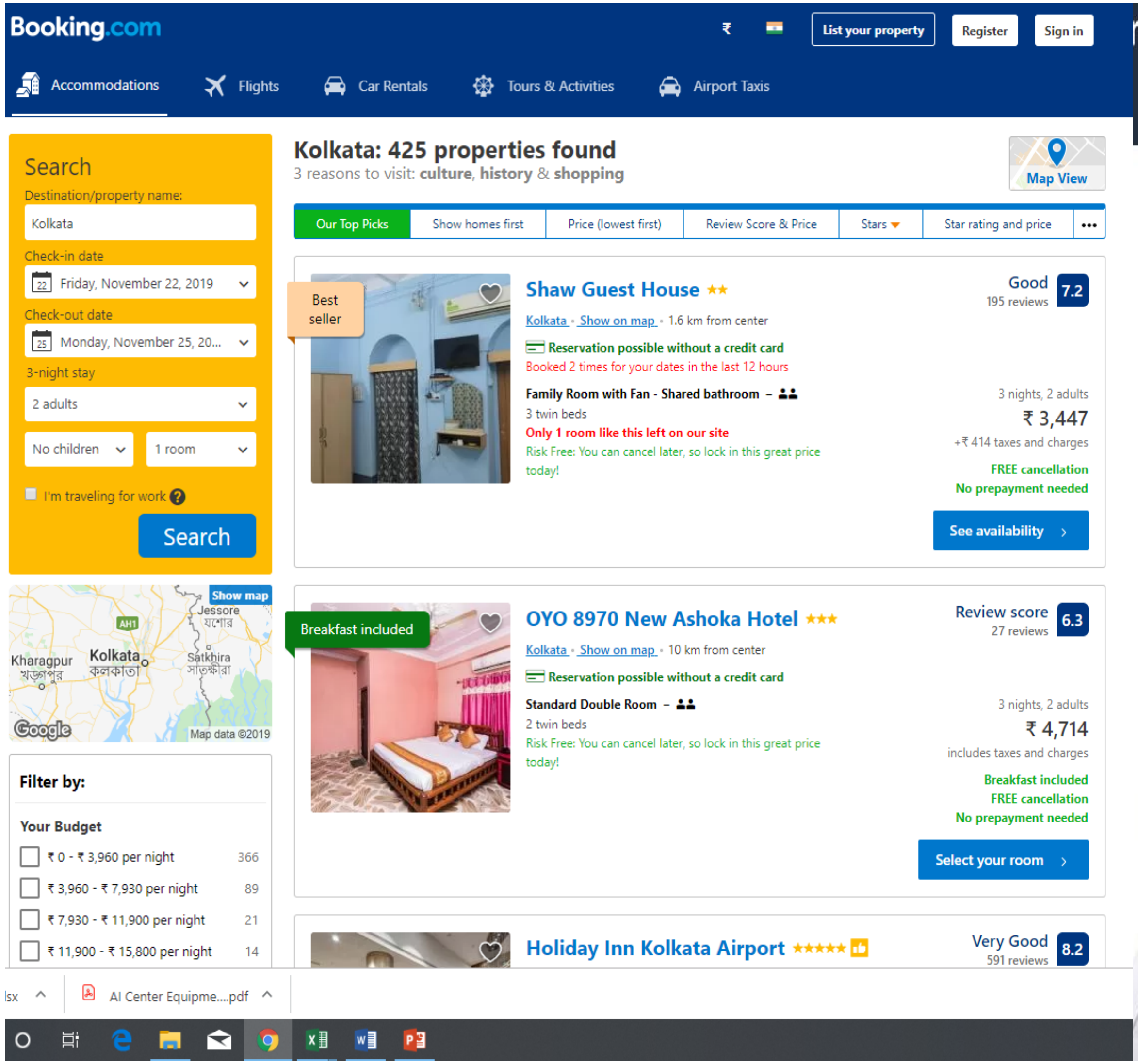
Linear, Regularized and Logistic Regression

AI42001 MLFA

Average Ratings



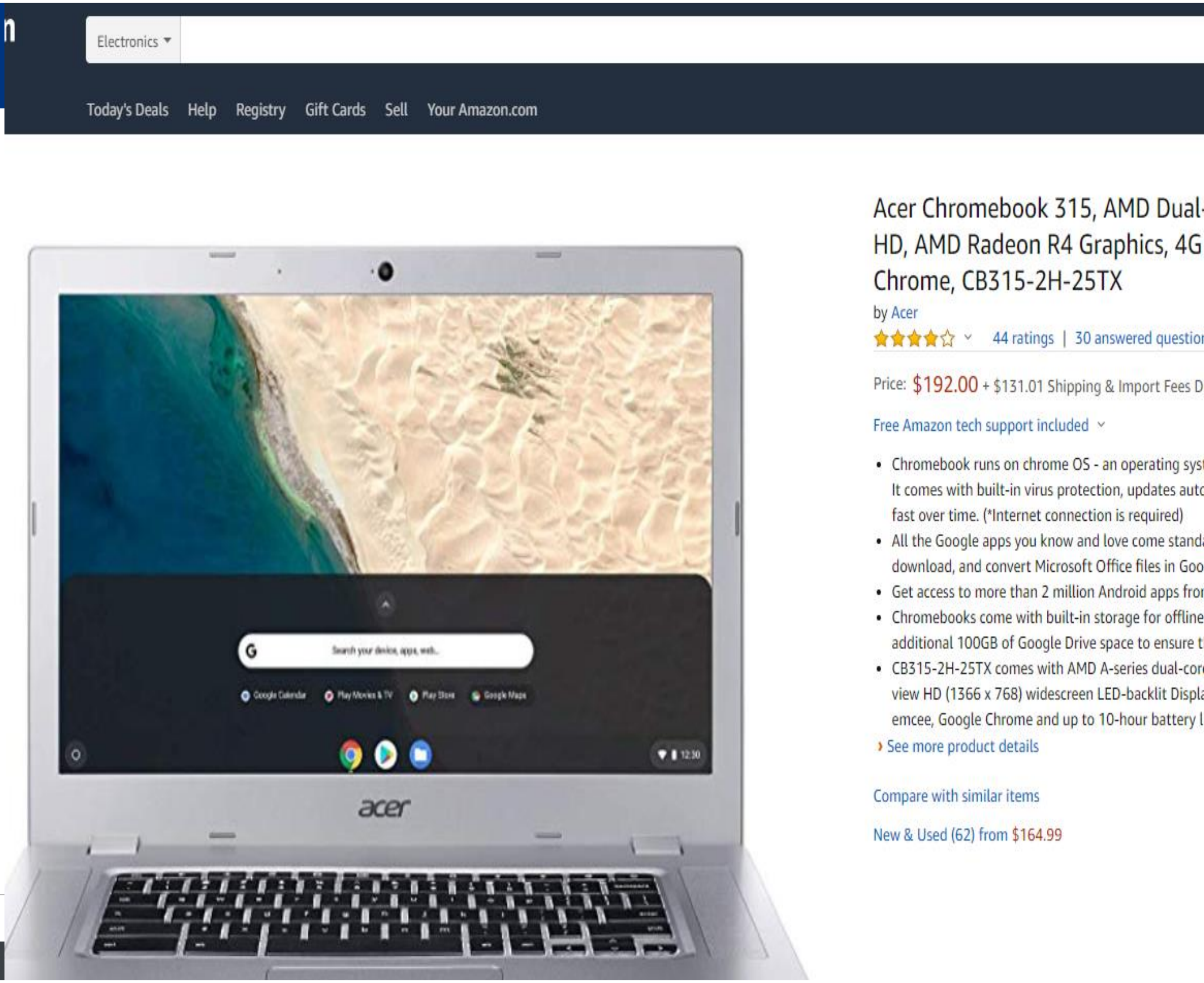
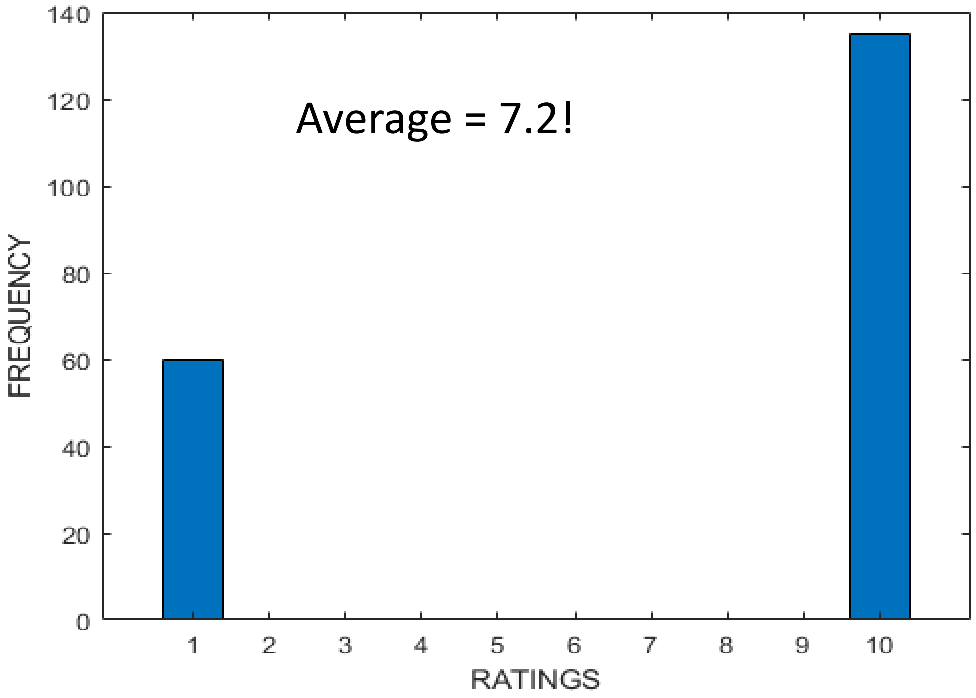
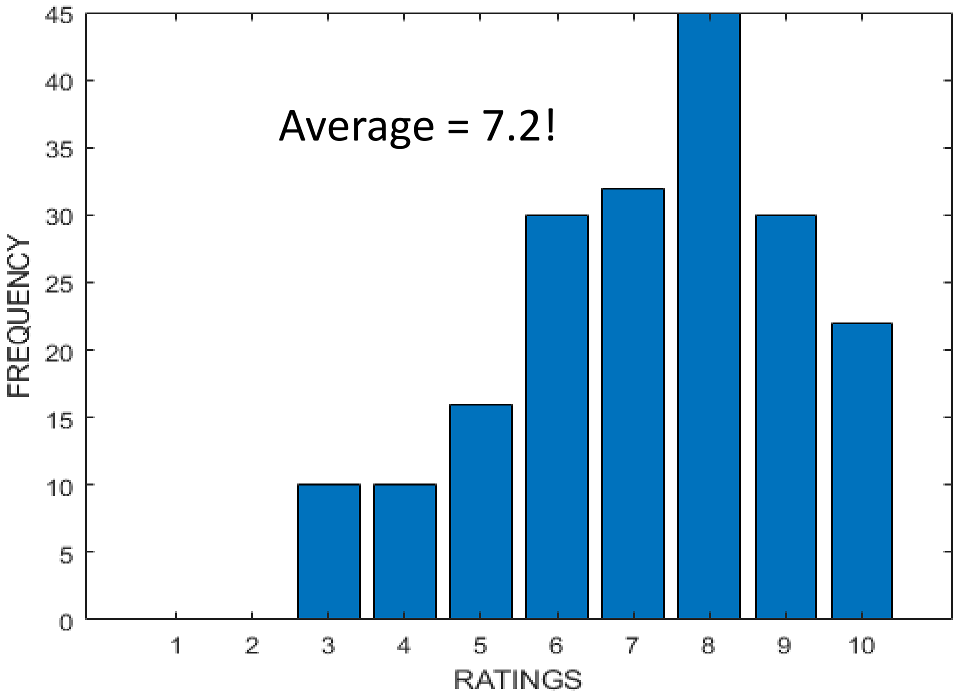


Image source: Google Images

Average Ratings

- 195 reviews, on a scale of 1 to 10
- Average rating: 7.2!
- There may be large or small variance among individual reviews





How do users rate a product?

