



# Lesson #05

## Logistic Regression

- Classification
- Binary Classification
- Decision Boundary
- Cost Function
- Multiclass Classification
- Regularization (L1, L2)
- Hands on Scikit-Learn



## MEDICAL MODEL



HEALTHY



SICK



## SPAM CLASSIFIER MODEL



NOT SPAM



SPAM

# Classification Problem


3





Test



Grades

Student 1  
Test: 9/10   
Grades: 8/10

Student 2  
Test: 3/10   
Grades: 4/10

Student 3  
Test: 7/10   
Grades: 6/10



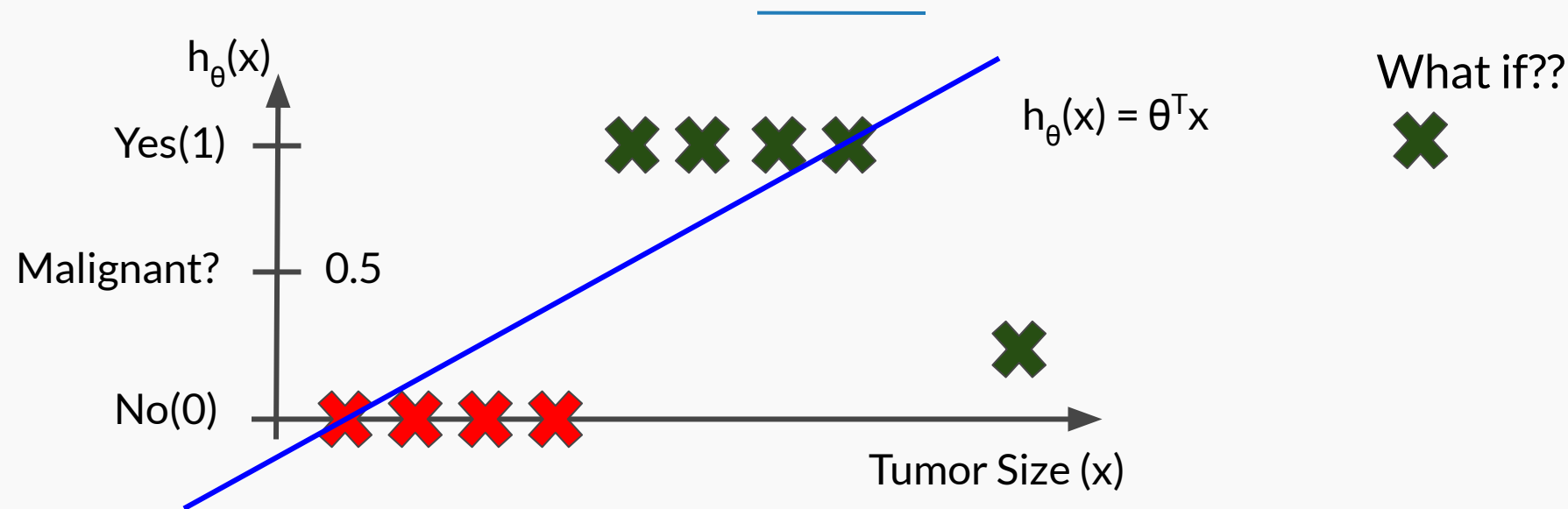
# Binary Classification Problem

---

- Email = {spam, not spam}
- Medical model = {healthy, sick}
- Fraudulent operation = {yes, not}
- Academic acceptance = {success, fail}
- Movie review = {good, bad}

$Y \in \{0,1\}$   0: negative class  
1: positive class

# Binary Classification Problem (observation #1)



Threshold classifier output as  $h_{\theta}(x)$ :

- If  $h_{\theta}(x) \geq 0.5$ , predict  $y = 1$
- If  $h_{\theta}(x) < 0.5$ , predict  $y = 0$

# Binary Classification Problem (observation #2)

---

- Y assume only two values: 0 or 1.
- In linear case,  $h_{\theta}(x) \geq 1$  and  $h_{\theta}(x) \leq 0$  can occur.



Logistic Regression  
 $0 \leq h_{\theta}(x) \leq 1$

# Logistic Regression - Hypothesis Representation

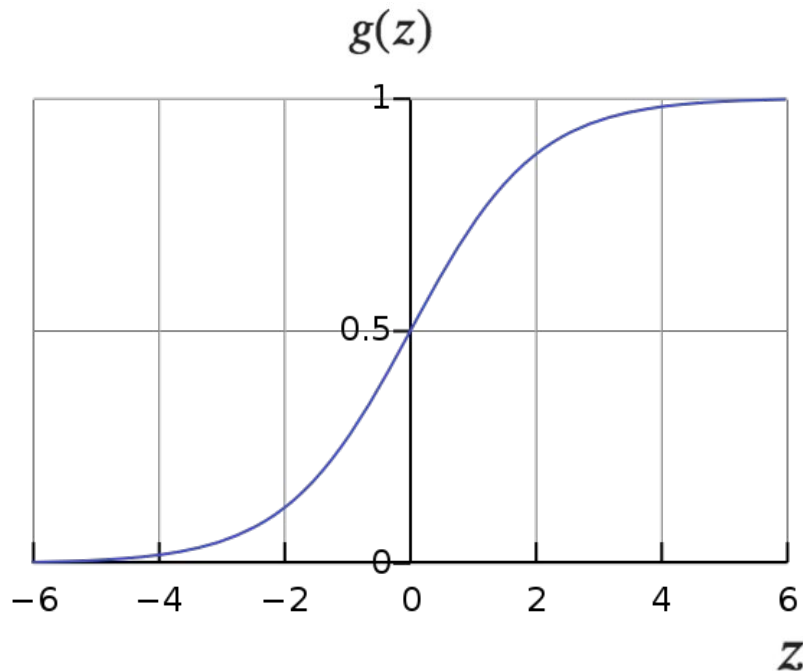
Target  $\rightarrow 0 \leq h_{\theta}(x) \leq 1$

$$h_{\theta}(x) = \theta^T x \quad (\text{doesn't work})$$

$$h_{\theta}(x) = g(z) \quad , \text{where } z = \theta^T x$$

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

Sigmoid function or  
Logistic function



# Logistic Regression - Decision Boundary

Suppose:

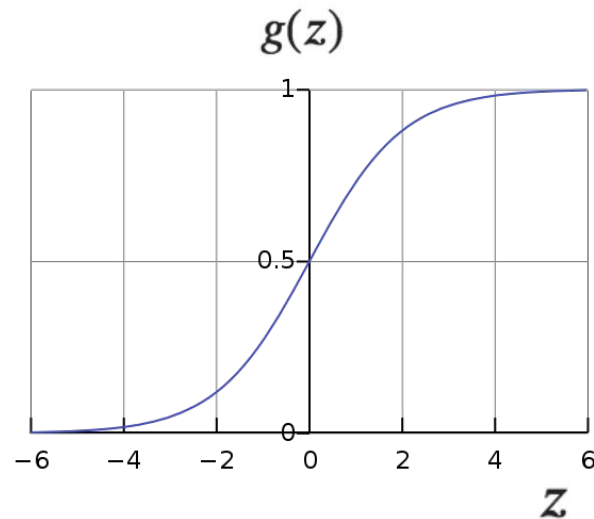
Predict  $y = 1$  if  $h_{\theta}(x) \geq 0.5$

$$g(z) \geq 0.5 \text{ when } z \geq 0$$

Suppose:

