ONLINE
Andrew Ng
6:14 / 9:42

Learned user and item vectors:

$v_u^{(j)}$ is a vector of length 32 that describes user j with features $x_u^{(j)}$

$v_m^{(i)}$ is a vector of length 32 that describes movie i with features $x_m^{(i)}$

To find movies similar to movie i: $\|v_m^{(k)} - v_m^{(i)}\|^2$ small

$\|x^{(k)} - x^{(i)}\|^2$

Note: This can be pre-computed ahead of time

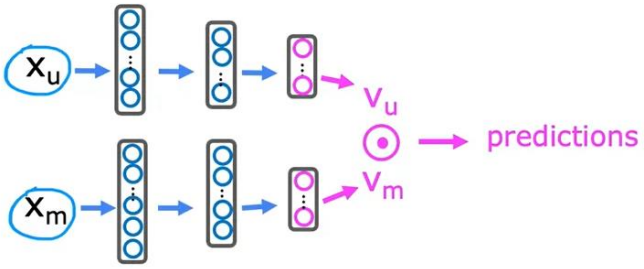
later when we talk about scaling

DeepLearning.AI Stanford ONLINE Andrew Ng

8:01 / 9:42

How to efficiently find recommendation from a large set of items?

- Movies 1000+
- Ads 1m+
- Songs 10m+
- Products 10m+



becomes computationally infeasible.

DeepLearning.AI Stanford ONLINE Andrew Ng

1:17 / 7:52

Two steps: Retrieval & Ranking

Retrieval:

- Generate large list of plausible item candidates ~100s
  e.g.
  1) For each of the last 10 movies watched by the user, find 10 most similar movies

$$\| v_m^{(k)} - v_m^{(i)} \|^2$$

  2) For most viewed 3 genres, find the top 10 movies
  3) Top 20 movies in the country

- Combine retrieved items into list, removing duplicates and items already watched/purchased

that the user has already purchased and

DeepLearning.AI Stanford ONLINE Andrew Ng

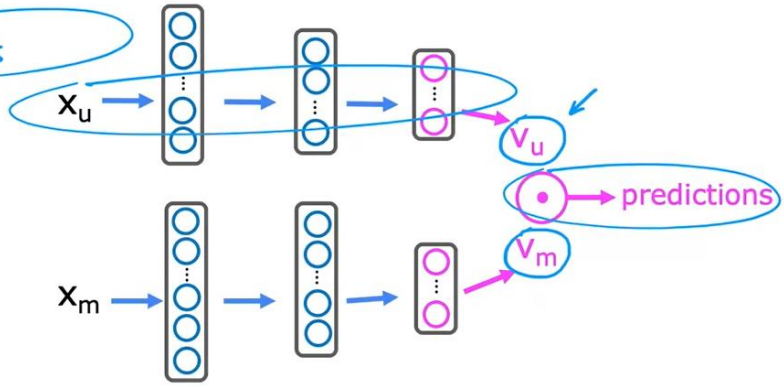
3:58 / 7:52

Two steps: Retrieval & ranking

Ranking:

~100s

- Take list retrieved and rank using learned model



- Display ranked items to user

To feed into the more accurate ranking step.

DeepLearning.AI Stanford ONLINE Andrew Ng

5:38 / 7:52

Retrieval step

- Retrieving more items results in better performance, but slower recommendations.
- To analyse/optimize the trade-off, carry out offline experiments to see if retrieving additional items results in more relevant recommendations (i.e., $p(y^{(i,j)}) = 1$ of items displayed to user are higher).

100 500

not worth doing the more detailed influence and inner product on.

DeepLearning.AI Stanford ONLINE Andrew Ng

6:57 / 7:52

What is the goal of the recommender system?

Recommend:

- Movies most likely to be rated 5 stars by user
- Products most likely to be purchased
- Ads most likely to be clicked on +high bid
- Products generating the largest profit
- Video leading to maximum watch time

Or they could also be

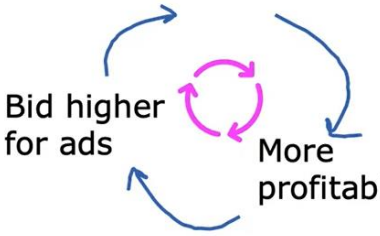
DeepLearning.AI Stanford ONLINE Andrew Ng

4:16 / 10:48

Ethical considerations with recommender systems

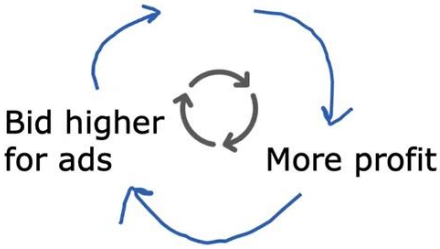
Travel industry

Good travel experience to more users



Payday loans

Squeeze customers more



Amelioration: Do not accept ads from exploitative businesses

But how do you define what is

Other problematic cases:

