LECTURE 21

# Logistic Regression

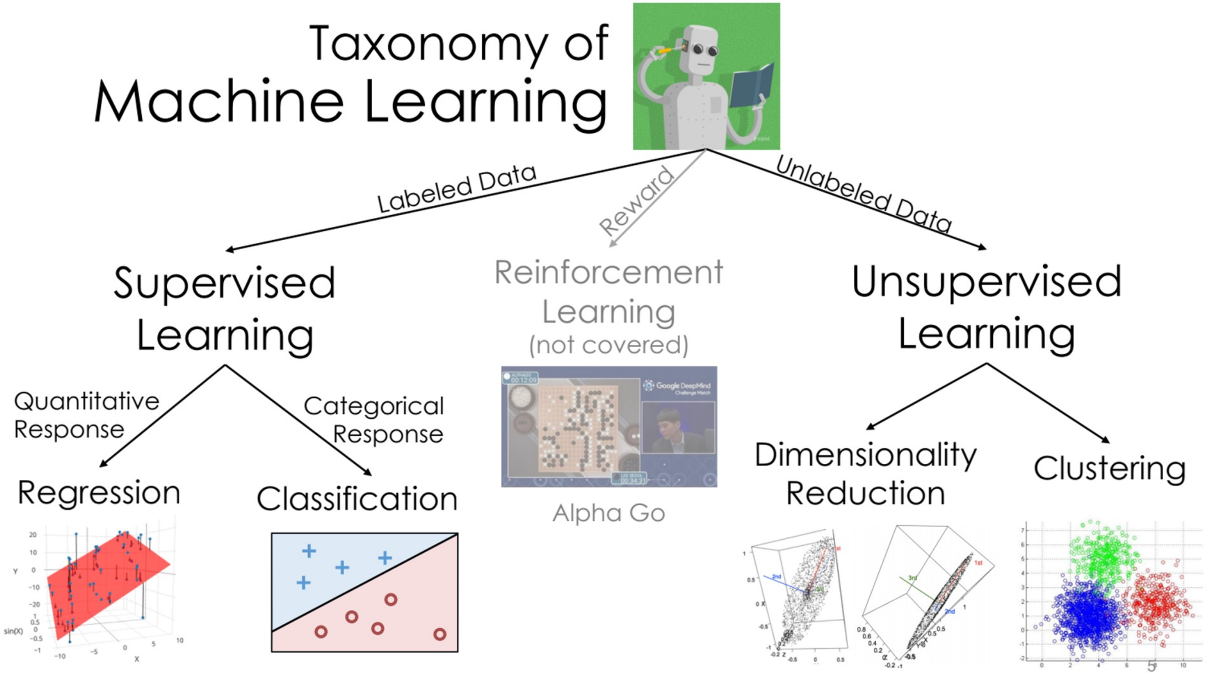
Moving from regression to classification.

Take home final: 24-hour coding-challenge (to be distributed on 4/22)
Extra credit: 2 pt for essay, 1 pt for course evaluation

# Machine learning taxonomy

Regression and Classification are both forms of **supervised learning**.

**Logistic regression**, the topic of this lecture, is mostly used for **classification**, even though it has “regression” in the name.



# Regression vs Classification

**Regression vs Classification**

Logistic regression model derivation

- Linear vs Logistic regression
- logistic function (sigmoid)
- Parameter interpretation

Loss function

- Pitfalls of Squared Loss
- Cross Entropy

Maximum likelihood estimation

## Linear Regression

In a **linear regression** model, our goal is to predict a **quantitative** variable (i.e., some real number) from a set of features.

- Our output, or **response**, y, could be any real number.
- We determined optimal model parameters by minimizing some average loss, and (sometimes) an added regularization penalty.

$$\hat{y} = f_\theta(x) = x^T\theta$$

Remember, .

$$x^T\theta = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_p x_p$$

# Classification

When performing classification, we are instead interested in predicting some **categorical** variable.

spam or ham

win or lose

disease or no disease

# Classification

- **Binary** classification: two classes.
    - Examples: spam / not spam.
    - Our **responses** are either 0 or 1.
    - Our focus today.
- **Multiclass** classification: many classes.
    - Examples: Image labeling (cat, dog, car), next word in a sentence, etc.