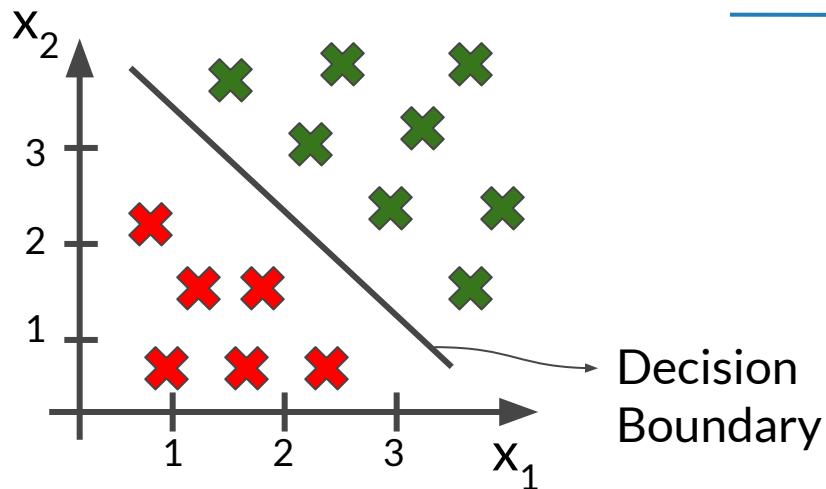
Predict  $y = 0$  if  $h_{\theta}(x) < 0.5$

$$g(z) < 0.5 \text{ when } z < 0$$





# Logistic Regression - Decision Boundary



Suppose:

Predict  $y = 1$  if  $z \geq 0$

$$-3 + x_1 + x_2 \geq 0$$

$$x_1 + x_2 \geq 3$$

$$h_{\theta}(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

$$\theta = \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix}$$

$$z = -3 + x_1 + x_2$$

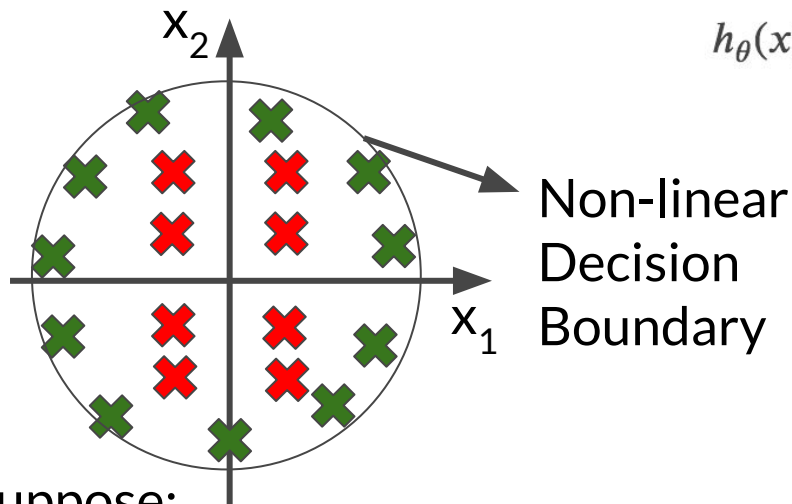
Suppose:

Predict  $y = 0$  if  $z < 0$

$$-3 + x_1 + x_2 < 0$$

$$x_1 + x_2 < 3$$

# Logistic Regression - Decision Boundary



Suppose:

Predict  $y = 1$  if  $z \geq 0$

$$-1 + x_1^2 + x_2^2 \geq 0$$

$$x_1^2 + x_2^2 \geq 1$$

$$h_{\theta}(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2)$$

$$\theta = \begin{bmatrix} -1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

$$z = -1 + x_1^2 + x_2^2$$

Suppose:

Predict  $y = 0$  if  $z < 0$

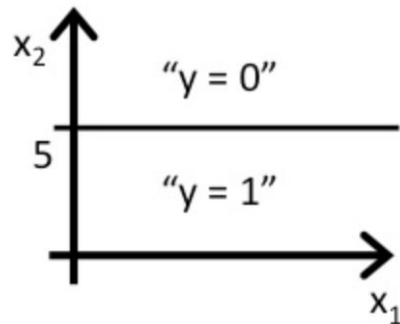
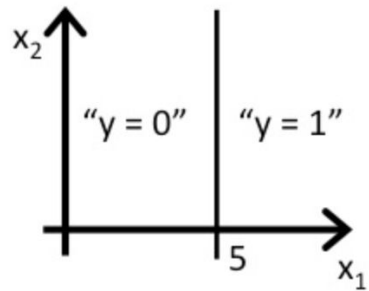
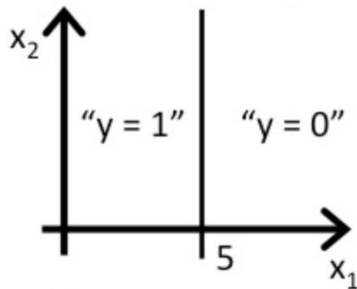
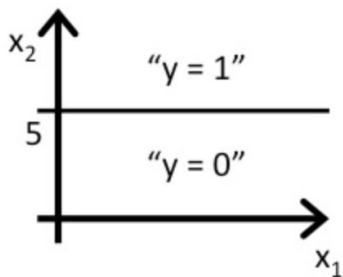
$$-1 + x_1^2 + x_2^2 < 0$$

$$x_1^2 + x_2^2 < 1$$

# Logistic Regression - Decision Boundary

Consider logistic regression with two features  $x_1$  and  $x_2$ . Suppose  $\Theta_0 = 5$ ,  $\Theta_1 = -1$  and  $\Theta_2 = 0$ , so that  $h_{\Theta}(x) = g(5 - x_1)$ .

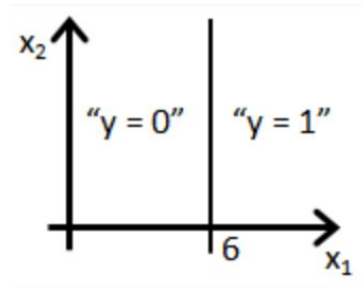
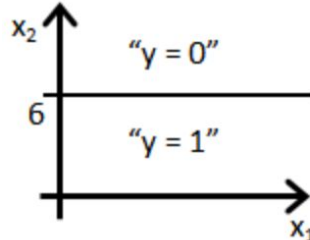
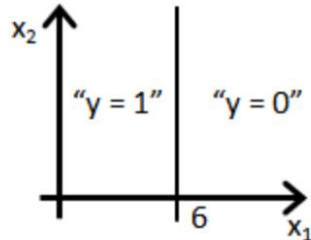
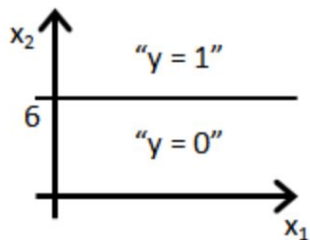
Which of these shows the decision boundary of  $h_{\Theta}(x)$ ?



# Logistic Regression - Decision Boundary

Consider logistic regression with two features  $x_1$  and  $x_2$ . Suppose  $\Theta_0 = 6$ ,  $\Theta_1 = 0$  and  $\Theta_2 = -1$ , so that  $h_{\Theta}(x) = g(\Theta_0 + \Theta_1 x_1 + \Theta_2 x_2)$ .