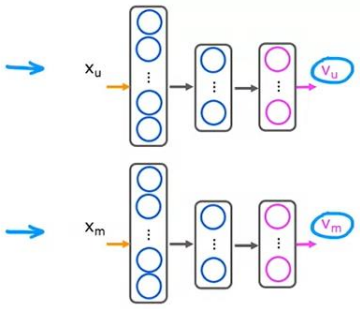
- Maximizing user engagement (e.g. watch time) has led to large social media/video sharing sites to amplify conspiracy theories and hate/toxicity

  → Amelioration : Filter out problematic content such as hate speech, fraud, scams and violent content

- Can a ranking system maximize your profit rather than users' welfare be presented in a transparent way?

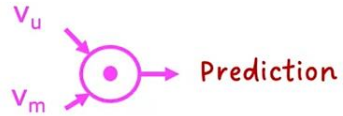
  → Amelioration : Be transparent with users

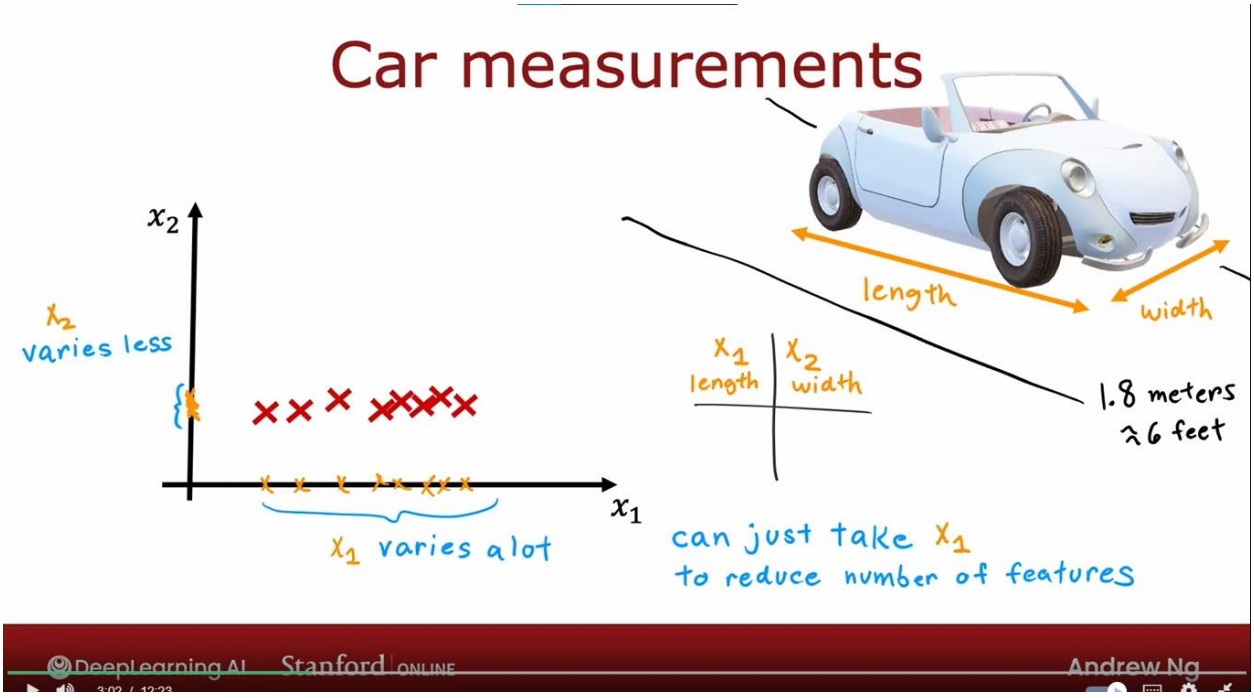
showing them and why will increase trust and



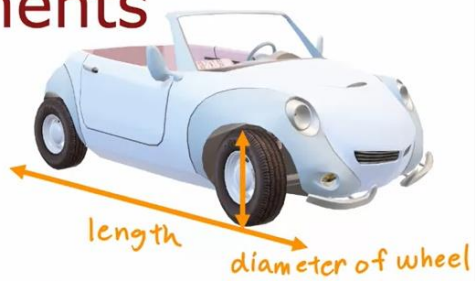

```
user_NN = tf.keras.models.Sequential([
    tf.keras.layers.Dense(256, activation='relu'),
    tf.keras.layers.Dense(128, activation='relu'),
    tf.keras.layers.Dense(32)
])

item_NN = tf.keras.models.Sequential([
    tf.keras.layers.Dense(256, activation='relu'),
    tf.keras.layers.Dense(128, activation='relu'),
    tf.keras.layers.Dense(32 )
])
```