# Logistic Regression model derivation

## Example dataset

In this lecture, we will primarily use data from the 2017-18 NBA season.

**Goal:** Predict whether or not a team will win, given their FG_PCT_DIFF.

- This is the difference in field goal percentage between the two teams.
- Positive FG_PCT_DIFF: team made more shots than the opposing team.

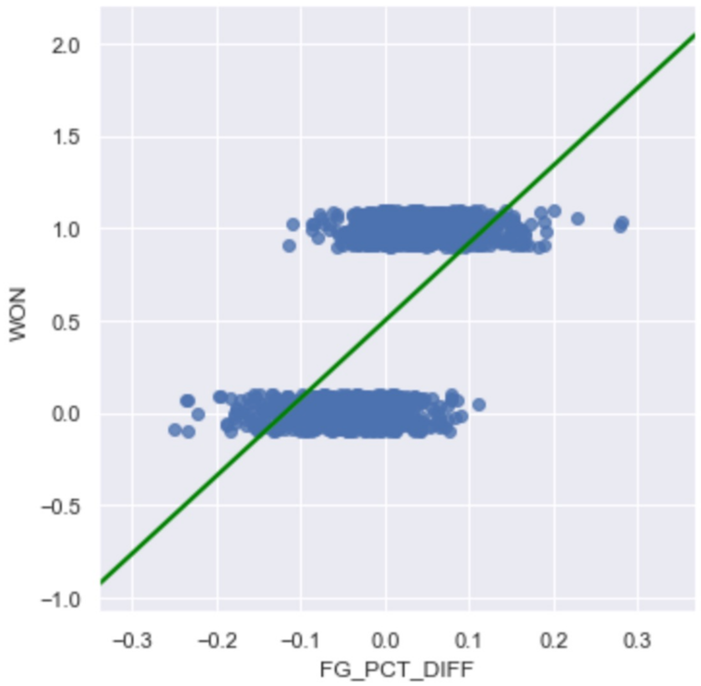
| TEAM_NAME | MATCHUP | WON | FG_PCT_DIFF |
|---|---|---|---|
| Boston Celtics | BOS @ CLE | 0 | -0.049 |
| Golden State Warriors | GSW vs. HOU | 0 | 0.053 |
| Charlotte Hornets | CHA @ DET | 0 | -0.030 |
| Indiana Pacers | IND vs. BKN | 1 | 0.041 |
| Orlando Magic | ORL vs. MIA | 1 | 0.042 |

1s represent wins, 0s represent losses.

# Why not use Ordinary Least Squares?

We already have a model that can predict any quantitative response. Why not use it here?

- The output can be outside of the range [0, 1]. What does a predicted WON value of -2 mean?
- Very sensitive to outliers/ imbalanced data.
- Many other statistical reasons.

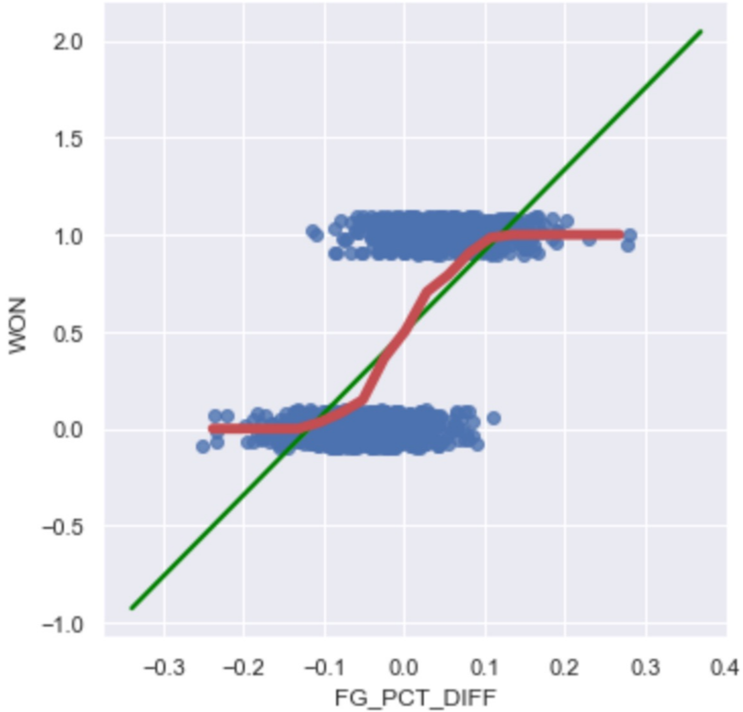


# Graph of averages

If we **bin the x-axis**, and take the **average** y-value for each bin, and tried to model that.