Title:

Presenter:

Date: Time:

Your job classification: ___ Classified ___ Professional/Technical ___ Administrator ___ Faculty

Please circle the appropriate response for each statement:

	Excellent	Good	Fair	Poor
1. The relevance of this topic to me was	4	3	2	1
2. The usefulness of materials was	4	3	2	1
3. The effectiveness of the presenter was	4	3	2	1
4. I expect the future usefulness of this topic to be	4	3	2	1
5. My overall evaluation of this session is	4	3	2	1

Your Account > Packaging Feedback

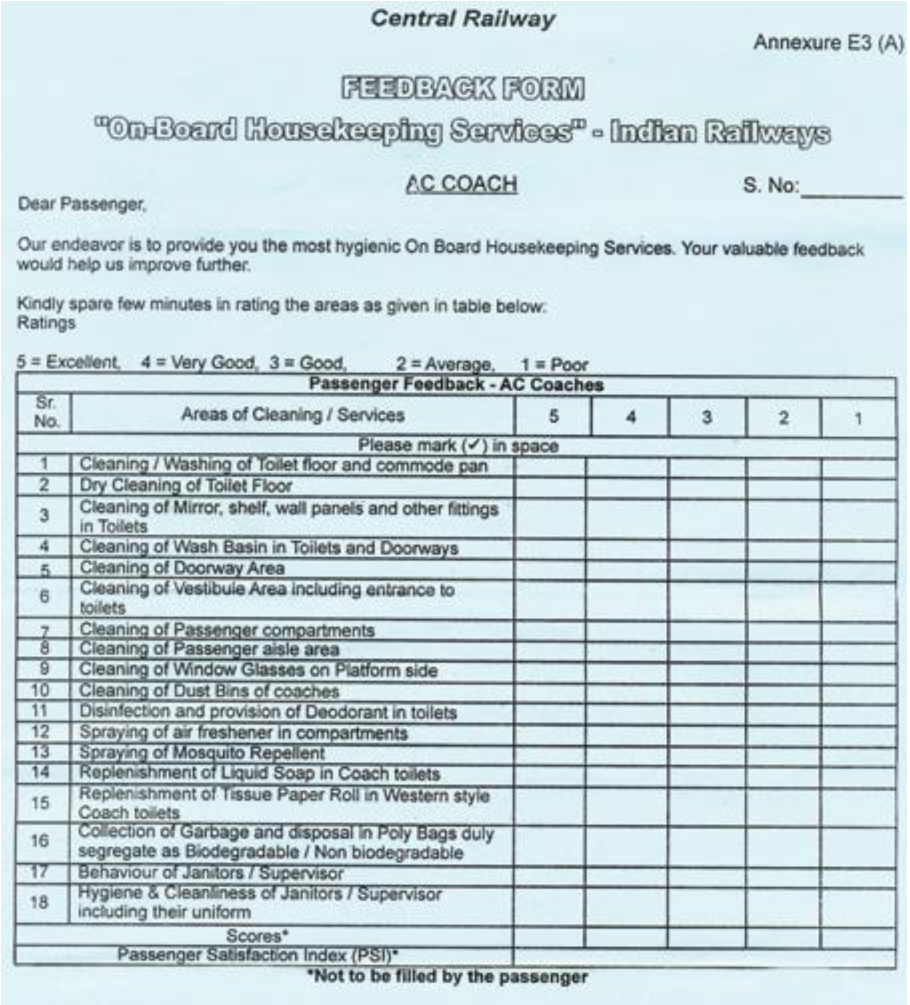
Rate Amazon's Packaging

Did the packaging protect your items adequately? — Protection — 1 star = Poor; 5 stars = Excellent

Was the box size and packaging appropriate for the items?
- Too Small
- About Right
- Too Big
- Way Too Big

Rate Item's Packaging

Ease of Opening — 1 star = Very Difficult; 5 stars = Very Easy



Central Railway

Annexure E3 (A)

FEEDBACK FORM

"On-Board Housekeeping Services" - Indian Railways

AC COACH

S. No:_________

Dear Passenger,

Our endeavor is to provide you the most hygienic On Board Housekeeping Services. Your valuable feedback would help us improve further.

Kindly spare few minutes in rating the areas as given in table below:
Ratings

5 = Excellent, 4 = Very Good, 3 = Good, 2 = Average, 1 = Poor

Passenger Feedback - AC Coaches						
Sr. No.	Areas of Cleaning / Services	5	4	3	2	1
	Please mark (✓) in space					
1	Cleaning / Washing of Toilet floor and commode pan					
2	Dry Cleaning of Toilet Floor					
3	Cleaning of Mirror, shelf, wall panels and other fittings in Toilets					
4	Cleaning of Wash Basin in Toilets and Doorways					
5	Cleaning of Doorway Area					
6	Cleaning of Vestibule Area including entrance to toilets					
7	Cleaning of Passenger compartments					
8	Cleaning of Passenger aisle area					
9	Cleaning of Window Glasses on Platform side					
10	Cleaning of Dust Bins of coaches					
11	Disinfection and provision of Deodorant in toilets					
12	Spraying of air freshener in compartments					
13	Spraying of Mosquito Repellent					
14	Replenishment of Liquid Soap in Coach toilets					
15	Replenishment of Tissue Paper Roll in Western style Coach toilets					
16	Collection of Garbage and disposal in Poly Bags duly segregate as Biodegradable / Non biodegradable					
17	Behaviour of Janitors / Supervisor					
18	Hygiene & Cleanliness of Janitors / Supervisor including their uniform					
	Scores*					
	Passenger Satisfaction Index (PSI)*					

*Not to be filled by the passenger

Image source: Google Images

How do users rate a product?

User 1:

Title:

Presenter:

Date: Time:

Your job classification: Classified Professional/Technical Administrator Faculty

Please circle the appropriate response for each statement:

	Excellent	Good	Fair	Poor	
1. The relevance of this topic to me was	4	3	2	1	3
2. The usefulness of materials was	4	3	2	1	3
3. The effectiveness of the presenter was	4	3	2	1	3
4. I expect the future usefulness of this topic to be	4	3	2	1	3
5. My overall evaluation of this session is	4	3	2	1	4

User 2:

Title:

Presenter:

Date: Time:

Your job classification: Classified Professional/Technical Administrator Faculty

Please circle the appropriate response for each statement:

	Excellent	Good	Fair	Poor	
1. The relevance of this topic to me was	4	3	2	1	3
2. The usefulness of materials was	4	3	2	1	5
3. The effectiveness of the presenter was	4	3	2	1	1
4. I expect the future usefulness of this topic to be	4	3	2	1	4
5. My overall evaluation of this session is	4	3	2	1	4

Image source: Google Images

How do users rate a product?

- Each product has D features (f_1, f_2, ... , f_D)
- The rating "y_i" given by any user "i" may be a weighted average of her scores (x_{i1}, x_{i2}, … , x_{iD}) on the individual features
- The weights (w_{i1}, w_{i2}, … , w_{iD}) may vary from one user to another according to their respective priorities
- Simplest model for user rating: $y_i = \sum_j w_{ij}x_{ij} + b_i$ (b_j: bias for user 'i')

How do users rate a product?

- Each product has D features (f_1, f_2, …. , f_D)
- The rating "y_i" given by any user "i" may be a weighted average of her scores (y_{i1}, y_{i2}, … , y_{iD}) on the individual features
- The weights (w_{i1}, w_{i2}, … , w_{iD}) may vary from one user to another according to their respective priorities
- Simplest model for user rating: $y_i = \sum_j w_{ij}\, y_{ij} + b_i$ (b_i: bias)
- Need to estimate the weights "w": M users x N features
- Too many parameters!!

How do users rate a product?

- Each product has N features (f_1, f_2, …. , f_N)
- The rating "y_i" given by any user "i" may be a weighted average of her scores (y_{i1}, y_{i2}, … , y_{iN}) on the individual features
- The weights (w_{i1}, w_{i2}, … , w_{iN}) may vary from one user to another according to their respective priorities
- Simplest model for user rating: $y_i = \sum_j w_{ij} y_{ij} + b_i$ (b_i: bias)
- Need to estimate the weights "w": M users x N features
- Too many parameters!!
- New approximate model: $y_i = \sum_j w_j x_{ij} + b$, i.e. all users have equal weights!

Linear Regression

- We know the feature scores "s_{ij}" and the final score "x_i"
- We want to find out the relative importance of the different features (on average)
- The answer: linear regression!
- General Recipe:

1) Define a model with parameter vector w
2) Define a measure on how well the model can fit the final scores
3) Choose the model parameters to improve this measure!

Linear Regression in one dimension

First, let us consider each product has only one feature

$\frac{dL}{dw} = 0 \implies 2\sum_i (y_i - wx_i - b)x_i = 0$

$\frac{dL}{db} = 0 \implies 2\sum_i (y_i - wx_i - b) = 0$

Solving these equations, we get

$b = \bar{y} - w\bar{x}$

$w = (\sum_i (\tilde{x}_i)^2)^{-1} (\sum_i \tilde{x}_i \tilde{y}_i)$

where $\bar{x} = \frac{1}{N}\sum_i x_i$, $\bar{y} = \frac{1}{N}\sum_i y_i$, $\tilde{x}_i = x_i - \bar{x}$

In [3]:

```
#initializing our inputs and outputs


#mean of our inputs and outputs
x_mean = np.mean(X)
y_mean = np.mean(Y)

#total number of values
n = len(X)

#using the formula to calculate the b1 and b0
numerator = 0
denominator = 0
for i in range(n):
    numerator += (X[i] - x_mean) * (Y[i] - y_mean)
    denominator += (X[i] - x_mean) ** 2

b1 = numerator / denominator
b0 = y_mean - (b1 * x_mean)

#printing the coefficient
print(b1, b0)
```

Linear Regression in Higher Dimensions

- The model in this case: $h_i = \sum_j w_j x_{ij} + w_0$ (h_i: predicted rating)
- Measurement of fit: squared error loss function
- Bias 'w_0' can be absorbed into 'w', by considering a special feature whose value is always 1)
- $L(y_i, h_i) = (y_i - h_i)^2 = (y_i - \sum_j w_j x_{ij})^2$