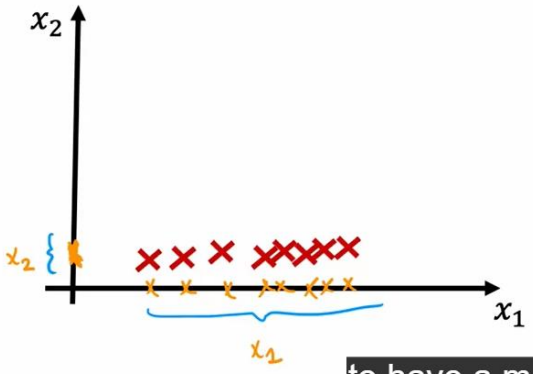
```
# create the user input and point to the base network
input_user = tf.keras.layers.Input(shape=(num_user_features))
vu = user_NN(input_user)
vu = tf.linalg.l2_normalize(vu, axis=1)

# create the item input and point to the base network
input_item = tf.keras.layers.Input(shape=(num_item_features))
vm = item_NN(input_item)
vm = tf.linalg.l2_normalize(vm, axis=1)

# measure the similarity of the two vector outputs
output = tf.keras.layers.Dot(axes=1)([vu, vm])

# specify the inputs and output of the model
model = Model([input_user, input_item], output)

# Specify the cost function
cost_fn = tf.keras.losses.MeanSquaredError()
```





Car measurements





x_1	x_2
length	diameter

can just take x_1

to have a meaningful[illegible]ce number of features

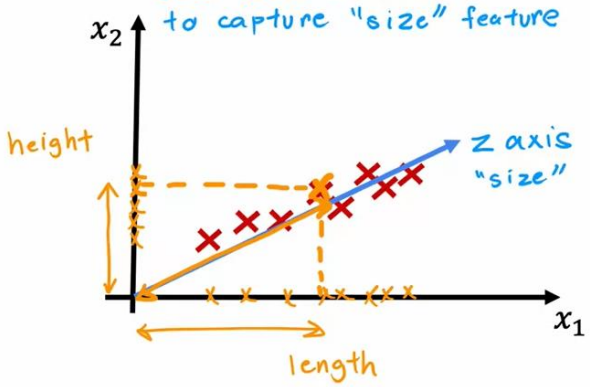
degree of variation.