Which of these shows the decision boundary of  $h_{\Theta}(x)$ ?



# Logistic Regression - Decision Boundary

---

Suppose that you have trained a logistic regression classifier, and it outputs on a new example  $x$  a prediction  $h_{\theta}(x) = 0.4$ . This means (check all that apply):

Our estimate for  $P(y = 0|x; \theta)$  is 0.4.

Our estimate for  $P(y = 0|x; \theta)$  is 0.6.

Our estimate for  $P(y = 1|x; \theta)$  is 0.4.

Our estimate for  $P(y = 1|x; \theta)$  is 0.6.

# RECAP

$f(x)$

cost function



Training Set:  $\{(\mathbf{x}^1, y^1), (\mathbf{x}^2, y^2), \dots, (\mathbf{x}^m, y^m)\}$   
m examples

$$x \in \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix}, \text{ <n+1 elements> } , x_0 = 1, y \in \{0, 1\}$$

$$h_{\theta}(x) = \frac{1}{1+e^{-\theta^T x}}$$

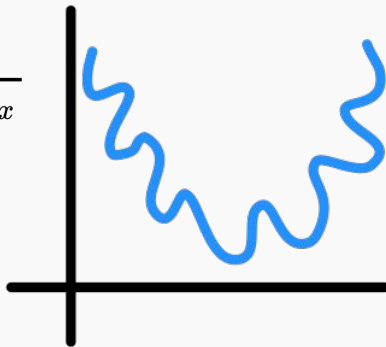
How to fit the parameter  $\theta$ ?

# Cost Function

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

non-convex

$$h_{\theta}(x) = \frac{1}{1+e^{-\theta^T x}}$$

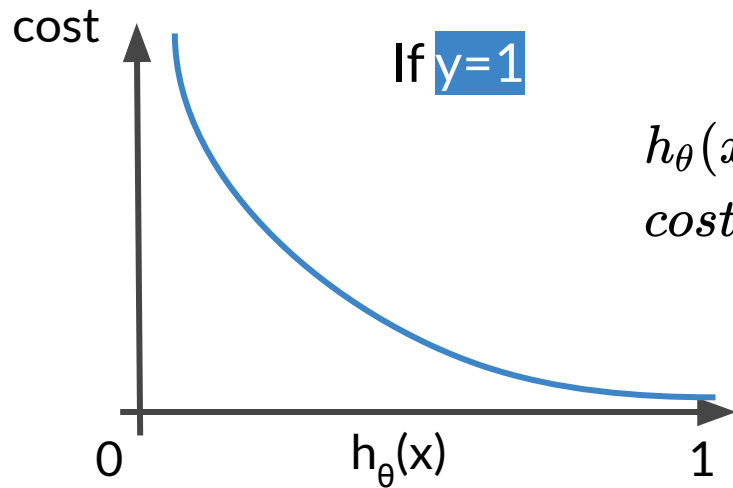


convex

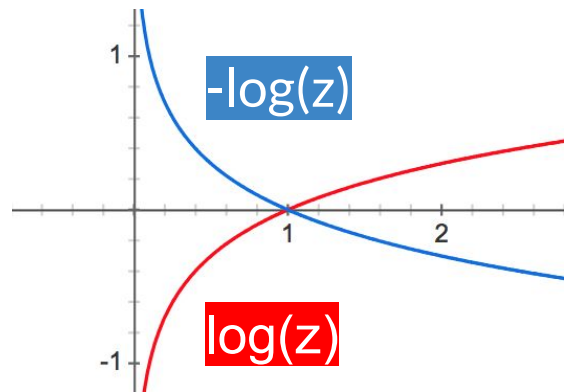


# Logistic Regression Cost Function

$$\text{cost}(h_{\theta}(x), y) = \begin{cases} -\log(h_{\theta}(x)) & \text{if } y=1 \\ -\log(1 - h_{\theta}(x)) & \text{if } y=0 \end{cases}$$

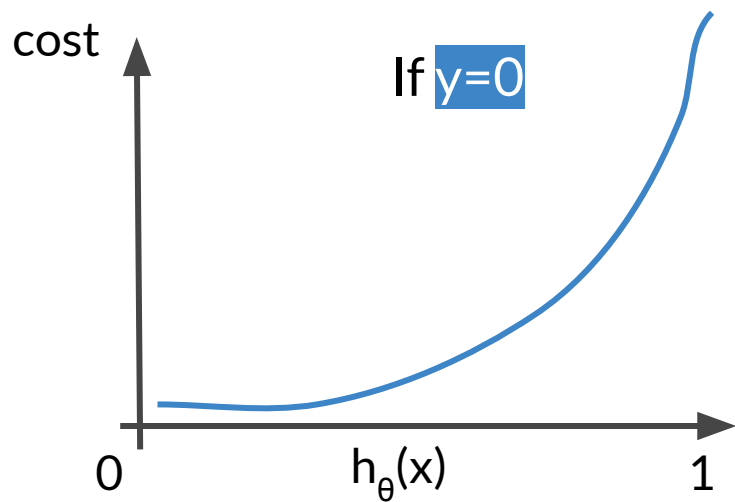


$$\begin{array}{ll} h_{\theta}(x) \rightarrow 1 & h_{\theta}(x) \rightarrow 0 \\ \text{cost} \rightarrow 0 & \text{cost} \rightarrow \infty \end{array}$$



# Logistic Regression Cost Function

$$\text{cost}(h_{\theta}(x), y) = \begin{cases} -\log(h_{\theta}(x)) & \text{if } y=1 \\ -\log(1 - h_{\theta}(x)) & \text{if } y=0 \end{cases}$$

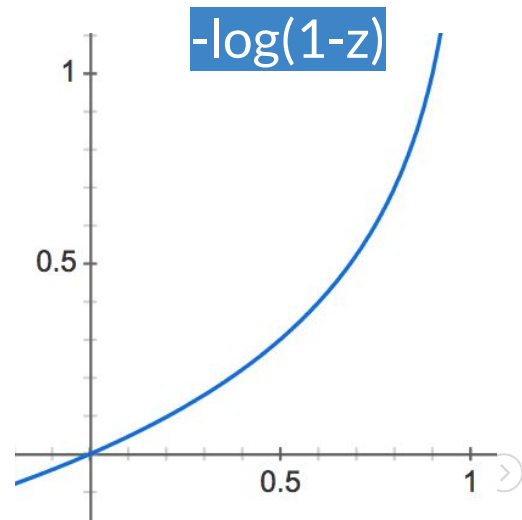


$$h_{\theta}(x) \rightarrow 1$$

$$\text{cost} \rightarrow \infty$$

$$h_{\theta}(x) \rightarrow 0$$

$$\text{cost} \rightarrow 0$$



# Simplified Cost Function & Gradient Descent

# Logistic Regression Cost Function

---

$$\text{cost}(h_{\theta}(x), y) = \begin{cases} -\log(h_{\theta}(x)) & \text{if } y=1 \\ -\log(1 - h_{\theta}(x)) & \text{if } y=0 \end{cases}$$

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

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



# Cost Function - Vectorized Implementation

---

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

$$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_k \end{bmatrix}$$

$$h = g(X\theta)$$

[m; k+1] x [k+1;1] = [m;1]

$$J(\theta) = \frac{1}{m} \cdot \left( -y^T \log(h) - (1 - y)^T \log(1 - h) \right)$$