Doing so here yields a curve that resembles an s.

- Since our true y is either 0 or 1, this curve models the **probability that WON = 1**, given FG_PCT_DIFF.
    - WON = 1 means "belong to class 1".
- **Our goal is to model this red curve as best as possible**.



# Log-odds of probability is roughly linear

In the demo, we noticed that the **log-odds of the probability of belonging to class 1 was linear**. **This is the assumption that logistic regression is based on**.

$$\text{odds}(p) = \frac{p}{1-p} \qquad \text{log-odds}(p) = \log\left(\frac{p}{1-p}\right)$$

For now, let's let $t$ denote our linear function (since log-odds is linear). Solving for $p$:

$$t = \log\left(\frac{p}{1-p}\right)$$

$$e^t = \frac{p}{1-p}$$

$$e^t - pe^t = p$$

$$p = \frac{e^t}{1+e^t} = \frac{1}{1+e^{-t}}$$

With logistic regression, we are always referring to log base e ("ln").

## Log-odds of probability is roughly linear

In the demo, we noticed that the **log-odds of the probability of belonging to class 1 was linear**. **This is the assumption that logistic regression is based on**.

$$\text{odds}(p) = \frac{p}{1-p} \qquad \text{log-odds}(p) = \log\left(\frac{p}{1-p}\right)$$

For now, let's let $t$ denote our linear function (since log-odds is linear). Solving for $p$:

$$t = \log\left(\frac{p}{1-p}\right)$$

$$e^t = \frac{p}{1-p}$$

$$e^t - pe^t = p$$

$$p = \frac{e^t}{1+e^t} = \frac{1}{1+e^{-t}}$$

This is called the **logistic function**, $\sigma$(t).

# Arriving at the logistic regression model

We know how to model linear functions quite well.

- We can substitute $t = x^T\theta$ , since $t$ was just a placeholder.

$$p = \frac{1}{1 + e^{-t}} = \sigma(t)$$

$p$ represents the probability of belonging to class 1.

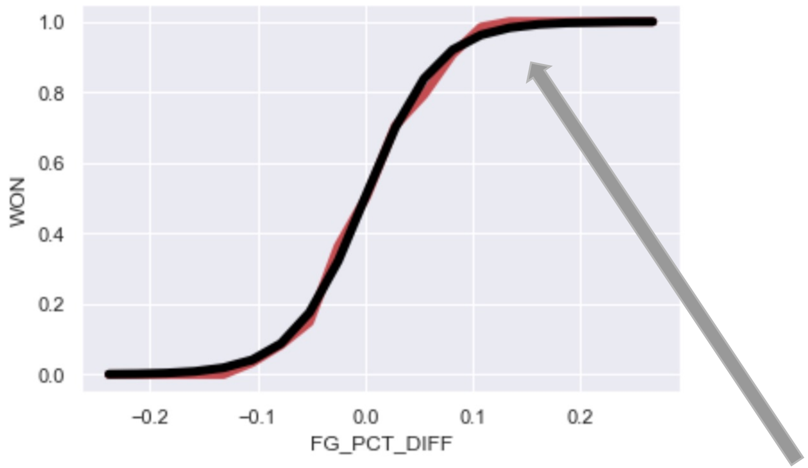
- We are modeling $P(Y = 1|x)$ .

Putting this all together:

$$P(Y = 1|x) = \frac{1}{1 + e^{-x^T\theta}} = \sigma(x^T\theta)$$

Looks just like the linear regression model, with a $\boldsymbol{\sigma}(\,)$ wrapped around it.
We call logistic regression a **generalized linear model**, since it is a non-linear transformation of a linear model.

# Arriving at the logistic regression model




* no transposes here, since we only looked at one feature (without an intercept term!)