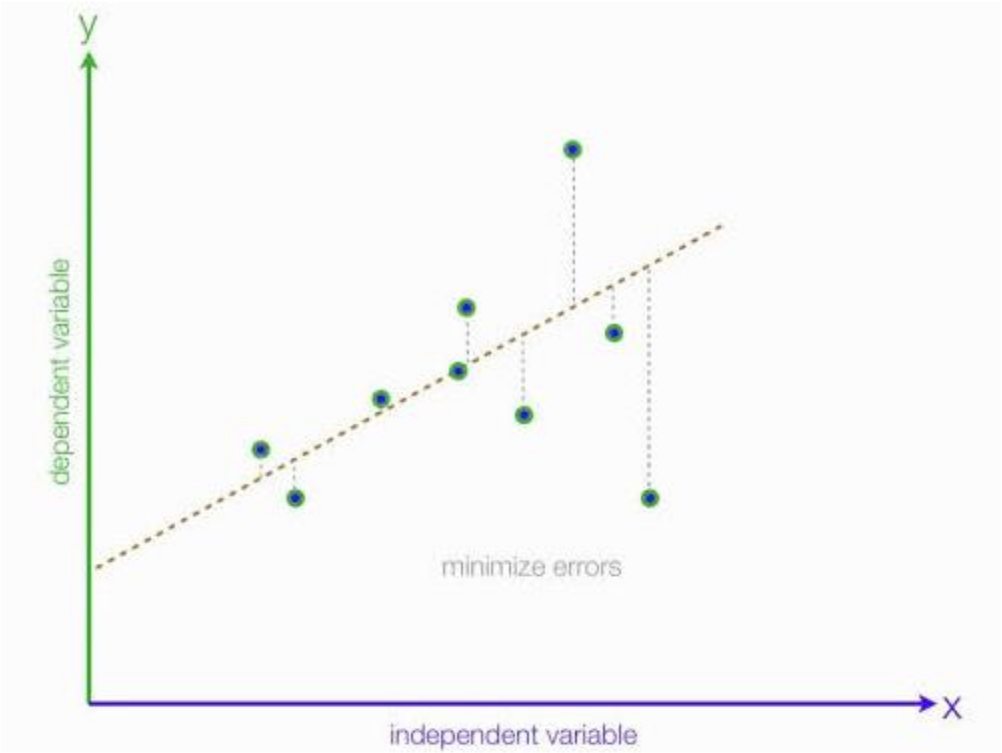


Image source: Google Images

Linear Regression in Higher Dimensions

- The model in this case: $h_i = \sum_j w_j x_{ij} + w_0$ (h_i: predicted rating)
- Measurement of fit: squared error loss function
- Bias 'w_0' can be absorbed into 'w', by considering a special feature whose value is always 1)
- Loss for user i: $L(y_i, h_i) = (y_i - h_i)^2 = (y_i - \sum_j w_j x_{ij})^2 = (y_i - w^T X_i)^2$
- Choose w to minimize total loss $\sum_i L(y_i, h_i)$ over all M users!
- Differentiate the total loss w.r.t. each variable in w, equate to 0, and solve an equation!

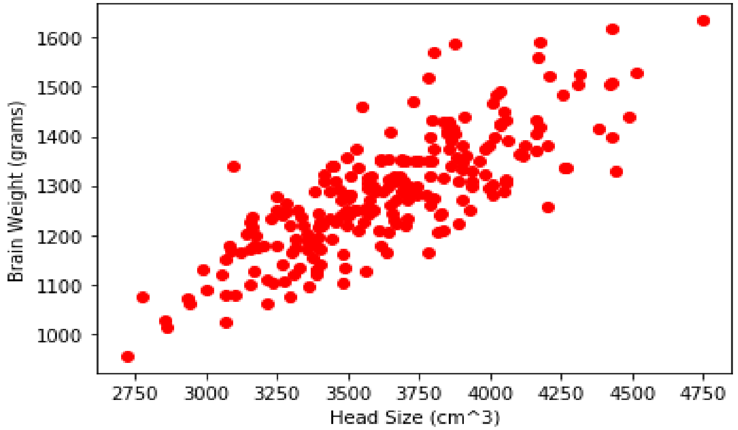
Python Implementation

In [2]:

```python
#import libraries
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

#reading data
dataset = pd.read_csv('dataset.csv')
print(dataset.shape)
dataset.head()

X = dataset['Head Size(cm^3)'].values
Y = dataset['Brain Weight(grams)'].values

#plot the data point
plt.scatter(X, Y, color='#ff0000', label='Data Point')

# x-axis label
plt.xlabel('Head Size (cm^3)')

#y-axis label
plt.ylabel('Brain Weight (grams)')
```

```
(237, 4)
```

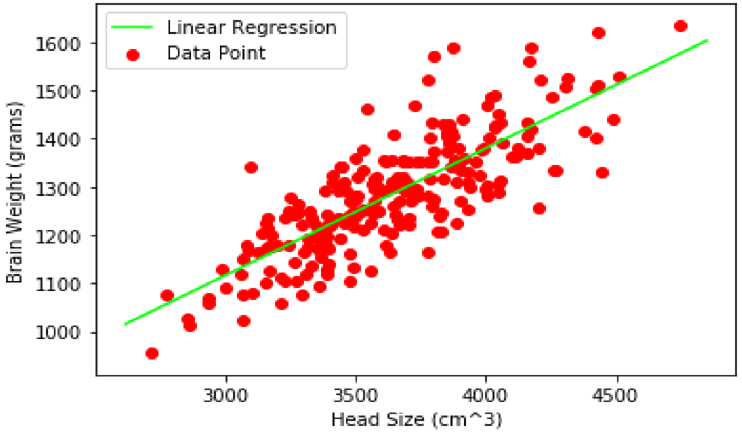
Out[2]: `Text(0, 0.5, 'Brain Weight (grams)')`




```python
#mean of our inputs and outputs
x_mean = np.mean(X)
y_mean = np.mean(Y)

#total number of values
n = len(X)

#using the formula to calculate the b1 and b0
numerator = 0
denominator = 0
for i in range(n):
    numerator += (X[i] - x_mean) * (Y[i] - y_mean)
    denominator += (X[i] - x_mean) ** 2

b1 = numerator / denominator
b0 = y_mean - (b1 * x_mean)

#printing the coefficient
print(b1, b0)
```

In [3]:

```python
#plotting values
x_max = np.max(X) + 100
x_min = np.min(X) - 100

#calculating line values of x and y
x = np.linspace(x_min, x_max, 1000)
y = b0 + b1 * x

plt.plot(x, y, color='#00ff00', label='Linear Regression') #plotting line
plt.scatter(X, Y, color='#ff0000', label='Data Point') #plot the data point
plt.xlabel('Head Size (cm^3)') # x-axis label
plt.ylabel('Brain Weight (grams)') #y-axis label

plt.legend()
plt.show()
```



Linear Regression: Vector Form

$$\mathcal{L}(w) = \sum_{i=1}^{N} (y_i - w^T x_i)^2 \quad [\text{LEAST SQUARE LOSS FN}]$$

$$\frac{\partial \mathcal{L}(w)}{\partial w} = \sum_{i=1}^{N} 2(y_i - w^T x_i)(-x_i) = 0 \quad [\text{FOR MINIMA}]$$

$$\Rightarrow \sum_{i=1}^{N} x_i y_i = \sum_{i=1}^{N} (w^T x_i) x_i$$

$$= \sum_{i=1}^{N} (x_i x_i^T) w \quad [\text{verify!}]$$

$$\Rightarrow w = \left[\sum_{i=1}^{N} x_i x_i^T\right]^{\dagger} \left(\sum_{i=1}^{N} x_i y_i\right)$$

$\dagger$: pseudo-inverse of matrix

Average Rating Prediction

- Given a new product, we need to predict it's "average rating"
- Average rating = $\mathit{mean}_i(y_i)$
- According to LR model:
- predicted average rating = $\mathit{mean}_i(h_i)$
- $= \mathit{mean}_i(\sum_j w_j x_{ij} + w_0) = \sum_j w_j \, \mathit{mean}_i(x_{ij}) + w_0$
- We have the weights "w_j" of its features and bias "w_0", by linear regression for <u>similar products</u>
- We can find the average user ratings of each feature $\mathit{mean}_i(x_{ij})$, based on <u>other products having same feature</u>

Average Rating Prediction

- New Product: a new camera model
- Features: resolution, battery life, memory, flash, weight, size
- Weights of features: calculate by linear regression from user ratings on other cameras
- New camera resolution: 5 MP
- Average rating on resolution: 4.0
- Weight of resolution: 0.54

Model	Resolution	Mean feature rating
Camera1	5 MP	4.1
Camera2	5 MP	3.9
Camera3	10 MP	4.4
Camera4	12 MP	4.1
Camera5	6 MP	4.0
Camera6	15 MP	4.3

Average Rating Prediction

- New Product: a new camera model
- Features: resolution, battery life, memory, flash, weight, size
- Weights of features: calculate by linear regression from user ratings on other cameras
- New camera battery life: 2 years
- Average rating on battery life : 3.8
- Weight of battery life : 0.36

Model	Battery Life	Mean feature rating
Camera1	3 years	4.5
Camera2	2 years	3.6
Camera3	2 years	3.8
Camera4	1 year	3.1
Camera5	2 years	3.9
Camera6	3 years	4.3

Average Rating Prediction

- New Product: a new camera model
- Features: resolution, battery life, memory, flash, weight, size
- Weights of features: calculate by linear regression from user ratings on other cameras
- New camera memory: 5 GB
- Average rating on memory: 4.5
- Weight of memory: 0.10

- Predicted average rating

$= 0.54*4.0 + 0.36*3.8 + 0.1*4.5 =$ 4.0!

Model	Memory	Mean feature rating
Camera1	1 GB	3.8
Camera2	1 GB	3.9
Camera3	2 GB	4.1
Camera4	3 GB	4.0
Camera5	5 GB	4.4
Camera6	5 GB	4.5

Feature Selection

- Linear regression model: $y_i = \sum_j w_j x_{ij}$, i.e. all feature ratings contribute to the final rating (including the bias that is absorbed into ‘w’)
- But in the examples, only a small number of features seem to influence the final rating, other features have little importance
- In case 1: One element in “w” will have high value, other elements will have small values
- In case 2: All elements except one in “w” have 0 value, i.e. “w” is sparse!

Feature Selection

- Feature selection: the task of identifying the “important” features
- Important feature: those which strongly influence the final ratings
- In the given examples, feature selection is easy by manual inspection
- Large dataset: many examples, many dimensions, noisy ratings, manual inspection impossible
- Can linear regression itself solve the feature selection problem?
- It can, if it returns a suitable “w”!

Sparse Regression for Feature Selection

- Case 1: we want “w” such that most of its elements are small
- Case 2: we want “w” such that most of its elements are 0
- Can we convert these demands into mathematical formulations?

Sparse Regression for Feature Selection

- Case 1: we want "w" such that most of its elements are small
- Case 2: we want "w" such that most of its elements are 0
- Can we convert these demands into mathematical formulations?
- General recipe: find a regularization function f(w)
- f(w) should have low value for suitable "w", high value for unsuitable "w"

Sparse Regression for Feature Selection

- Case 1: we want “w” such that most of its elements are small
- Case 2: we want “w” such that most of its elements are 0
- Can we convert these demands into mathematical formulations?
- General recipe: find a regularization function $h(w)$
- $f(w)$ should have low value for suitable “w”, high value for unsuitable “w”

- Find (w,b) to minimize $L(w,b) + \lambda h(w)$
- First term to find w that fits data, second term to find “w” that is suitable, λ to balance them!