[1;m] x [m;1] = scalar



# General Form of Gradient Descent

---

*Repeat* {

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

}

*Repeat* {

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

}



Vectorized Implementation

$$\theta := \theta - \frac{\alpha}{m} X^T (g(X\theta) - \vec{y})$$

# Multiclass Classification:

## One vs All

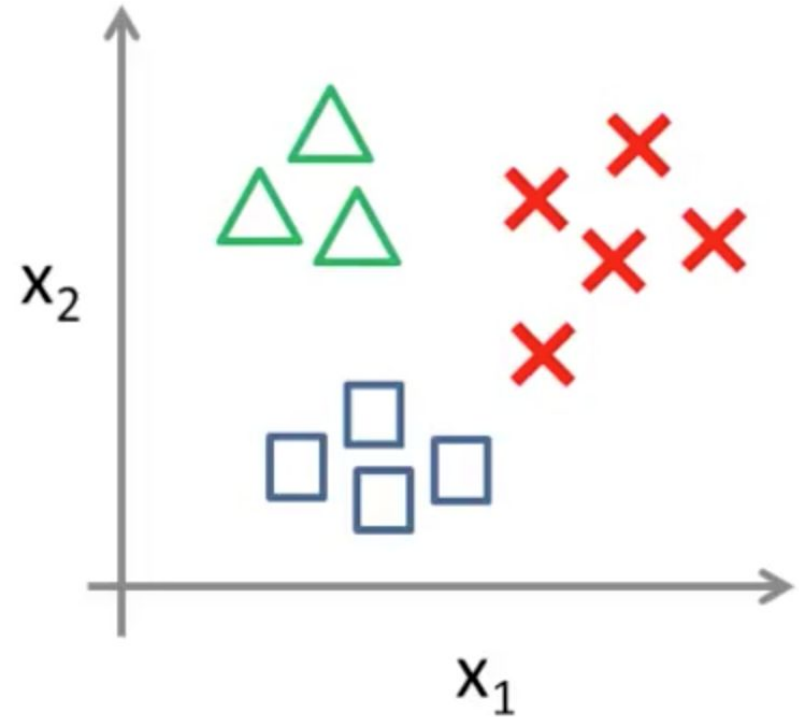
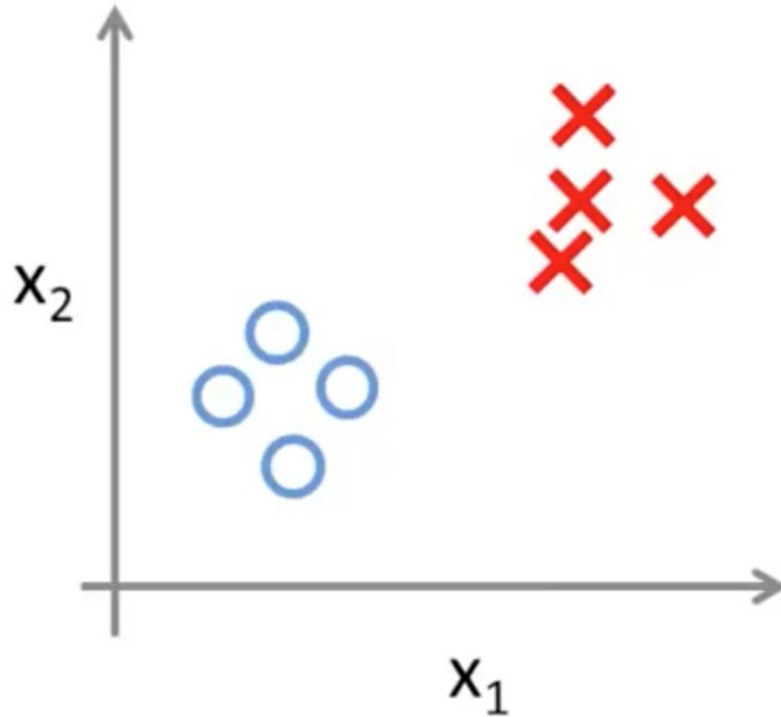
# Multiclass Classification

---

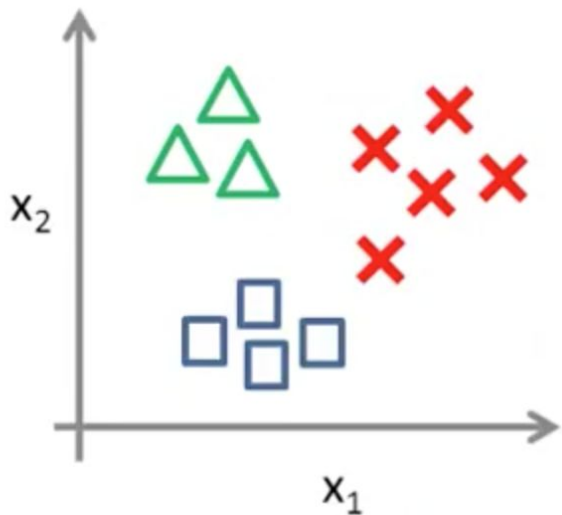
- Email foldering/tagging: work, ad, family, friends, hobby
- Medical diagrams: not ill, cold, flu
- Weather: sunny, cloudy, rain, snow


# Binary vs Multiclass Classification

---



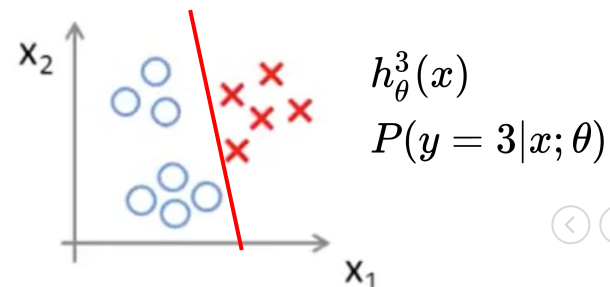
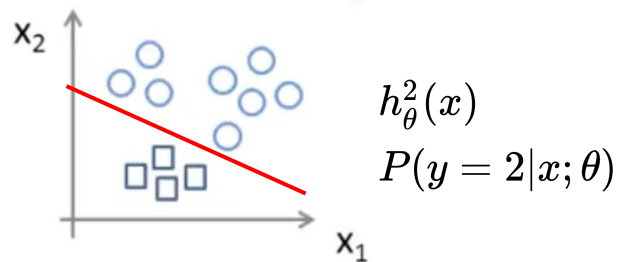
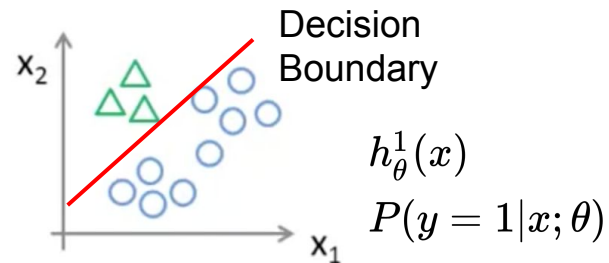
# Multiclass Classification (One vs All)



Class 1: 

Class 2: 

Class 3: 





# Multiclass Classification (One vs All)

---

Train a logistic regression classifier  $h_{\theta}^{(i)}(x)$  for each class  $i$ :