**In red:**
Empirical graph of averages

**In black:**
$$\hat{y} = \sigma(30 \cdot \text{FG PCT DIFF})$$

# Logistic Regression

1. Choose a model

2. Choose a loss function

3. Fit the model

4. Evaluate model performance

**Logistic Regression** $\hat{y} = f_\theta(x) = P(Y = 1|x) = \sigma(x^T\theta)$

# Linear vs Logistic Regression

# Linear vs. logistic regression

In a **linear regression** model, we predict a **quantitative** variable (i.e., some real number) as a linear function of features.

- Our output, or **response**, y, could be any real number.

$$\hat{y} = f_\theta(x) = x^T\theta$$

In a **logistic regression** model, our goal is to predict a binary **categorical** variable (class 0 or class 1) as a linear function of features, passed through the logistic function.

- Our **response** is the probability that our observation belongs to class 1.
- Haven't yet done classification!

$$\hat{y} = f_\theta(x) = P(Y = 1|x) = \sigma(x^T\theta)$$

# Example calculation

Suppose I want to predict the probability that LeBron's shot goes in, given **shot distance** (first feature) and **# of seconds left on the shot clock** (second feature).

I fit a logistic regression model using my training data, and somehow compute

$$\hat{\theta}^T = \begin{bmatrix} 0.1 & -0.5 \end{bmatrix}$$

Under the logistic model, compute the probability his shot goes in, given that

- He shoots it from 15 feet.
- There is 1 second left on the shot clock.

# Example calculation (solution)

$$x^T = [\,15 \qquad 1\,] \qquad \hat{\theta}^T = [\,0.1 \qquad -0.5\,]$$

$$\begin{aligned} P(Y=1|x) &= \sigma(x^T\hat{\theta}) \\ &= \sigma(\hat{\theta}_1 \cdot \text{SHOT DISTANCE} + \hat{\theta}_2 \cdot \text{SECONDS LEFT}) \\ &= \sigma(0.1 \cdot 15 + (-0.5) \cdot 1) \\ &= \sigma(1) \\ &= \frac{1}{1+e^{-1}} \\ &\approx 0.7311 \end{aligned}$$

An explicit expression representing our model.

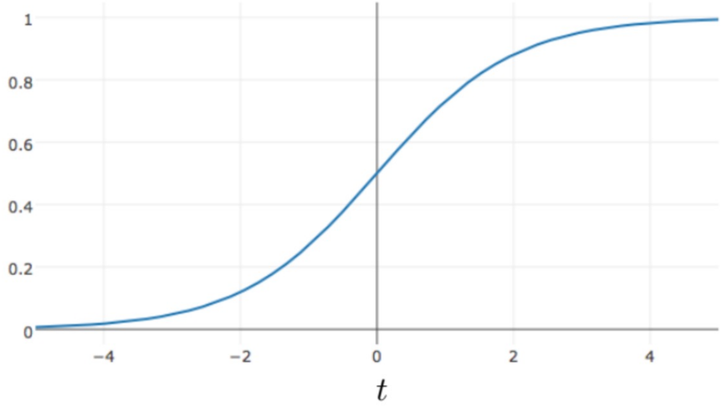
# Logistic Function

# Properties of the logistic function

The logistic function is a type of **sigmoid**, a class of functions that share certain properties.

$$\sigma(t) = \frac{1}{1 + e^{-t}} \qquad -\infty < t < \infty$$

- Its output is bounded between 0 and 1, no matter how large t is.
  - Fixes an issue with using linear regression to predict probabilities.
- We can interpret it as mapping real numbers to probabilities.