Ridge Regression

- f(w) : how to choose?
- Simplest $h(w) = ||w||_2^2$
- The L_2-norm of vector "w", $||w||_2^2$

  $= w^T w = \sum_j w_j^2$
- Limits the distance of "w" from origin

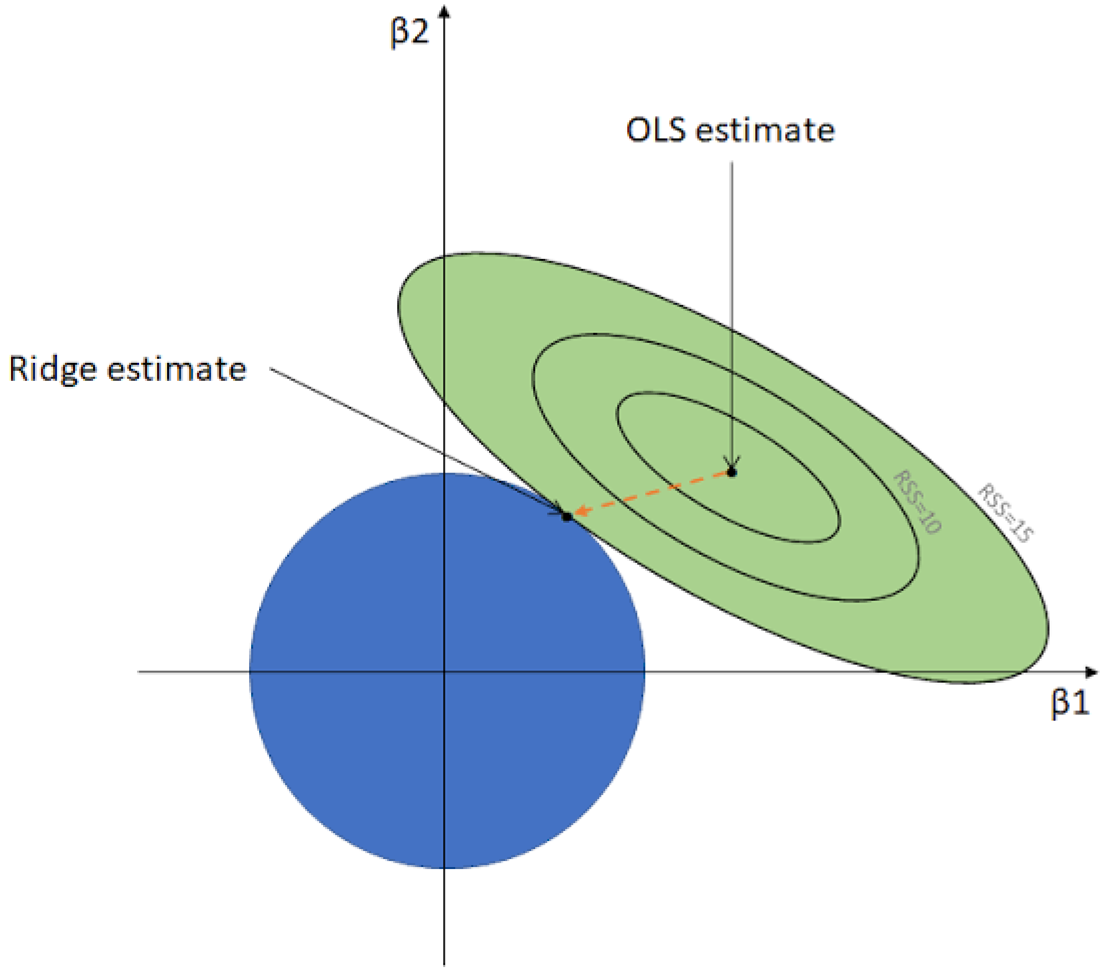


Image source: Google Images

Ridge Regression

- f(w) : how to choose?
- Simplest $h(w) = ||w||_2^2$
- The L2-norm of vector "w", $||w||_2^2$

  $= w^T w = \sum_j w_j^2$
- Limits the distance of "w" from origin

  i.e. constrains the different dimensions
- Low value of $||w||_2^2$ indicates that all features will have restricted weights.
- Popularly known as "ridge regression"

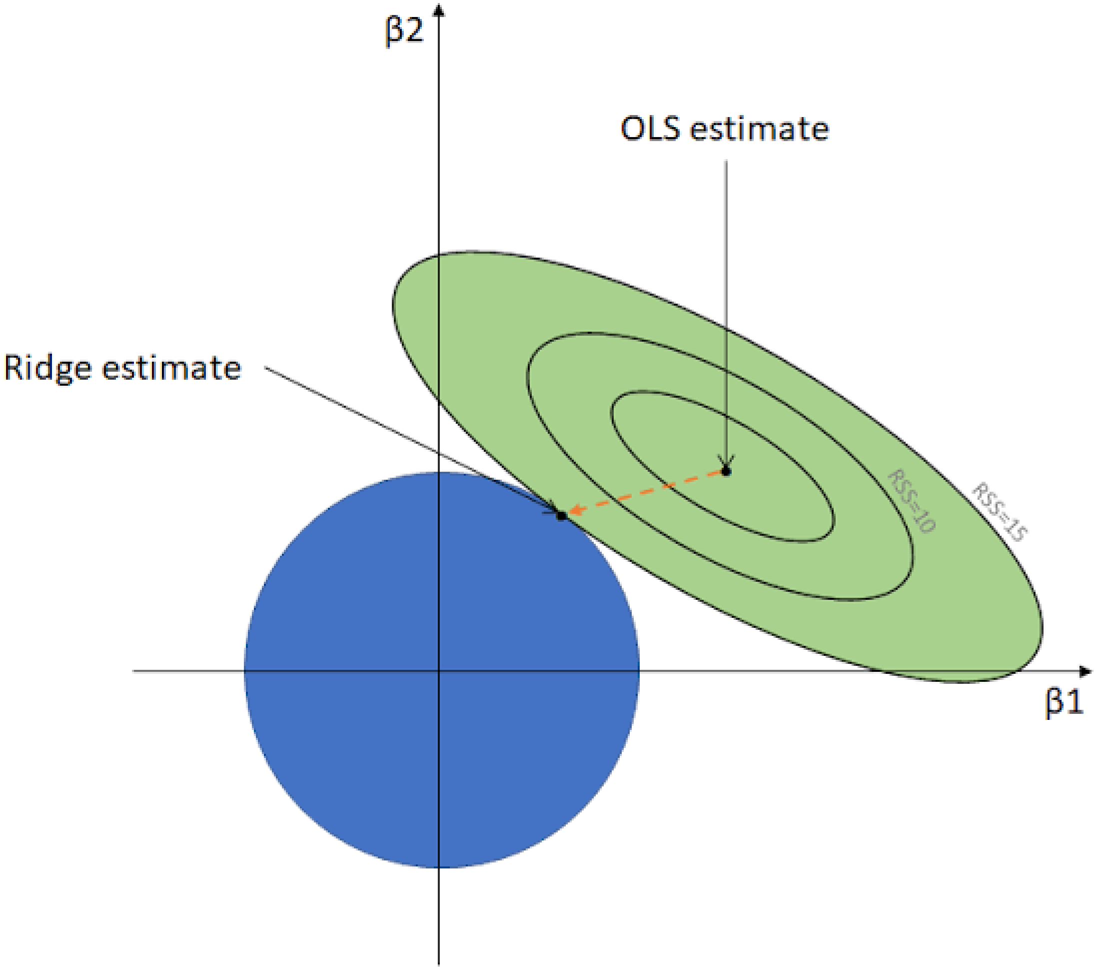


Image source: Google Images

Ridge Regression: Mathematics

$$\mathcal{L}(W) = \sum_{i=1}^{N} (y_i - W^T x_i)^2 + \lambda W^T W \qquad W^T W = \|W\|_2^2$$

$$\frac{\partial \mathcal{L}(w)}{\partial W} = \sum_{i=1}^{N} 2(y_i - W^T x_i)(-x_i) + 2\lambda W = 0 \quad \text{[FOR MINIMA]}$$

$$\Rightarrow \sum_{i=1}^{N} x_i y_i = \sum_{i=1}^{N} (w^T x_i) x_i + \lambda W$$

$$= \sum_{i=1}^{N} (x_i x_i^T) W + \lambda W = \left[\sum_{i=1}^{N} x_i x_i^T + \lambda I\right] W$$

$$\Rightarrow W_{RIDGE} = \left[\sum_{i=1}^{N} x_i x_i^T + \lambda I\right]^{\dagger} \left(\sum_{i=1}^{N} x_i y_i\right)$$

$\dagger$: pseudo-inverse of matrix

λ : trade-off parameter

The role of λ-parameter

- λ decides the relative importance of fitting error and regularizer
- Small value of λ: regularization not important!

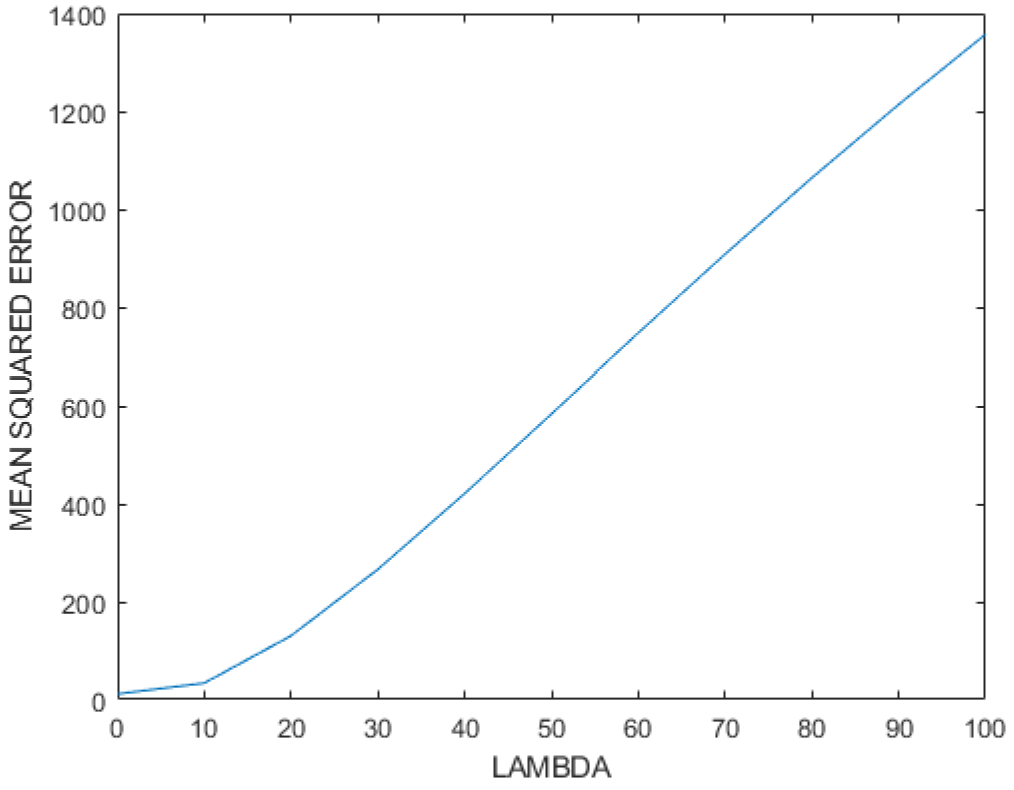
  low error, "w" vector may contain large values!