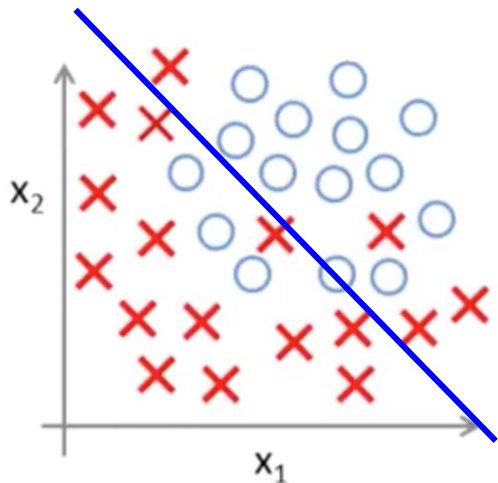
$$h_{\theta}^i(x) = P(y = i|x; \theta)$$

On a new input  $x$ , to make a prediction, pick the class  $i$  that maximizes:

$$\max_i h_{\theta}^{(i)}(x)$$

# overfitting problem

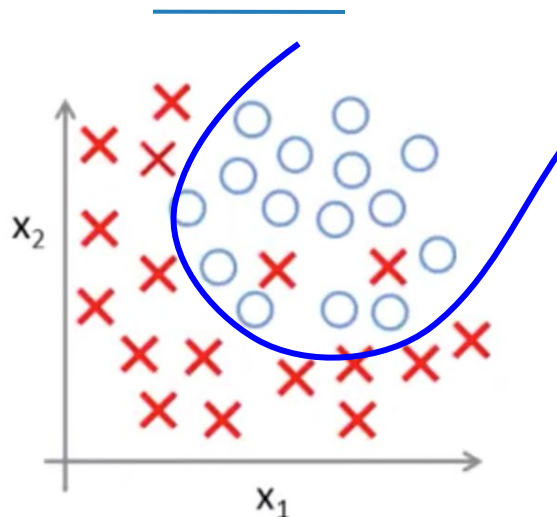
# Logistic Regression



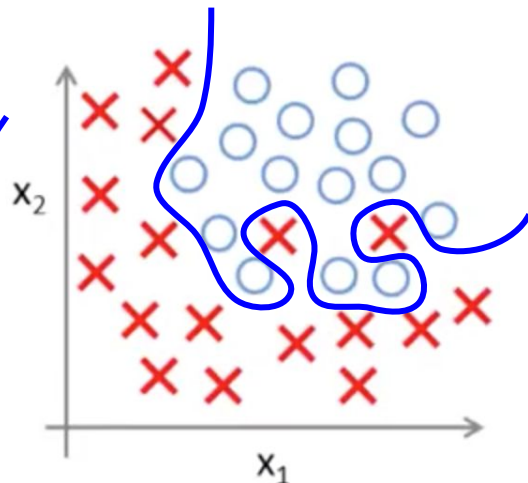
$$h_{\theta}(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2)$$

(  $g$  = sigmoid function)

Underfit



$$g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2 + \theta_5 x_1 x_2)$$



$$g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^2 x_2 + \theta_4 x_1^2 x_2^2 + \theta_5 x_1^2 x_2^3 + \theta_6 x_1^3 x_2 + \dots)$$

Overfit

# Addressing Overfitting

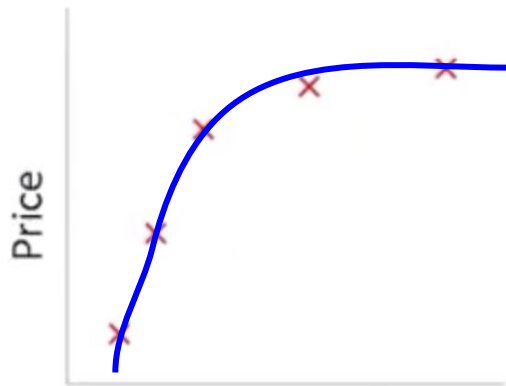
---

1. Reduce number of features
  - a. Manually select which feature to keep
2. Regularization
  - a. Keep all the features, but reduce magnitude/values of parameters  $\Theta_j$
  - b. Works well when we have a lot of features, each of which contributes a bit to predicting  $y$ .

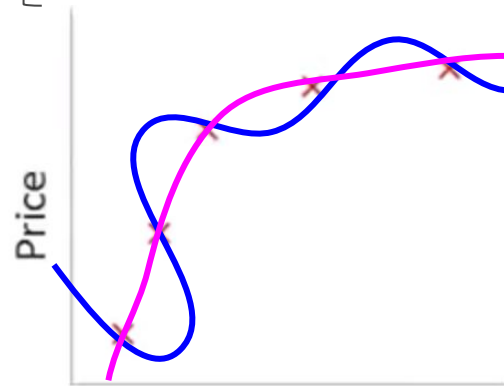
# Intuition - Regularized Linear Regression

$$\min_{\theta} \frac{1}{2m} \left[ \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^n \theta_j^2 \right]$$

Regularization  
Parameter



$$\theta_0 + \theta_1 x + \theta_2 x^2$$



$$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$$

Suppose we penalize and make  $\Theta_3$  and  $\Theta_4$  very small

$$\min_{\theta} \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2 + 1000 \cdot \theta_3^2 + 1000 \cdot \theta_4^2$$


# Intuition - Gradient Descent (L2)

Repeat {

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)}) x_0^{(i)}$$

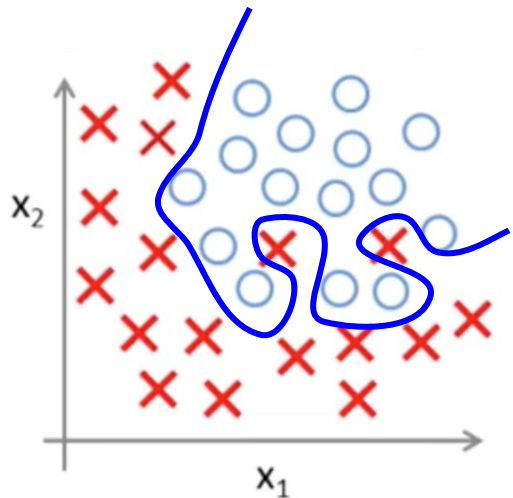
$$\theta_j := \theta_j - \alpha \left[ \left( \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)}) x_j^{(i)} \right) + \frac{\lambda}{m} \theta_j \right] \quad j \in \{1, 2 \dots n\}$$

}


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



# Intuition - Regularized Logistic Regression

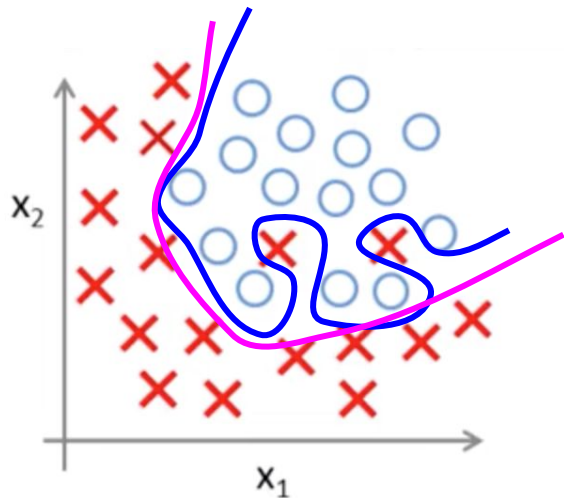


$$h_{\theta}(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^2 x_2 + \theta_4 x_1^2 x_2^2 + \theta_5 x_1^2 x_2^3 + \dots)$$