# Properties of the logistic function

**Definition**

$$\sigma(t) = \frac{1}{1 + e^{-t}} = \frac{e^t}{1 + e^t}$$

**Range**

$$0 < \sigma(t) < 1$$

**Inverse**

$$t = \sigma^{-1}(p) = \log\left(\frac{p}{1 - p}\right)$$

**Reflection and Symmetry**

$$1 - \sigma(t) = \frac{e^{-t}}{1 + e^{-t}} = \sigma(-t)$$

**Derivative**

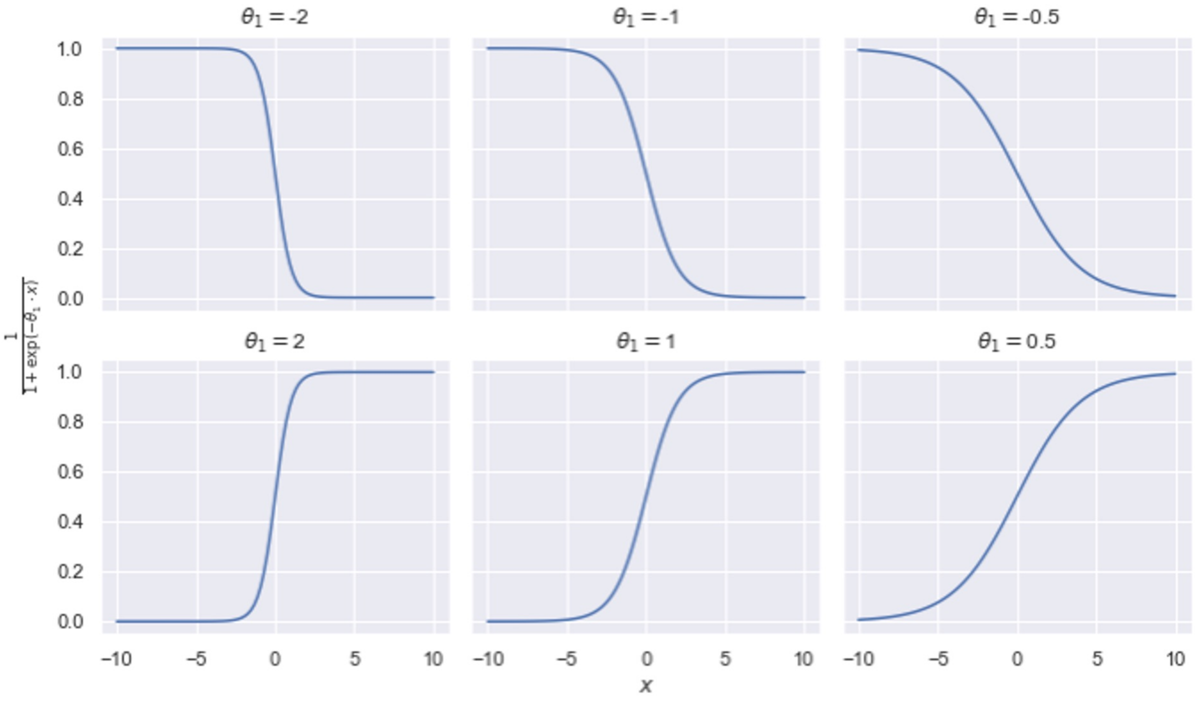
$$\frac{d}{dt}\sigma(t) = \sigma(t)(1 - \sigma(t)) = \sigma(t)\sigma(-t)$$

# Shape of the logistic function

Consider the plot of $\sigma(\theta_1 x)$ , for several different values of $\theta_1$.

- If $\theta_1$ is positive, the curve increases to the right.
- The further $\theta_1$ is from 0, the steeper the curve.

In the notebook, we explore more sophisticated logistic curves.



# Parameter interpretation

# Parameter interpretation

Recall, we arrived at the model by assuming that the **log-odds of the probability of belonging to class 1 was linear.**

$$P(Y=1|x) = \sigma(x^T\theta) \quad \Leftarrow \quad \log\left(\frac{P(Y=1|x)}{P(Y=0|x)}\right) = x^T\theta \quad \Leftarrow \quad \frac{P(Y=1|x)}{P(Y=0|x)} = e^{x^T\theta}$$

This is the same as $\frac{p}{1-p}$ because

$$P(Y=1|x) + P(Y=0|x) = 1$$

(Remember, we are dealing with binary classification – we are predicting 1 or 0.)

# Parameter interpretation

Let's suppose our linear component has just a single feature, along with an intercept term.

$$\frac{P(Y=1|x)}{P(Y=0|x)} = e^{\theta_0 + \theta_1 x}$$

What happens if you increase *x* by one unit?

- Odds is multiplied by $e^{\theta_1}$.
- If $\theta_1 > 0$, the odds increase.
- If $\theta_1 < 0$, the odds decrease.

What happens if $x^T\theta = \theta_0 + \theta_1 x = 0$?

- This means class 1 and class 0 are equally likely.
- $$e^0 = 1 \implies \frac{P(Y=1|x)}{P(Y=0|x)} = 1 \implies P(Y=1|x) = P(Y=0|x)$$

The odds ratio can be interpreted as the "number of successes for each failure."