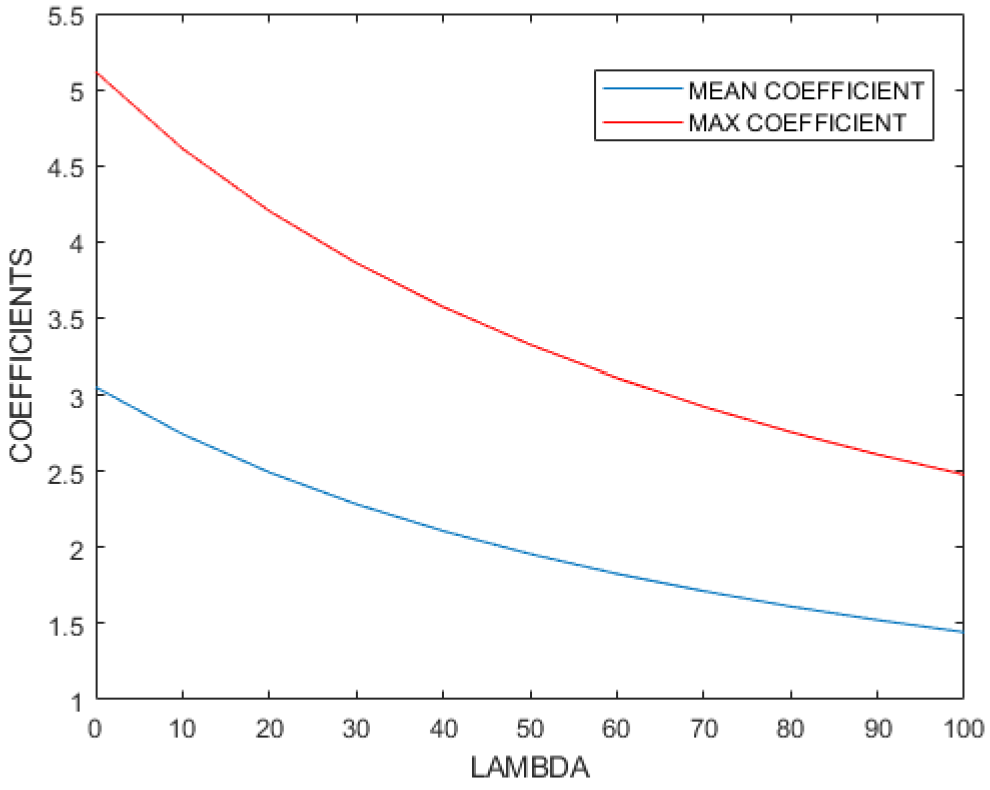
  result similar to linear regression!
- Large value of λ: fitting error not important!

  high error, but "w" contains small values

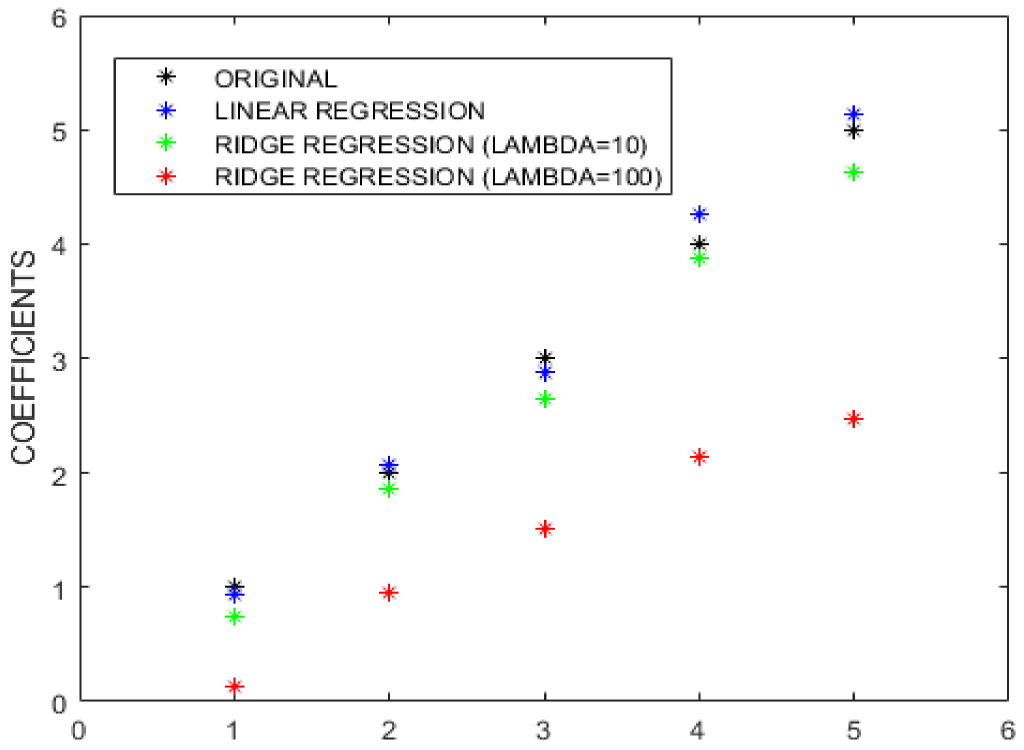
  result different from linear regression!

The role of λ-parameter

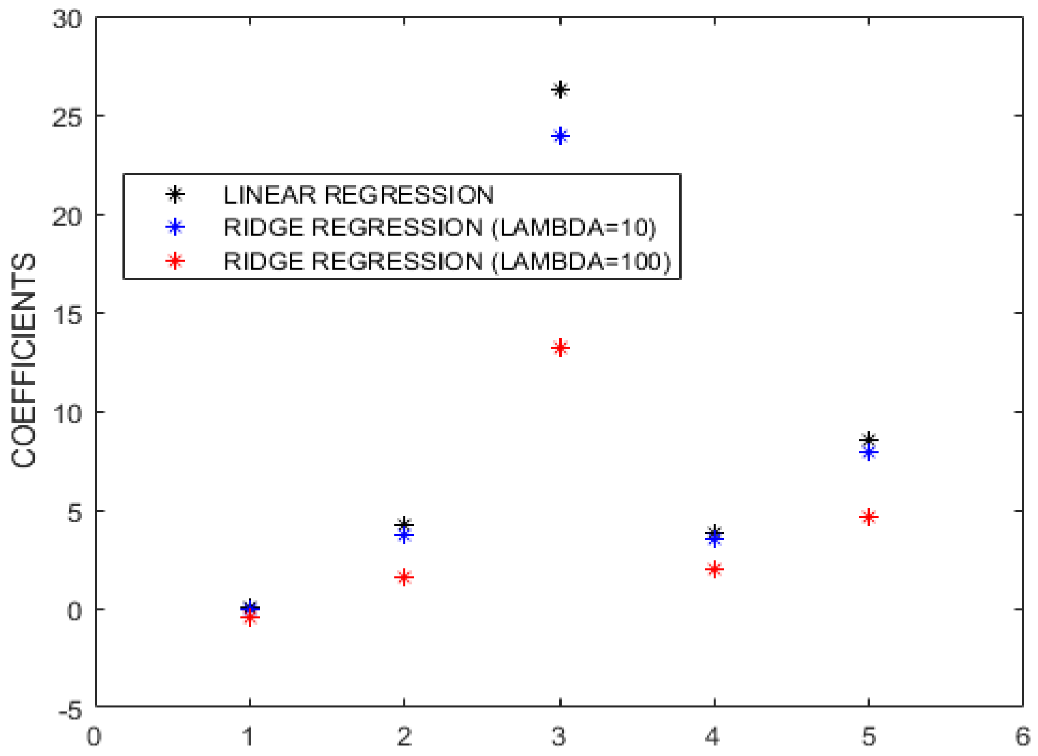




Ridge Regression vs Linear Regression



Original function: linear



Original function: non-linear

LASSO regression

- Our original aim: “sparse w”!
- The L_0-norm of vector “w”: number of non-zero elements
- Regularizer $f(w) = ||w||_0$ promotes sparse “w”!
- New problem: $L(w,b) + \lambda h(w)$
- Non-differentiable function!!!

LASSO regression

- Our original aim: “sparse w”!
- The L_0-norm of vector “w”: number of non-zero elements
- Regularizer $f(w) = ||w||_0$ promotes sparse “w”!
- New problem: $L(w,b) + \lambda h(w)$
- Non-continuous function!!!
- Relaxation: $f(w) = ||w||_1 = \sum_j |w_j|$ = sum of absolute values of elements!
- Low value of $||w||_1$: most values of w “close to 0”
- “Almost sparse” w!

LASSO vs Ridge Regression

- Both are compromise between squared loss minimization and feasible region

  Feasible region shape different in both cases

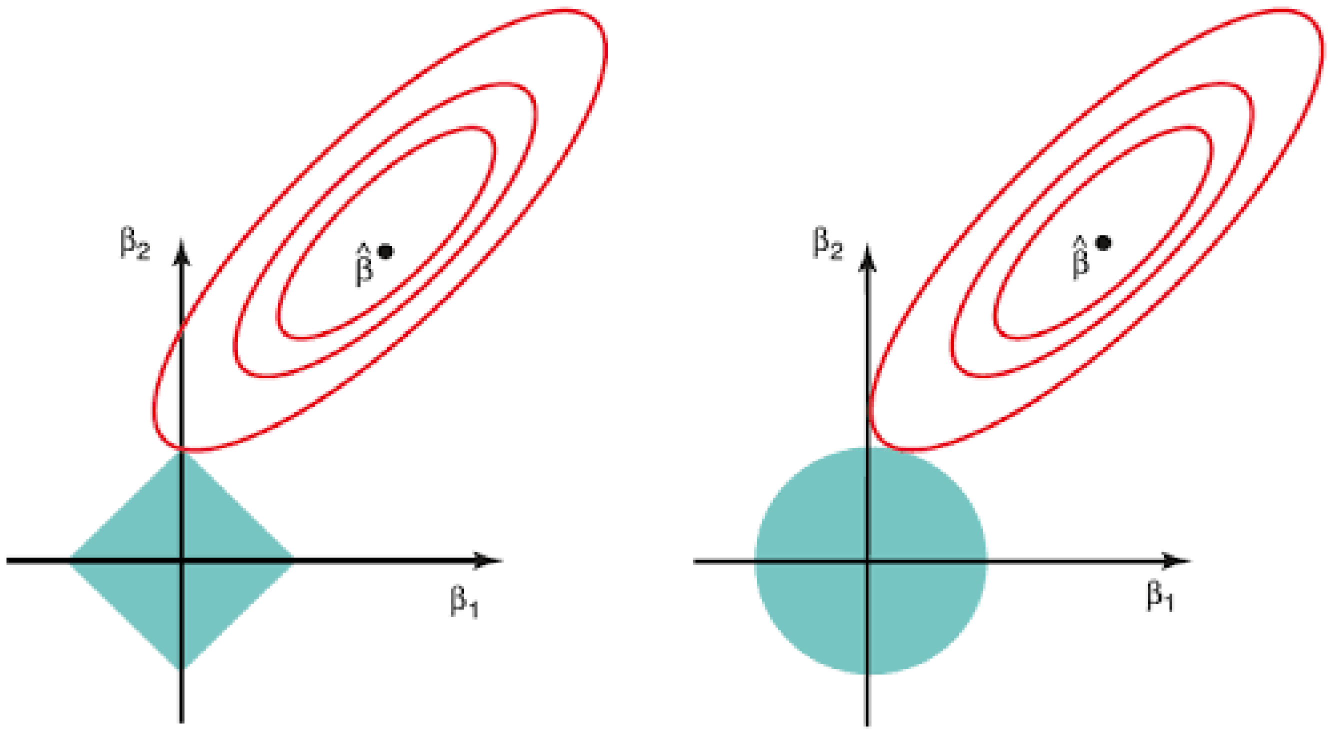


Image source: Google Images

LASSO regression

- Objective function: $\Sigma_i(y_i - w^T x_i)^2 + \lambda||w||_1$
- Difficult to solve by differentiation!