Cost function:

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

# Intuition - Regularized Logistic Regression



$$h_{\theta}(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_1^2 + \theta_3 x_1^2 x_2 + \theta_4 x_1^2 x_2^2 + \theta_5 x_1^2 x_2^3 + \dots)$$

Cost function:

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^m [y^{(i)} \log(h_{\theta}(x^{(i)})) + (1 - y^{(i)}) \log(1 - h_{\theta}(x^{(i)}))] + \frac{\lambda}{2m} \sum_{j=1}^n \theta_j^2$$


# Intuition - Gradient Descent (L2)

Repeat {

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)}) x_0^{(i)}$$

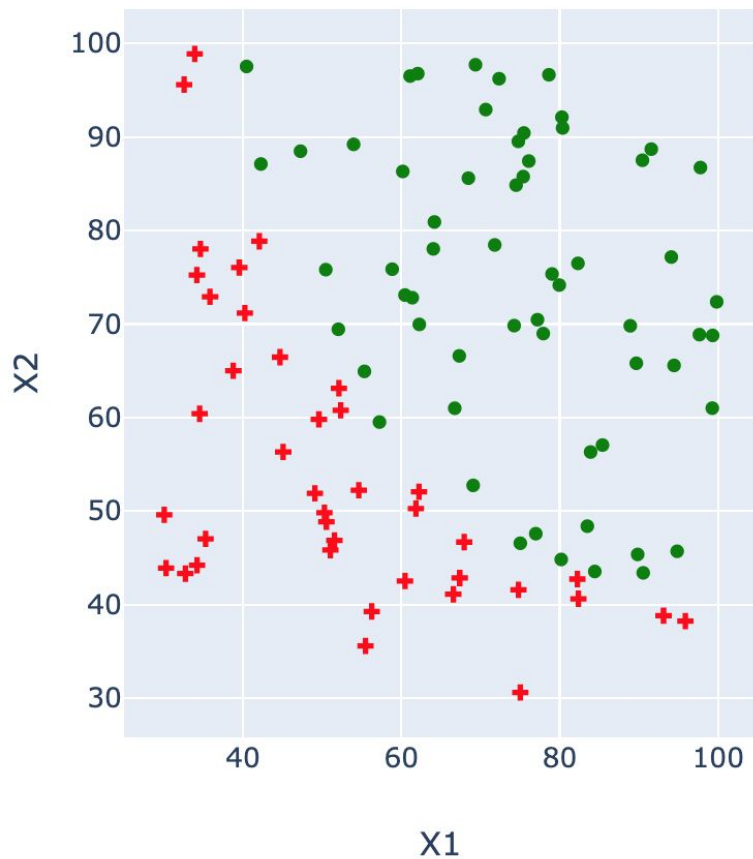
$$\theta_j := \theta_j - \alpha \left[ \left( \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)}) x_j^{(i)} \right) + \frac{\lambda}{m} \theta_j \right] \quad j \in \{1, 2 \dots n\}$$

}

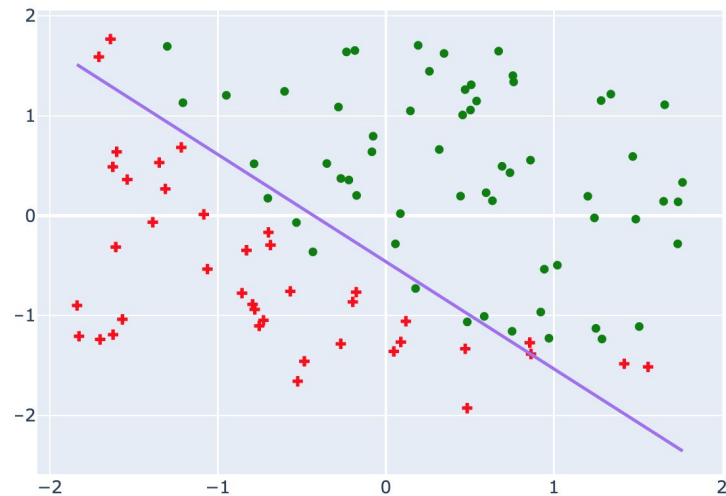

$$\theta_j := \theta_j (1 - \alpha \frac{\lambda}{m}) - \alpha \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

# Admitted vs not Admitted

36

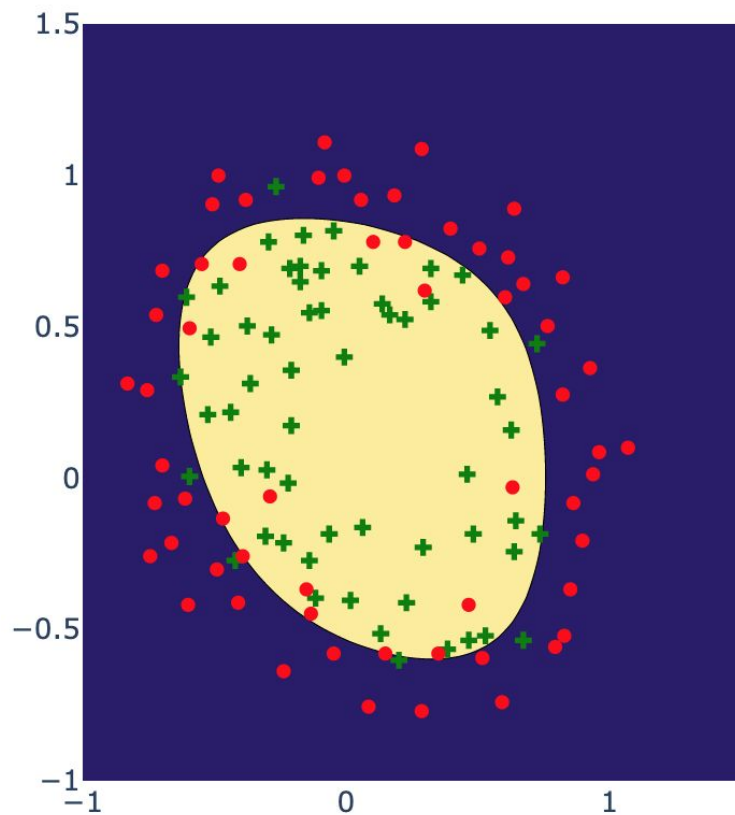
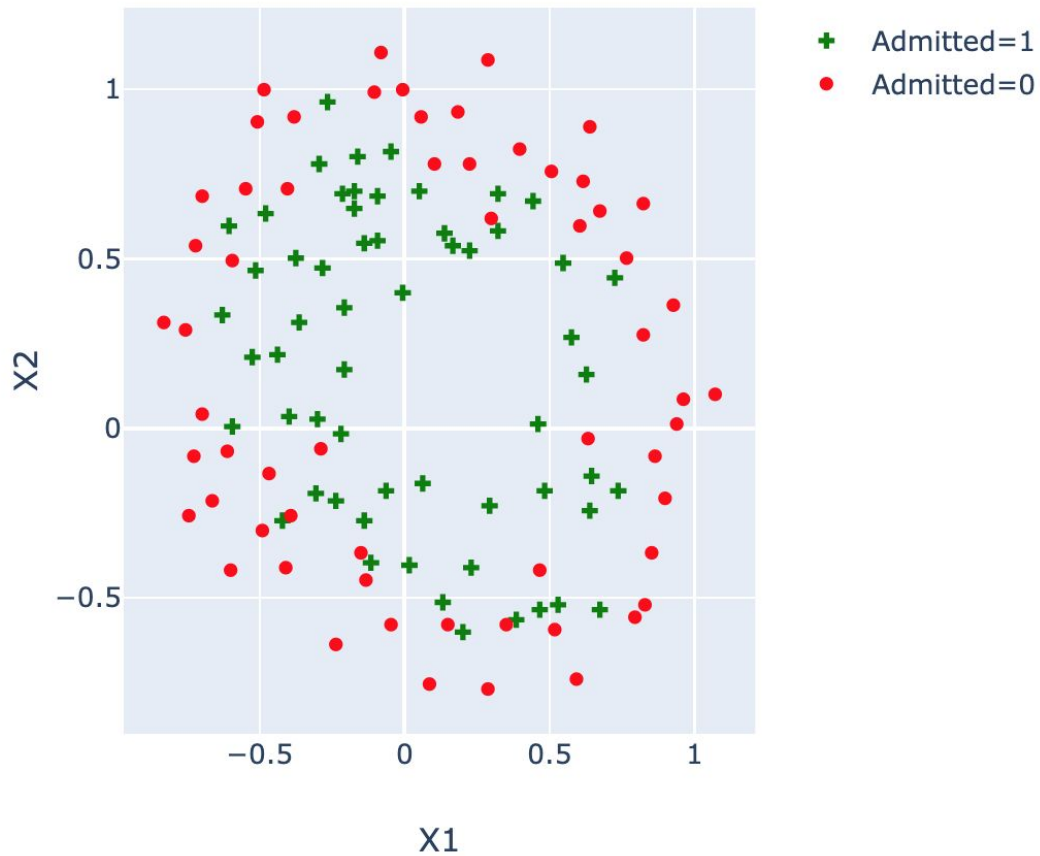