DeepLearning.AI Stanford ONLINE Andrew Ng

3:51 / 12:23

Size

PCA: find new axis and coordinates
use fewer numbers
to capture "size" feature



x_1	x_2
length	height

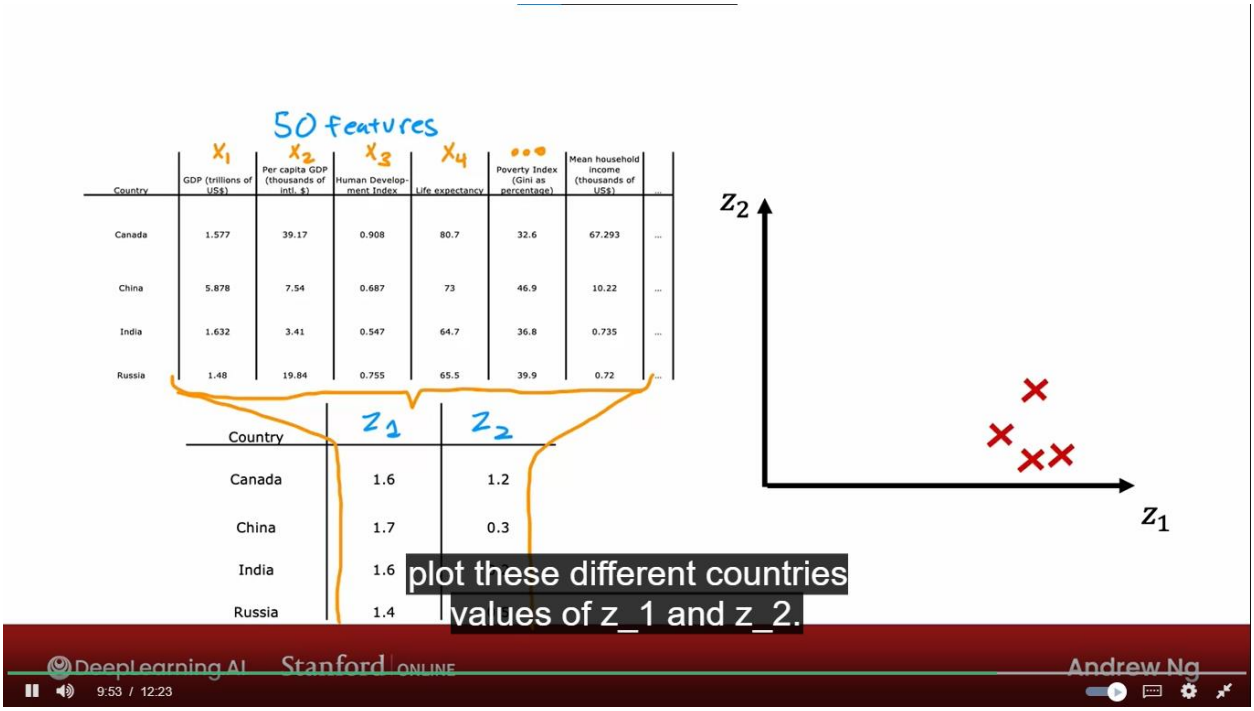
2 → 1 features