- Alternative: use numerical method instead of analytical!
- Gradient Descent: to be covered later!

Python Implementation using sklearn

In [64]:

```python
TrainX=np.asarray(X)
TrainY=np.asarray(Y)

type(NewX)
```

Out[64]: numpy.ndarray

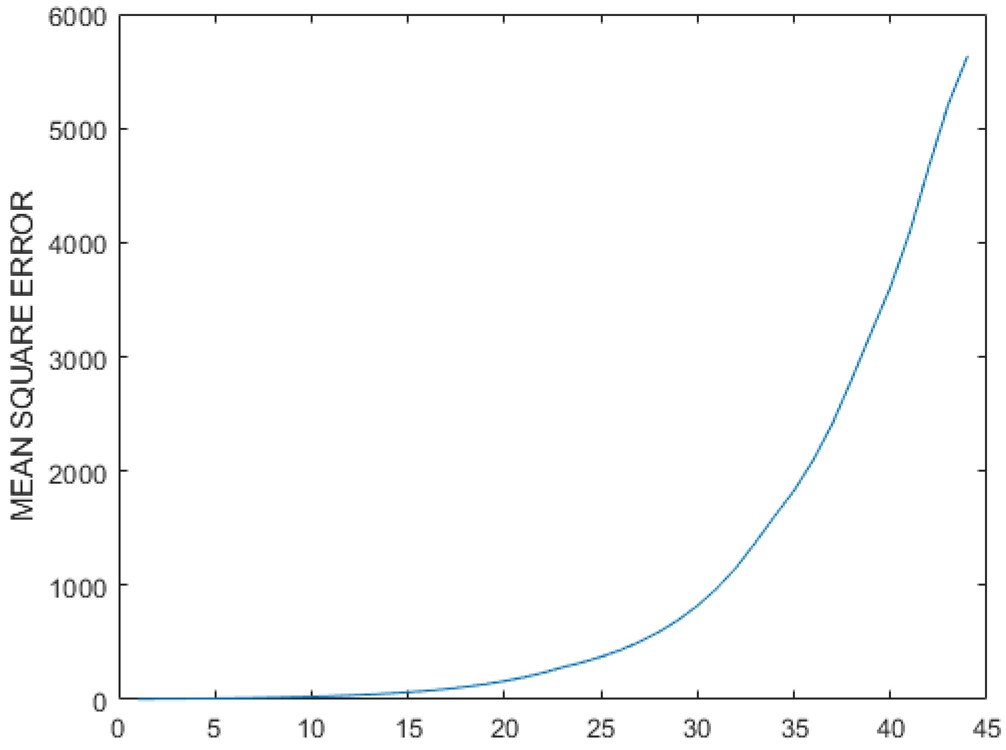
In [0]:

```python
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model  import Lasso
from sklearn.linear_model  import Ridge
```

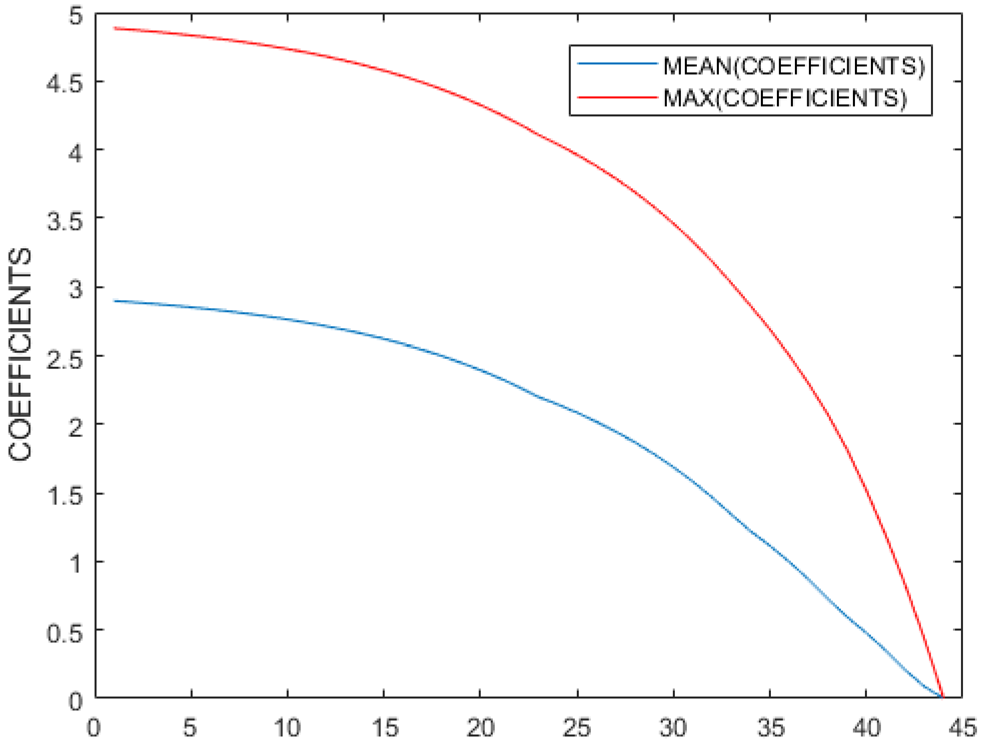
In [73]:

```python
lasso=Lasso()
parameters={'alpha': [0.001,0.01,0.1, 0.5,1]}
lassoReg=GridSearchCV(lasso,parameters,scoring='neg_mean_squared_error',cv=3)    #using gridsearch for cross validation
lassoReg.fit(TrainX.reshape(-1,1),TrainY.reshape(-1,1))    # training

ridge=Ridge()
parameters={'alpha': [0.1, 0.5,1]}
ridgeReg=GridSearchCV(ridge,parameters,scoring='neg_mean_squared_error',cv=3)    #using gridsearch for cross validation
ridgeReg.fit(TrainX.reshape(-1,1),TrainY.reshape(-1,1))    # training
```