# Loss Function

# Logistic Regression with squared loss?

1. Choose a model

**Logistic Regression** $\hat{y} = f_\theta(x) = P(Y=1|x) = \sigma(x^T\theta)$

2. Choose a loss function

$$R(\theta) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \sigma(\mathbb{X}_i^T\theta))^2$$

?

3. Fit the model

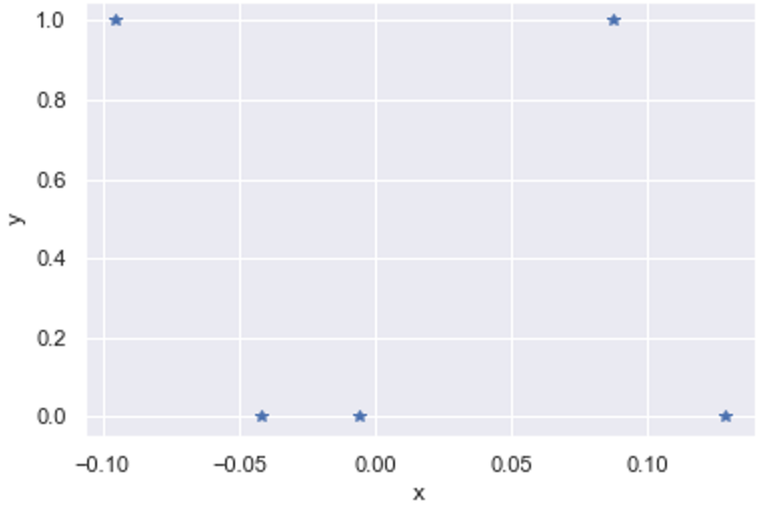
4. Evaluate model performance

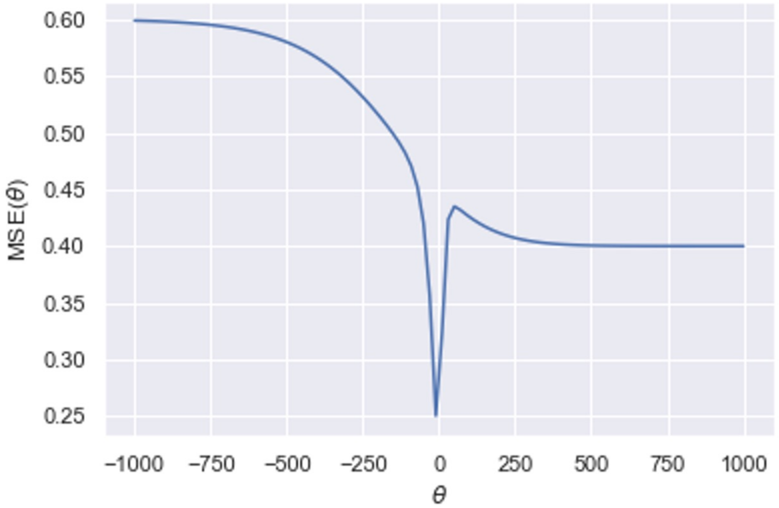
## Pitfalls of squared loss with logistic regression

On the left, we have a toy dataset (i.e. we've plotted the original data, y vs. x).
On the right, we have a plot of the mean squared error of this dataset when fitting a single-feature logistic regression model, for different values of $\theta$ (i.e. the loss surface).

What is the issue?