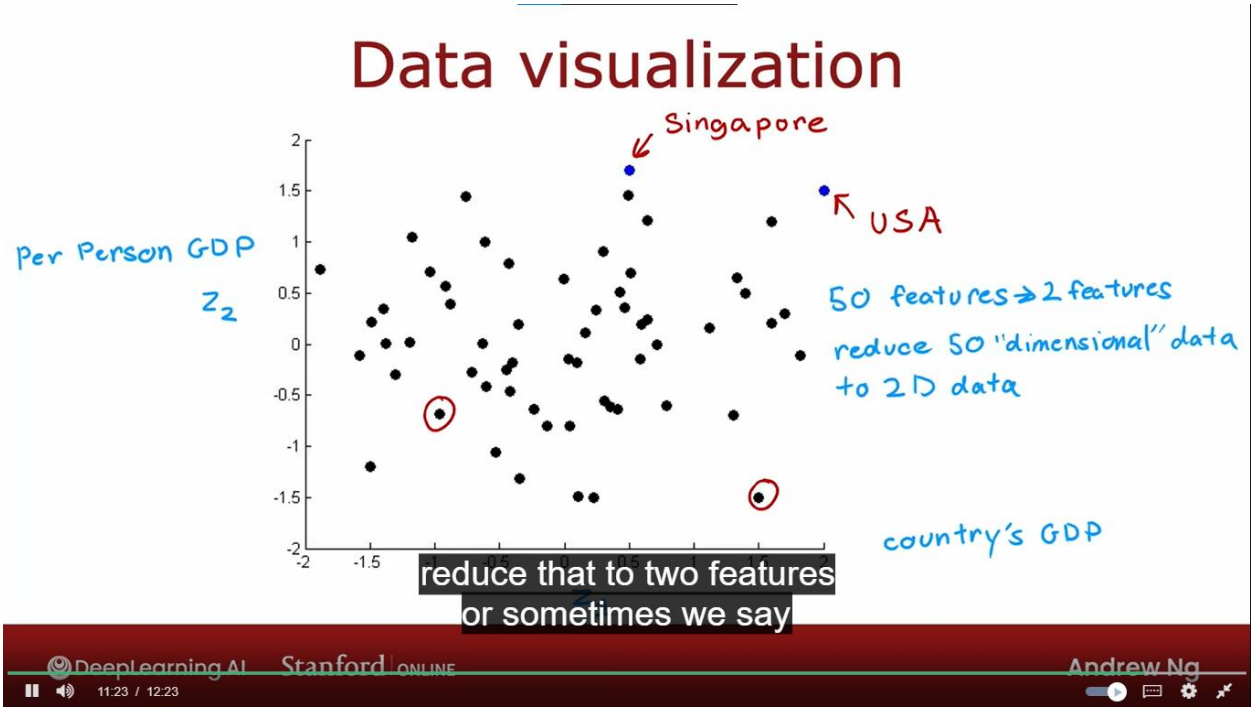
many features → 2 or 3 features

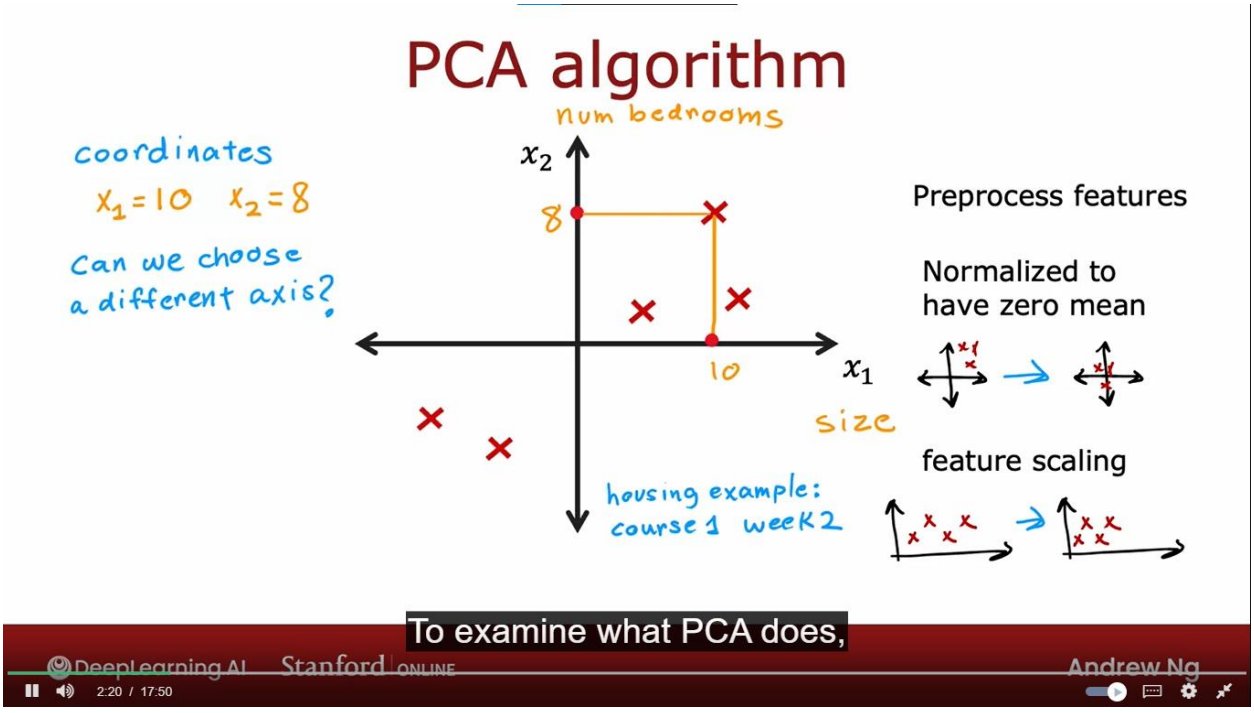
Let's look at one more example.

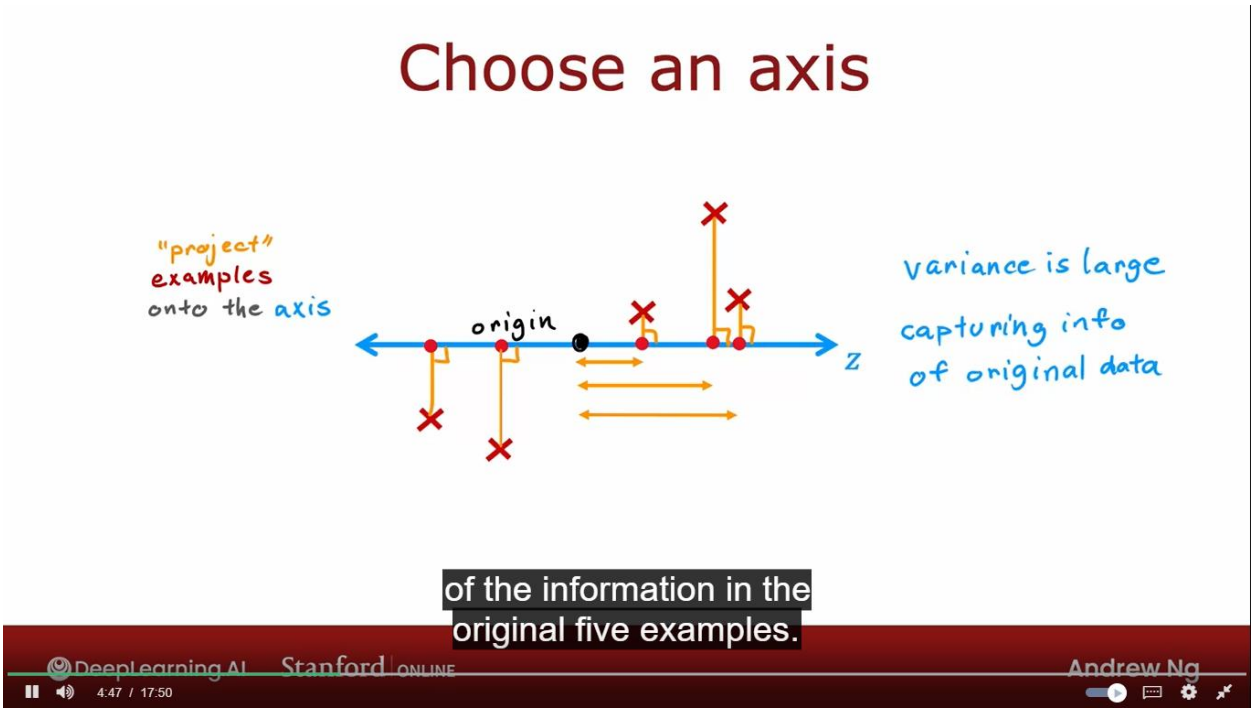
DeepLearning.AI Stanford ONLINE Andrew Ng