LASSO regression



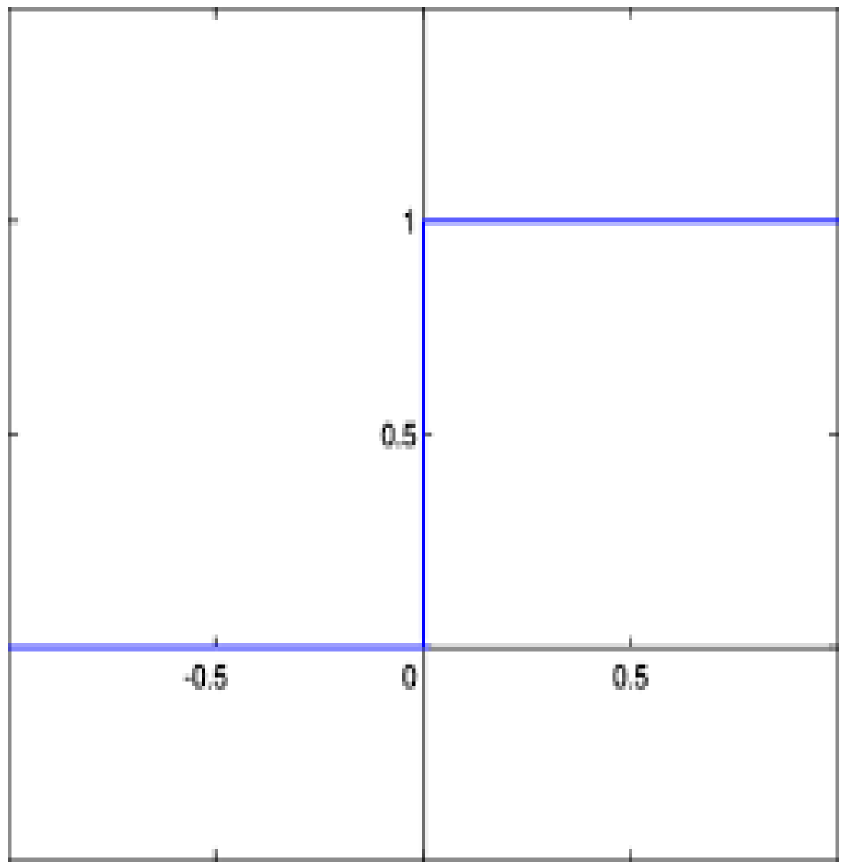
LAMBDA



LAMBDA

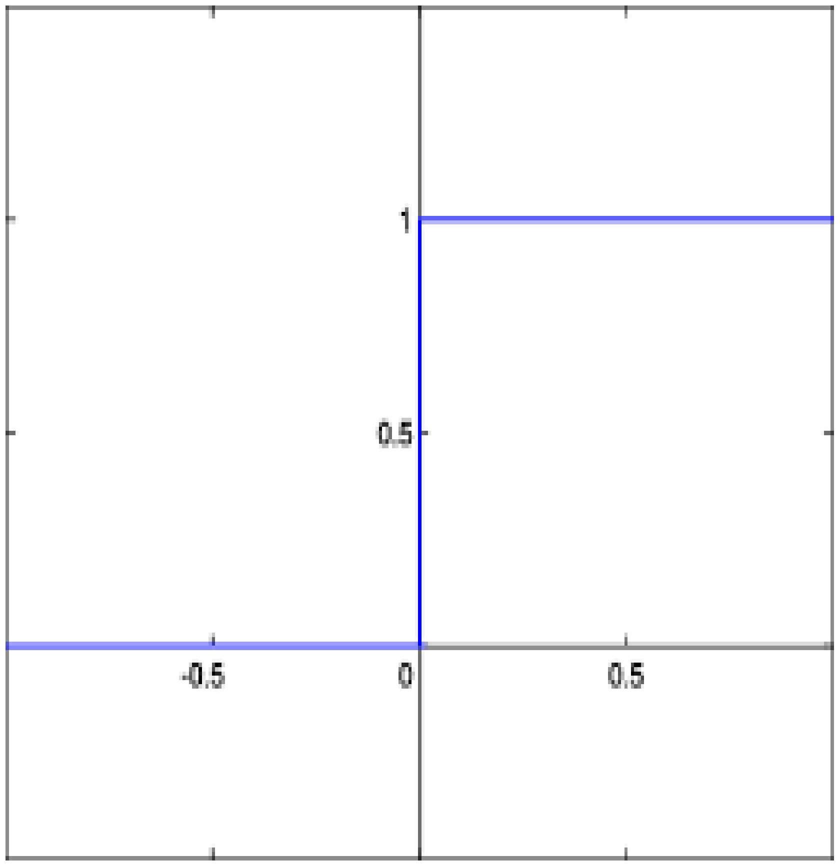
Threshold-based classification

For numeric (integer or real-valued) features, threshold is a simple classifier



$$P(Y = 1 \mid X) = 1 \text{ if } X > 0$$
$$= 0 \text{ if } X < 0$$

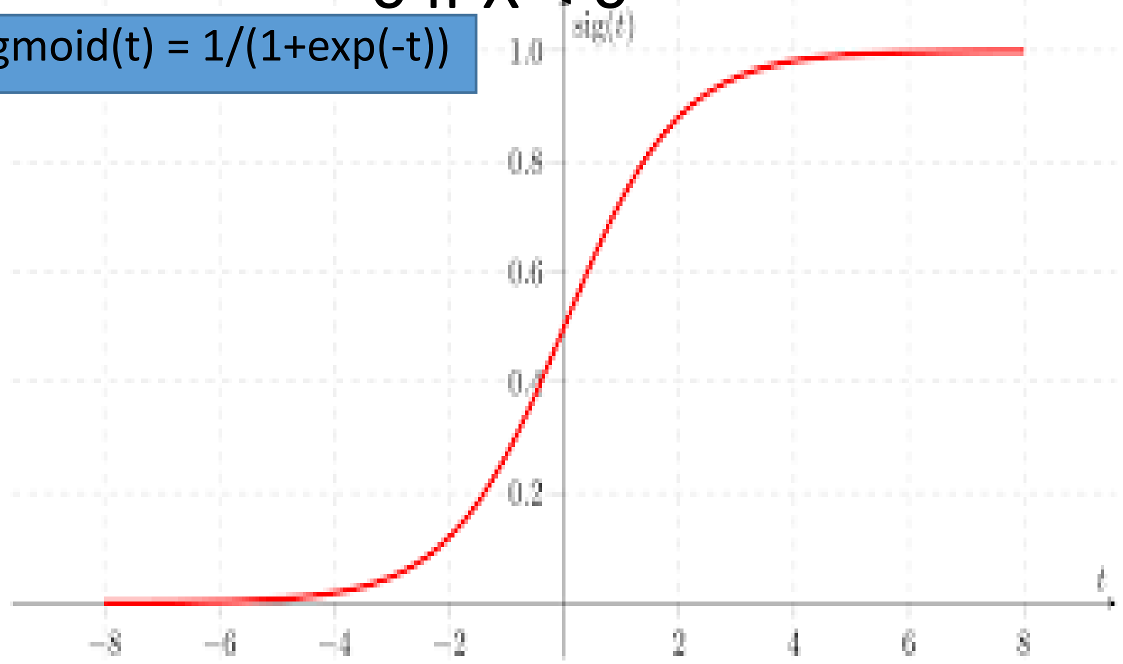
Logistic Regression for Classification



P(Y = 1 | X) = 1 if X > 0

= 0 if X < 0

sigmoid(t) = 1/(1+exp(-t))



Logistic Regression

- $P(Y = 1 \mid X) = 1$ if $X > 0$

  $= 0$ if $X < 0$
- Approximation: $p(Y = 1 \mid X) = 1/(1 + \exp(-X)) = \sigma(X)$
- Multi-dimensional features: consider weighted combination w.X
- $P(Y = 1 \mid X) = 1/(1 + \exp(-w.X)) = \sigma(w.X)$ LOGISTIC

Logistic Regression

- $P(Y = 1 \mid X) = 1$ if $X > 0$

  $= 0$ if $X < 0$

- Approximation: $p(Y = 1 \mid X) = 1/(1 + \exp(-X))$

- Multi-dimensional features: consider weighted combination w.X

- $P(Y = 1 \mid X) = 1/(1 + \exp(-w.X))$ LOGISTIC REGRESSION!
- But how to find w?

Logistic Regression Loss Function

- Logistic regression: probabilistic binary classification
- prob(y=1 | x) = σ(w.x) = 1/(1 + exp(-w.x))   [w: model parameters]
- prob(y=0 | x) = 1 - prob(y=1 | x)

- Loss function: -y*log(σ(w.x)) – (1-y)* log(1- σ(w.x))
- Why is this a valid loss function????

Logistic Regression Loss Function

- Logistic regression: probabilistic binary classification
- prob(y=1 | x) = σ(w.x) = 1/(1 + exp(-w.x))   [w: model parameter]
- prob(y=0 | x) = 1 - prob(y=1 | x)

- Loss function: -y*log(σ(w.x)) – (1-y)* log(1- σ(w.x))
- Why is this a valid loss function????

	y=1	y=0
	-log(σ(w.x))	-log(1- σ(w.x))

Logistic Regression Loss Function

- Logistic regression: probabilistic binary classification
- prob(y=1 | x) = σ(w.x) = 1/(1 + exp(-w.x))   [w: model parameters]
- prob(y=0 | x) = 1 - prob(y=1 | x)

- Loss function: -y*log(σ(w.x)) – (1-y)* log(1- σ(w.x))
- Why is this a valid loss function????

	y=1	y=0
σ(w.x) = 0.99	-log(0.99) = 0	-log(0.01) = 4.6
σ(w.x) = 0.01	-log(0.01) = 4.6	-log(0.99) = 0

Logistic Regression for Multi-classification

- $P(Y=1 \mid X) = 1/(1+\exp(-w_1.x))$
- $P(Y=2 \mid X) = 1/(1+\exp(-w_2.x))$ ………….

- $P(Y=K \mid X) = 1/(1+\exp(-w_k.x))$

- $w_1, w_2, \ldots.\ w_K$ are the model parameters
- New loss function: $-\Sigma_i I(y_i=k)\log(p(yi=k|X))$ [cross-entropy for k-dim. PMF]

- Estimation of W: minimize the loss function by gradient descent!

Numerical Approach

- In some cases, analytical approach does not work
- Reasons:

  1) loss function not differentiable

     e.g. 0-1 loss function

  2) derivative equations cannot be solved

     e.g. Loss function for logistic regression

- In such cases, we need numerical approach!

Numerical Approach

- Numerical approach to minimizing any function f:

  1) Start with any initial point x0

  2) Move to point x1 = x0 – a.f’(x0) /// f’(x0): derivative of f(x) at x=x0

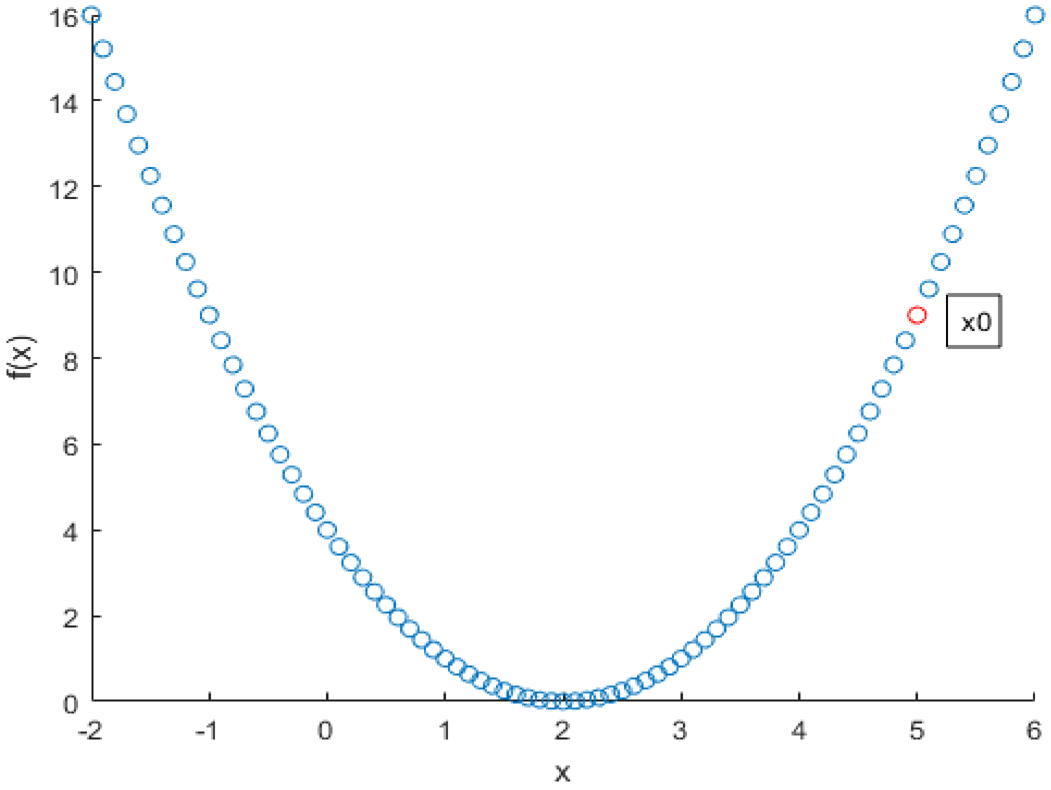
     /// a: constant learning rate

  3) If x1 = x0, STOP /// x0 is a minima of function f

     else set x0 = x1, GOTO 2

Numerical Approach

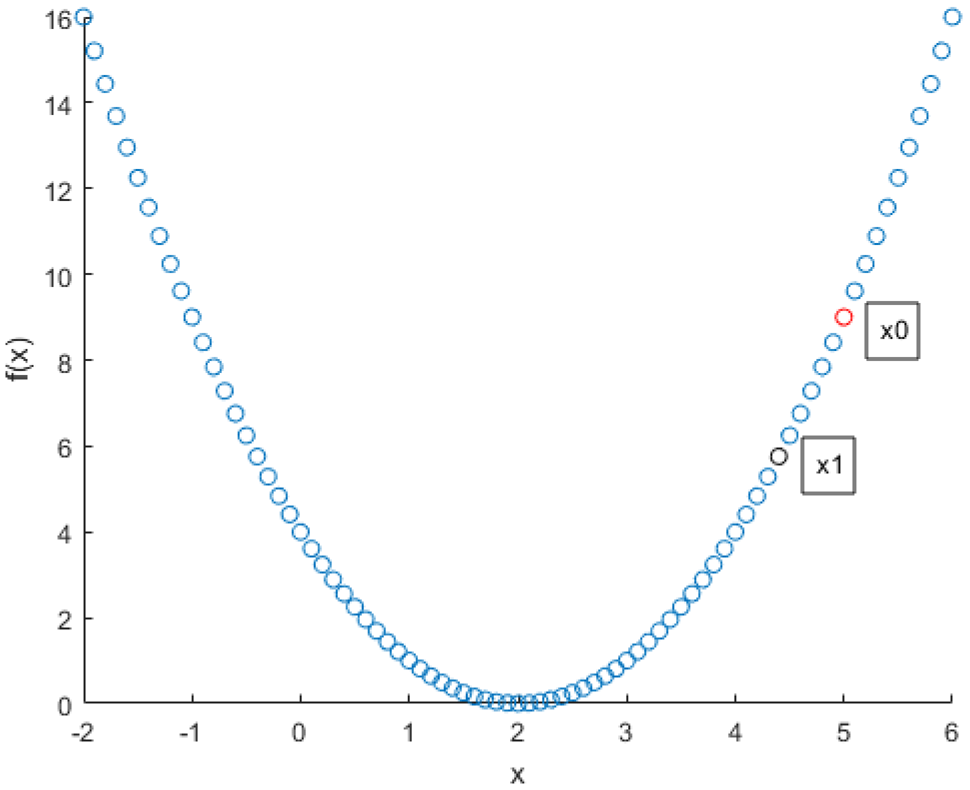
- Consider $f(x) = x^2 - 4x + 4$; $f'(x) = 2x - 4$
- Set initial point x=5, learning rate a = 0.1