$$\mathrm{MSE}(\theta) = \frac{1}{5}\sum_{i=1}^{5}(y_i - \sigma(\theta x_i))^2$$

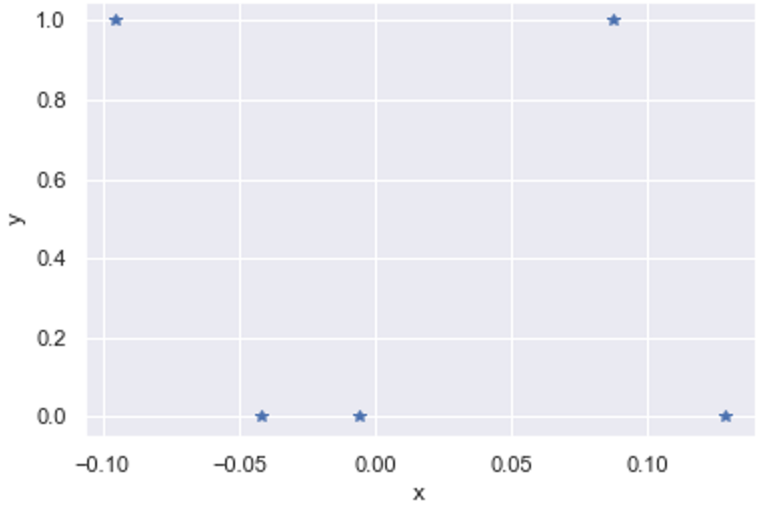


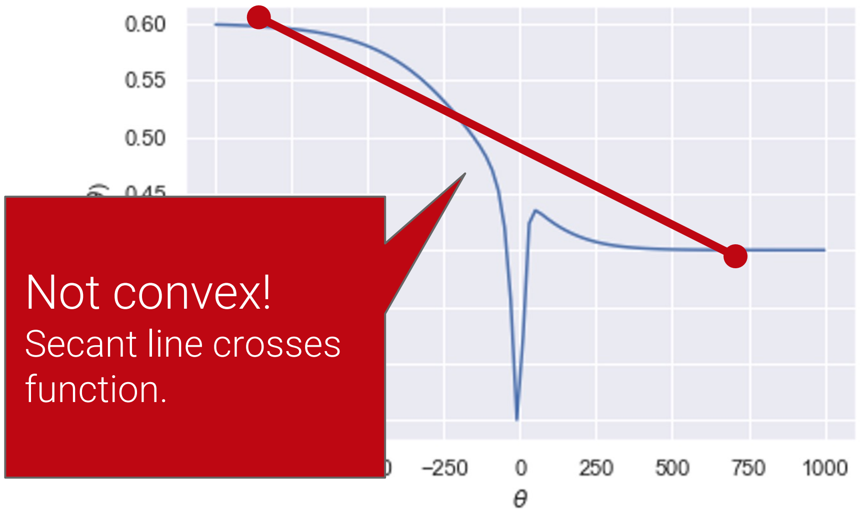



# Pitfalls of squared loss with logistic regression

On the left, we have a toy dataset (i.e. we've plotted the original data, y vs. x).
On the right, we have a plot of the mean squared error of this dataset when fitting a single-feature logistic regression model, for different values of $\theta$ (i.e. the loss surface).

$$\mathrm{MSE}(\theta) = \frac{1}{5}\sum_{i=1}^{5}(y_i - \sigma(\theta x_i))^2$$



# Pitfalls of squared loss with logistic regression

For this particular loss surface, different initial guesses for thetahat yield different "optimal values", as per scipy.optimize.minimize:

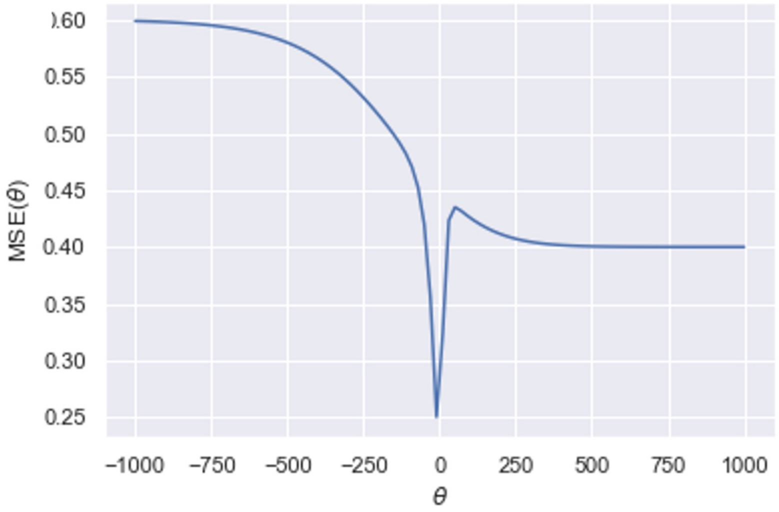
```
minimize(mse_loss_single_arg_toy, x0 = 0)["x"][0]
```

```
-4.801981341432673
```

```
minimize(mse_loss_single_arg_toy, x0 = 500)["x"][0]
```

```
500.0
```

This loss surface is not convex. We'd like it to be.