7:30 / 12:23

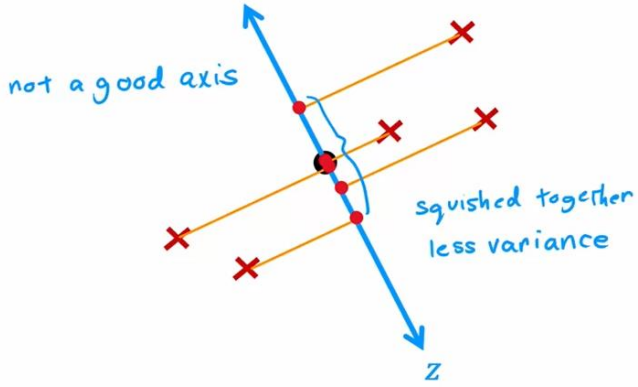

50 features
x_1
x_2
x_3
x_4
Country
GDP (trillions of US$)
Per capita GDP (thousands of intl. $)
Human Development Index
Life expectancy
Poverty Index (Gini as percentage)
Mean household income (thousands of US$)
Canada 1.577 39.17 0.908 80.7 32.6 67.293
China 5.878 7.54 0.687 73 46.9 10.22
India 1.632 3.41 0.547 64.7 36.8 0.735
Russia 1.48 19.84 0.755 65.5 39.9 0.72
z_1
z_2
Country
Canada 1.6 1.2
China 1.7 0.3
India 1.6
Russia 1.4
z_2
z_1
plot these different countries
values of z_1 and z_2.
DeepLearning.AI Stanford ONLINE Andrew Ng
9:53 / 12:23


Data visualization
Singapore
USA
Per Person GDP
z_2
50 features → 2 features
reduce 50 "dimensional" data
to 2D data
country's GDP
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1.5
reduce that to two features
or sometimes we say
DeepLearning.AI Stanford ONLINE Andrew Ng
11:23 / 12:23


PCA algorithm
num bedrooms
coordinates
$x_1 = 10$ $x_2 = 8$
Can we choose a different axis?
x_2
8
10
x_1
size
housing example: course 1 week 2
Preprocess features
Normalized to have zero mean
feature scaling
To examine what PCA does,
DeepLearning.AI Stanford ONLINE
Andrew Ng
2:20 / 17:50


Choose an axis
"project" examples onto the axis
origin
z
variance is large
capturing info of original data
of the information in the original five examples.
DeepLearning.AI Stanford ONLINE
Andrew Ng
4:47 / 17:50

Choose an axis

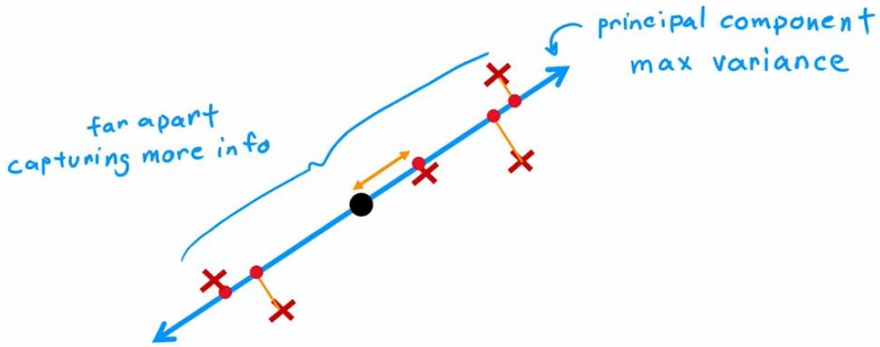


partially squish all
five examples together.