+ Admitted=0  
● Admitted=1



## Admitted vs not Admitted

37



## Lesson#05 - Some experimentation.ipynb

- Visualize different decision boundaries
- Analyze the results (thetas) considering the follow regularization techniques: L1, L2
- Changes values of lambda, iterations, so on
- Check the equations

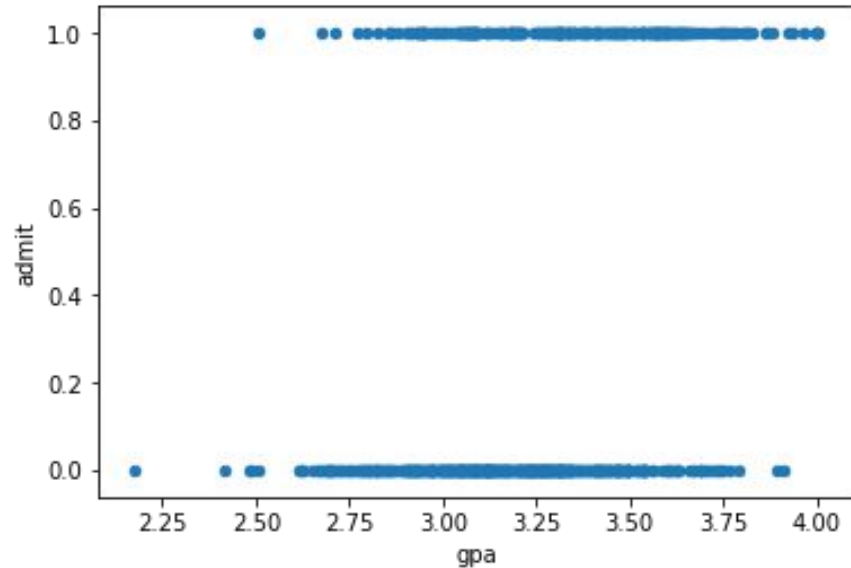




# Binary Classification

---

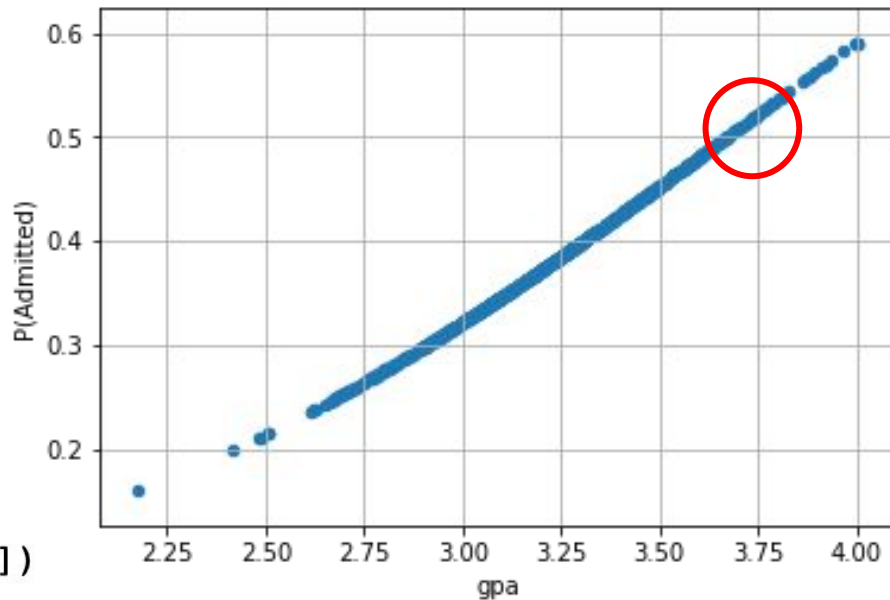
admit	gpa	gre
0	3.177277	594.102992
0	3.412655	631.528607
0	2.728097	553.714399
0	3.093559	551.089985
0	3.141923	537.184894