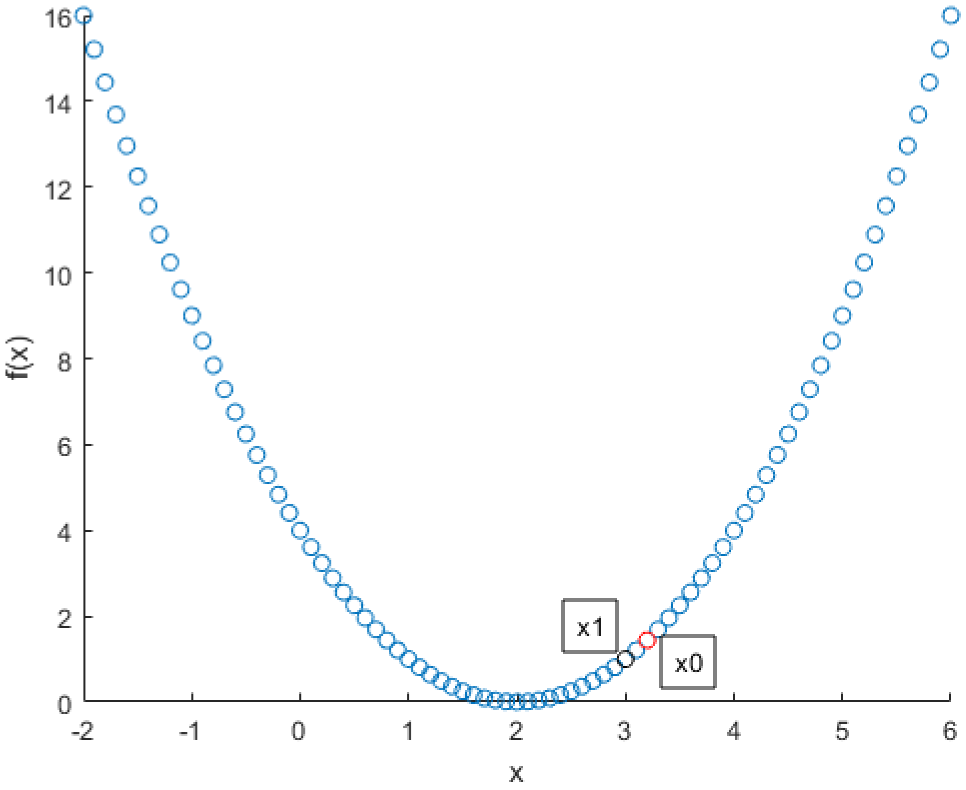
INITIALIZATION

Numerical Approach

- Consider $f(x) = x^2 - 4x + 4$; $f'(x) = 2x - 4$
- Set initial point x=5, learning rate a = 0.1



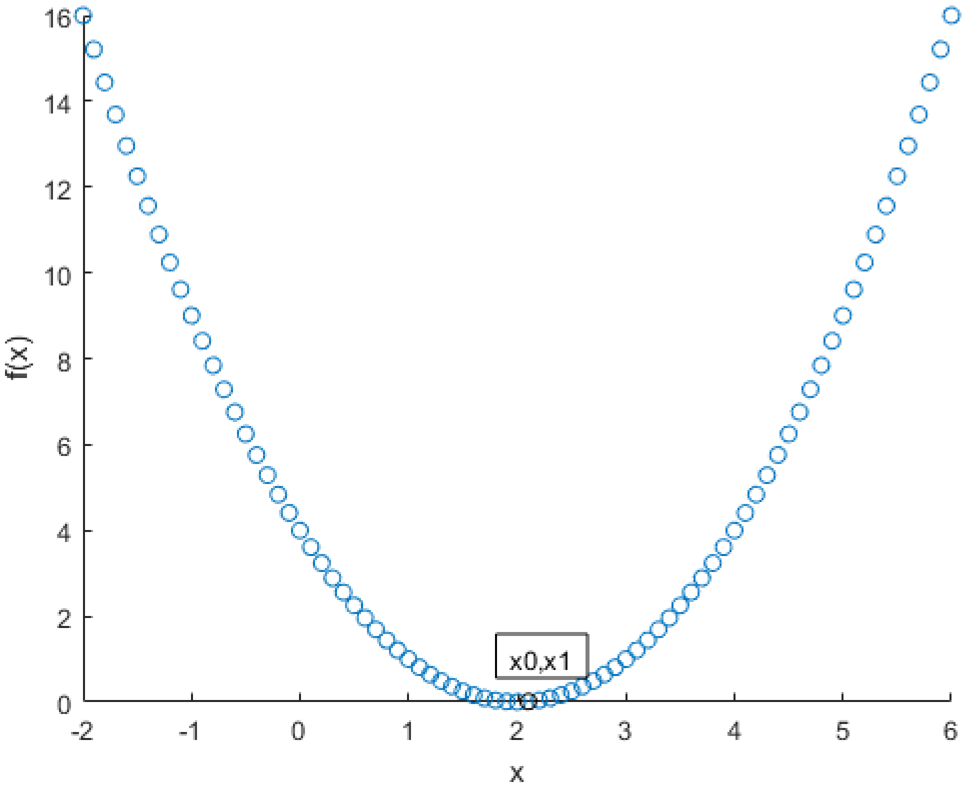
ITERATION 1



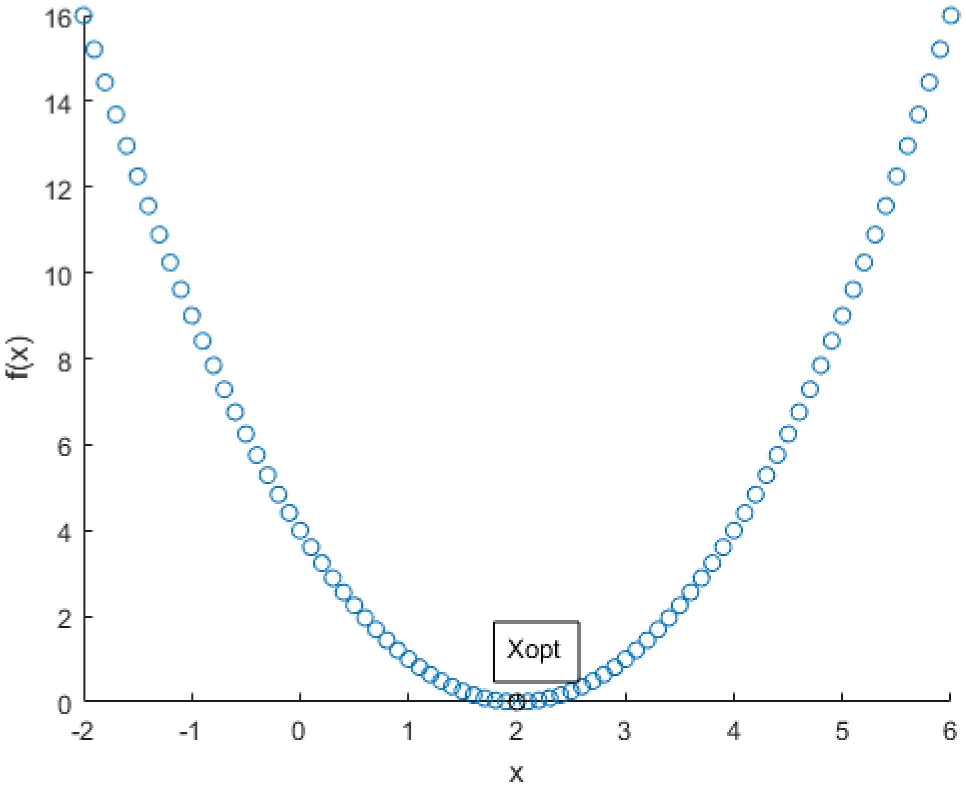
ITERATION 5

Numerical Approach

- Consider $f(x) = x^2 - 4x + 4$; $f'(x) = 2x - 4$
- Set initial point x=5, learning rate a = 0.1



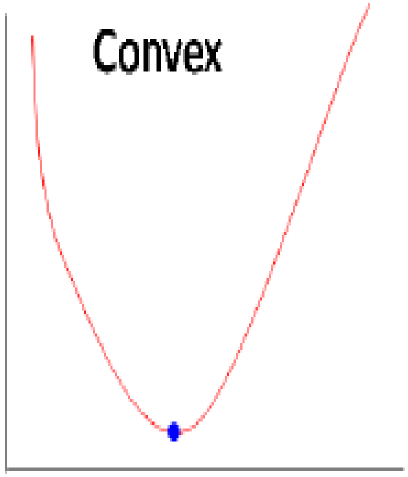
ITERATION 15

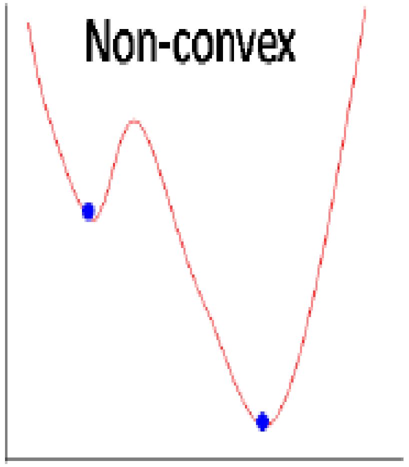


ITERATION 20: CONVERGENCE!

Gradient Descent Issues

- Does it always converge?
  - Depends on the "learning rate"
  - low learning rate: slow convergence
  - high learning rate: may oscillate around the minima!
- Does it always give the optimal solution?
  - Yes, if the function is **Convex** (has *unique minima*)
  - Otherwise, it converges at *any minima*





Logistic Regression Loss Function

Loss function $$L(\boldsymbol{w}) = -\sum_{n=1}^{N}(y_n \boldsymbol{w}^\top \boldsymbol{x}_n - \log(1 + \exp(\boldsymbol{w}^\top \boldsymbol{x}_n)))$$

First task: find the derivative L'(w)!

$$\begin{aligned}
\mathbf{g} = \frac{\partial L(\boldsymbol{w})}{\partial \boldsymbol{w}} &= \frac{\partial}{\partial \boldsymbol{w}}[-\sum_{n=1}^{N}(y_n \boldsymbol{w}^\top \boldsymbol{x}_n - \log(1 + \exp(\boldsymbol{w}^\top \boldsymbol{x}_n)))] \\
&= -\sum_{n=1}^{N}\left(y_n \boldsymbol{x}_n - \frac{\exp(\boldsymbol{w}^\top \boldsymbol{x}_n)}{(1 + \exp(\boldsymbol{w}^\top \boldsymbol{x}_n))}\boldsymbol{x}_n\right) \\
&= -\sum_{n=1}^{N}(y_n - \mu_n)\boldsymbol{x}_n = \mathbf{X}^\top(\boldsymbol{\mu} - \boldsymbol{y})
\end{aligned}$$

Gradient Descent for Logistic Regression

- Initialize $\boldsymbol{w}^{(1)} \in \mathbb{R}^D$ randomly.
- Iterate the following until convergence

$$\underbrace{\boldsymbol{w}^{(t+1)}}_{\text{new value}} = \underbrace{\boldsymbol{w}^{(t)}}_{\text{previous value}} - \eta \underbrace{\sum_{n=1}^{N}(\mu_n^{(t)} - y_n)\boldsymbol{x}_n}_{\text{gradient at previous value}}$$

where η is the learning rate and $\mu^{(t)} = \sigma(\boldsymbol{w}^{(t)^\top}\boldsymbol{x}_n)$ is the predicted label probability for $\boldsymbol{x}_n$ using $\boldsymbol{w} = \boldsymbol{w}^{(t)}$ from the previous iteration

And finally, this is what ChatGPT said (with my help):

Regression, a dance with numbers and lines,
A journey through data, a search for signs.
Inquest of the past, a quest for more,
To find the patterns, to open the door.

From simple to complex, the models we make,
Linear or logistic, for goodness' sake.
A constant pursuit for the coefficient,
That can make predictions more efficient.

To enforce its structure, we can regularize,
The beauty of the ridge, to enjoy and visualize.
Sparse or dense, it can take any form,
If we only can find the right norm.

So let us embrace, the art of regression,
Descend the gradients, to make progression.
With each new data, we learn and grow,
And find meanings, we thought we'd never know