DeepLearning.AI Stanford ONLINE Andrew Ng

5:33 / 17:50

Choose an axis

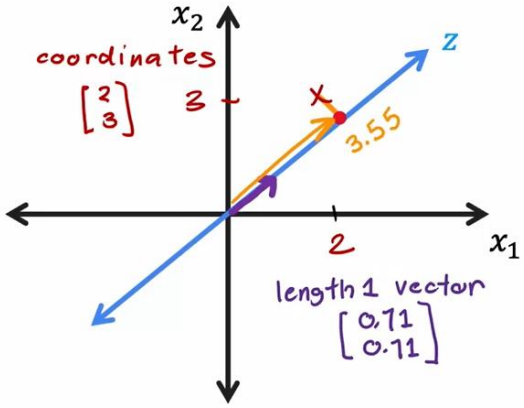


Let me show you a
visualization of how

DeepLearning.AI Stanford ONLINE Andrew Ng

6:51 / 17:50

Coordinate on the new axis



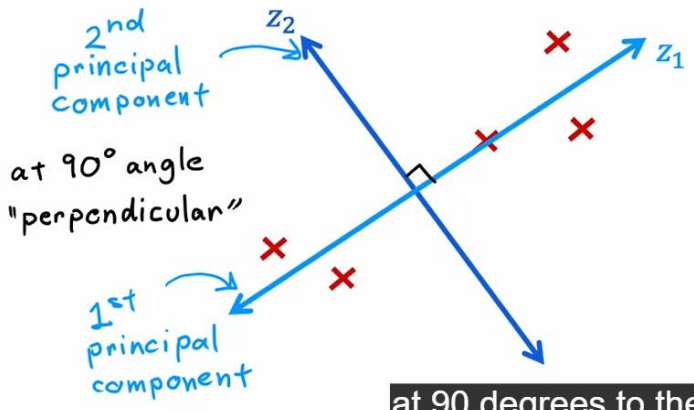
dot product

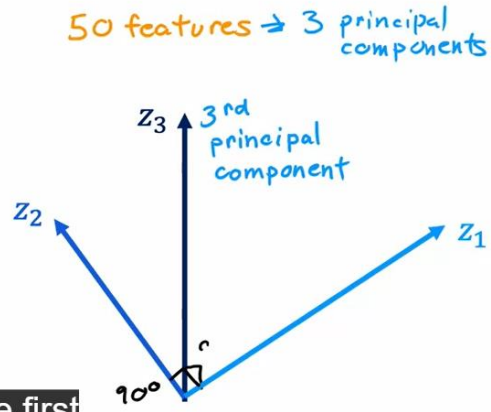
$$\begin{bmatrix} 2 \\ 3 \end{bmatrix} \cdot \begin{bmatrix} 0.71 \\ 0.71 \end{bmatrix}$$

$2 \times 0.71 + 3 \times 0.71 = 3.55$

also called sometimes the first principal component.