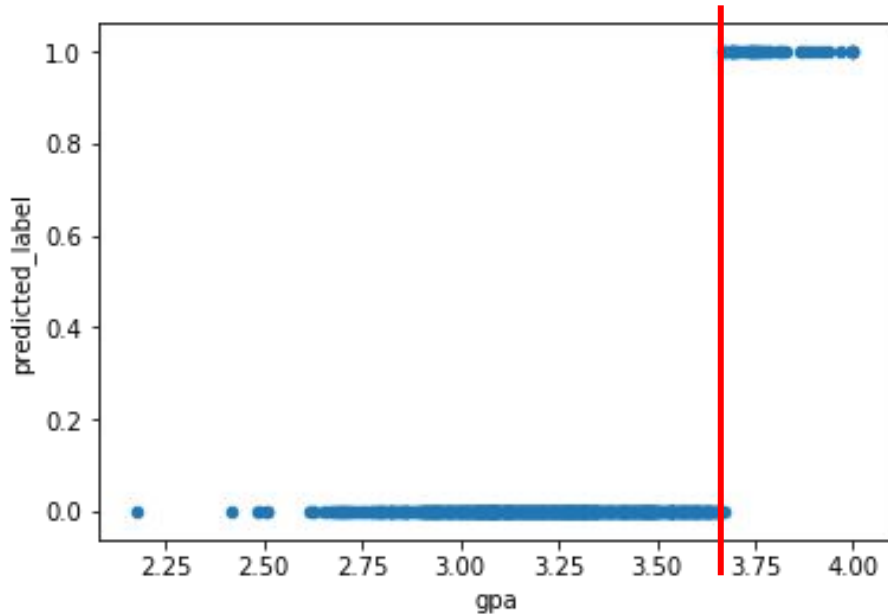


# Logistic Regression Model (fit, predict prob.)

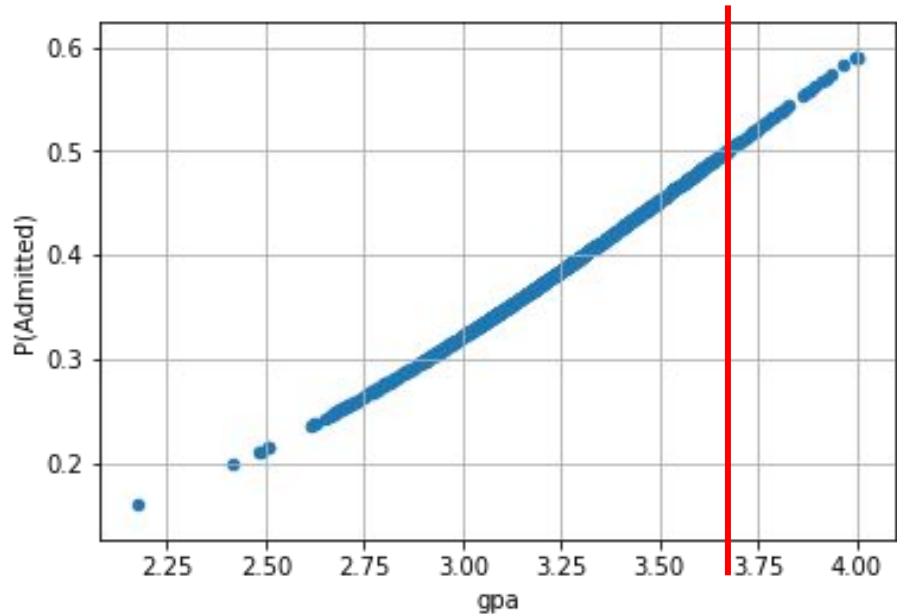
```
# create a model
model = LogisticRegression()
# training
model.fit(data[["gpa"]], data["admit"])
# predict
prob = model.predict_proba(data[["gpa"]])
# store in a dataframe
data["P(Admitted)"] = prob[:,1]
```



# Logistic Regression Model (fit, predict class)



```
model.predict(data[["gpa"]])
```



```
model.predict_proba(data[["gpa"]])[:,1]
```

# Evaluating Binary Classifiers

Prediction	Observation	
	Admitted (1)	Rejected (0)
Admitted (1)	True Positive (TP)	False Positive (FP)
Rejected (0)	False Negative (FN)	True Negative (TN)

```
from sklearn.metrics import confusion_matrix
```

```
tn, fp, fn, tp = confusion_matrix(data.admit,  
                                  data.predicted_label).ravel()
```

# Evaluating Binary Classifiers

Prediction	Observation	
	Admitted (1)	Rejected (0)
Admitted (1)	True Positive (TP)	False Positive (FP)
Rejected (0)	False Negative (FN)	True Negative (TN)

$$Accuracy = \frac{\text{\#correct predictions}}{\text{\#observations}}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP}$$

$$TPR = \frac{\text{\#true positives}}{\text{\#true positives} + \text{\#false negatives}}$$

Recall

$$TNR = \frac{\text{\#true negatives}}{\text{\#true negatives} + \text{\#false positives}}$$

# Evaluating Binary Classifiers

Prediction	Observation	
	Admitted (1)	Rejected (0)
Admitted (1)	True Positive (TP)	False Positive (FP)
Rejected (0)	False Negative (FN)	True Negative (TN)

$$precision = \frac{TP}{(TP + FP)}$$

$$precision = \frac{TN}{(TN + FN)}$$

$$F_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

```
from sklearn.metrics import classification_report

print(classification_report(data[["admit"]],
                             data[["predicted_label"]],
                             target_names=['0', '1']))
```

	precision	recall	f1-score	support
0	0.64	0.96	0.77	400
1	0.67	0.13	0.21	244
accuracy			0.65	644
macro avg	0.66	0.54	0.49	644
weighted avg	0.66	0.65	0.56	644

# Multiclass Classification

---

	mpg	cylinders	displacement	horsepower	weight	acceleration	year	origin
0	18.0	8	307.0	130.0	3504.0	12.0	70	1
1	15.0	8	350.0	165.0	3693.0	11.5	70	1
2	18.0	8	318.0	150.0	3436.0	11.0	70	1
3	16.0	8	304.0	150.0	3433.0	12.0	70	1
4	17.0	8	302.0	140.0	3449.0	10.5	70	1

**origin** -- Integer and Categorical. 1: North America, 2: Europe, 3: Asia.

# Dummy Variables

cyl_3	cyl_4	cyl_5	cyl_6	cyl_8
0	0	0	0	1
0	0	0	0	1
0	0	0	0	1
0	0	0	0	1
0	0	0	0	1

```
dummy_cylinders = pd.get_dummies(
    cars["cylinders"], prefix="cyl")
cars = pd.concat([cars, dummy_cylinders],
                  axis=1)
cars.head()
```

[illegible]



# Training a Multiclass Logistic Regression Model

```
from sklearn.linear_model import LogisticRegression

unique_origins = cars["origin"].unique()
unique_origins.sort()

models = {}
features = [c for c in train.columns
             if c.startswith("cyl") or c.startswith("year")]

for origin in unique_origins:
    model = LogisticRegression()

    X_train = train[features]
    y_train = train["origin"] == origin

    model.fit(X_train, y_train)
    models[origin] = model
```

# Testing (One vs All)

---

	1	2	3
0	0.613723	0.131164	0.262305
1	0.536781	0.226177	0.236130
2	0.613723	0.131164	0.262305
3	0.678392	0.174871	0.154612
4	0.616443	0.226177	0.162931



Terminal 2000

```
INITIATING... WAIT. HOLD A BIT MORE. ALREADY
#####
PENDING SUBROUTINE NO. 01-
WELCOME TO FUNCTION: READER
GETTING FROM PARANTHER: SUPER CRONES
UNKNOWING
COURTESY IS NOT TO GETTING: UNKNOWING CRONES
C-CHAVE, THE WAY YOU HAVE NOW -LAND TO GET. REVEAL
WRITTEN UNKNOWINGLY TO SAY: GEMILE
C-CHAVE, IN REVEAL: REVEAL THE WAY YOU HAVE NOW
HEAT SHOOTING
REVEAL UNKNOWINGLY REVEAL: IS IT PARTLY THE
T...
FIGHT INITIATE FLY GEMILE JAMES WICKS FIGHTING
FIGHT: FLY
T...
DIS-CHAVE... REVEAL WAIT
#####
SUCCESSFULLY INITIATED
C-CHAVE
NAME: JAMES WICKS: FLY WICKS: FLY WICKS
NAME: JAMES WICKS: FLY WICKS: FLY WICKS
NAME: JAMES WICKS: FLY WICKS: FLY WICKS
```