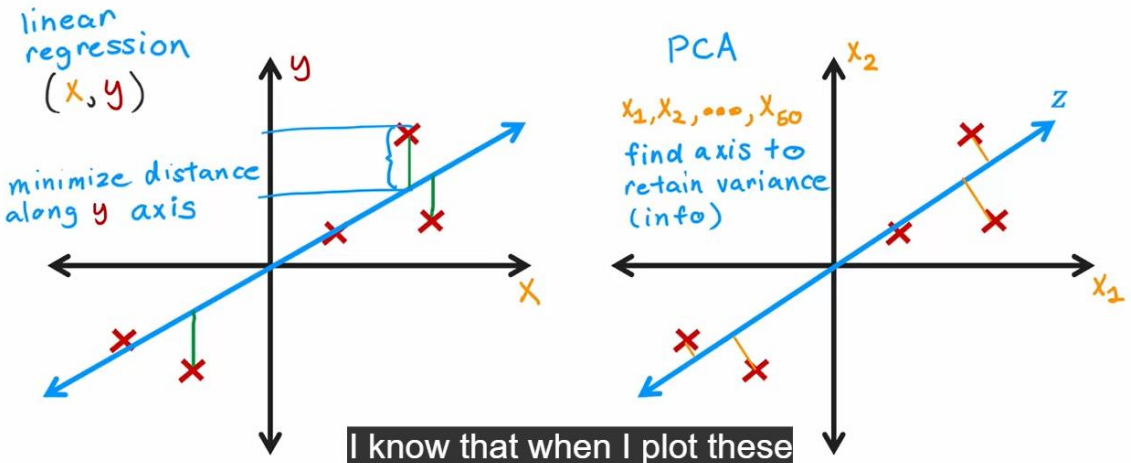
More principal components





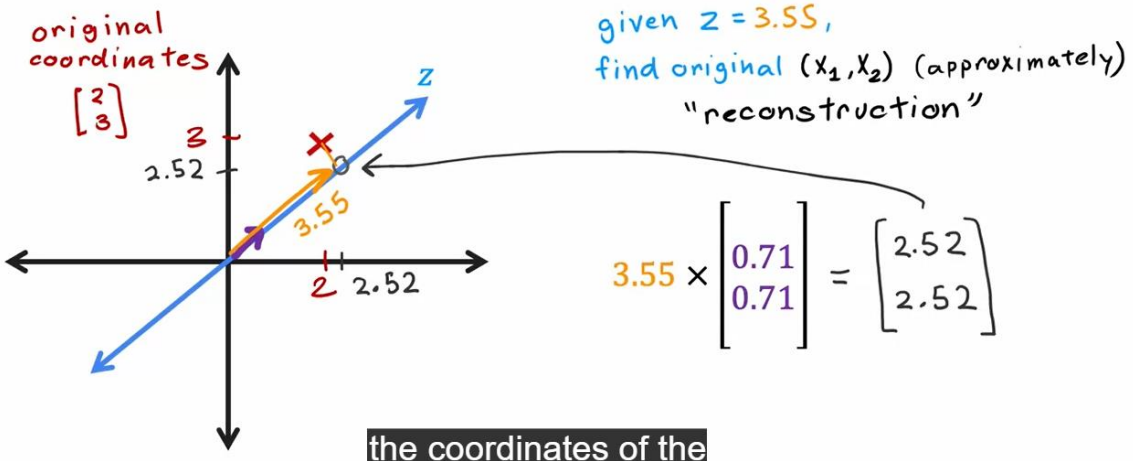
at 90 degrees to the first and the second axis.

PCA is not linear regression



I know that when I plot these things in two-dimensions,

Approximation to the original data

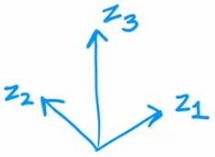


the coordinates of the original training example.

PCA in scikit-learn

Optional pre-processing: Perform feature scaling

1. "fit" the data to obtain 2 (or 3) new axes (principal components)
   for visualization
   `fit` includes mean normalization



2. Optionally examine how much variance is explained by each principal component.
   `explained_variance_ratio`
   info

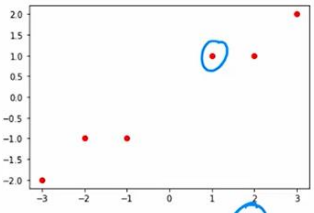
3. Transform (project) the data onto the new axes
   `transform`

to visualize your data.

DeepLearning.AI Stanford ONLINE Andrew Ng

2:45 / 11:03

Example



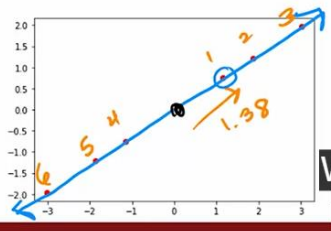
```
X = np.array([[1, 1], [2, 1], [3, 2],
              [-1, -1], [-2, -1], [-3, -2]])
```

2D

```
pca_1 = PCA(n_components=1)
pca_1.fit(X)
pca_1.explained_variance_ratio_
X_trans_1 = pca_1.transform(X)
X_reduced_1 = pca.inverse_transform(X_trans_1)
```

0.992

```
array([
[ 1.38340578],
[ 2.22189802],
[ 3.6053038 ],
[-1.38340578],
[-2.22189802],
[-3.6053038 ]])
```



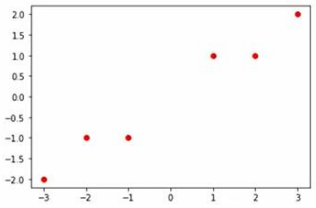
1D

What if you were to compute two principal components?

DeepLearning.AI Stanford ONLINE Andrew Ng

5:14 / 11:03

Example

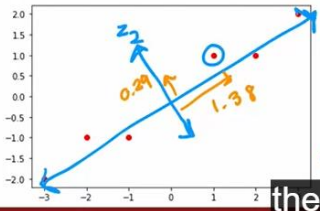


```
X = np.array([[1, 1], [2, 1], [3, 2],
              [-1, -1], [-2, -1], [-3, -2]])
```

2D

```
pca_2 = PCA(n_components=2)
pca_2.fit(X)
pca_2.explained_variance_ratio_
X_trans_2 = pca.transform(X)
X_reduced_2 = pca.inverse_transform(X_trans_2)
```

z_1 z_2
0.992 0.008



2D

```
array([
[ 1.38340578,  0.2935787 ],
[ 2.22189802, -0.25133484],
[ 3.6053038 ,  0.04224385],
[-1.38340578, -0.2935787 ],
[-2.22189802,  0.25133484],
[-3.6053038 , -0.04224385]])
```

the optional lab where you can

Applications of PCA

Visualization reduce to 2 or 3 features

Less frequently used for:

- Data compression
  (to reduce storage or transmission costs) 50 → 10
- Speeding up training of a supervised learning model

$n = 1000 \rightarrow 100$

This used to make a difference in the running time