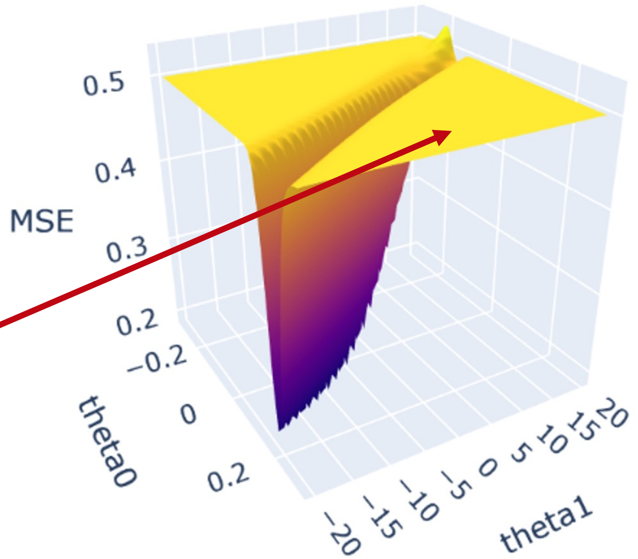
## Pitfalls of squared loss with logistic regression

The loss surface of MSE for a logistic regression model with a single slope plus an intercept often looks something like this.

If your initial guess for $\hat{\theta}$ is way out in the flat yellow region, routine can get stuck.

If the gradient is 0, your update rule will stop changing.

$$\theta^{(t+1)} = \theta^{(t)} - \alpha \boxed{\nabla_\theta R(\theta, \mathbb{X}, \mathbb{Y})}$$



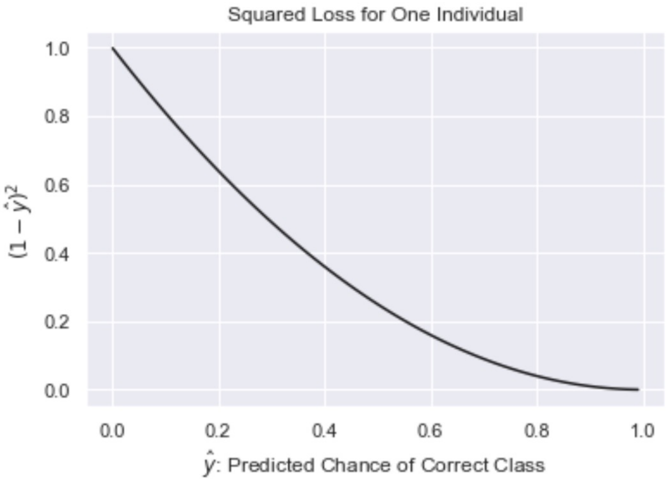
# Pitfalls of squared loss with logistic regression

Another issue: since $y_i$ is either 0 or 1, and $\hat{y}_i$ is between 0 and 1, $(y_i - \hat{y}_i)^2$ is also bounded between 0 and 1.

- Even if our probability is nowhere close, the loss isn't that large in magnitude.
    - If we say the probability is 10^-6, but it happens anyway, error should be large.
- We want to penalize wrong answers significantly.

Suppose the observation we're trying to predict is actually in **class 1**.

On the right, we have a plot of $(1 - \hat{y})^2$ vs $\hat{y}$ .
This is the squared loss for a single prediction.



## Summary of issues with squared loss and logistic regression

While it can work, squared loss is not the best choice of loss function for logistic regression.

- Average squared loss is not convex.
    - Numerical methods will struggle to find a solution.
- Wrong predictions aren't penalized significantly enough.
    - Squared loss (and hence, average squared loss) are bounded between 0 and 1.

Fortunately, there's a solution.

# Logistic Regression with cross entropy

1. Choose a model

**Logistic Regression** $\hat{y} = f_\theta(x) = P(Y=1|x) = \sigma(x^T\theta)$

2. Choose a loss function

$$\cancel{R(\theta) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \sigma(\mathbb{X}_i^T\theta))^2}$$

$$\text{loss} = -y\log(\hat{y}) - (1-y)\log(1-\hat{y})$$

$$R(\theta) = -\frac{1}{n}\sum_{i=1}^{n}\left(y_i\log(\sigma(\mathbb{X}_i^T\theta)) + (1-y_i)\log(1-\sigma(\mathbb{X}_i^T\theta))\right)$$

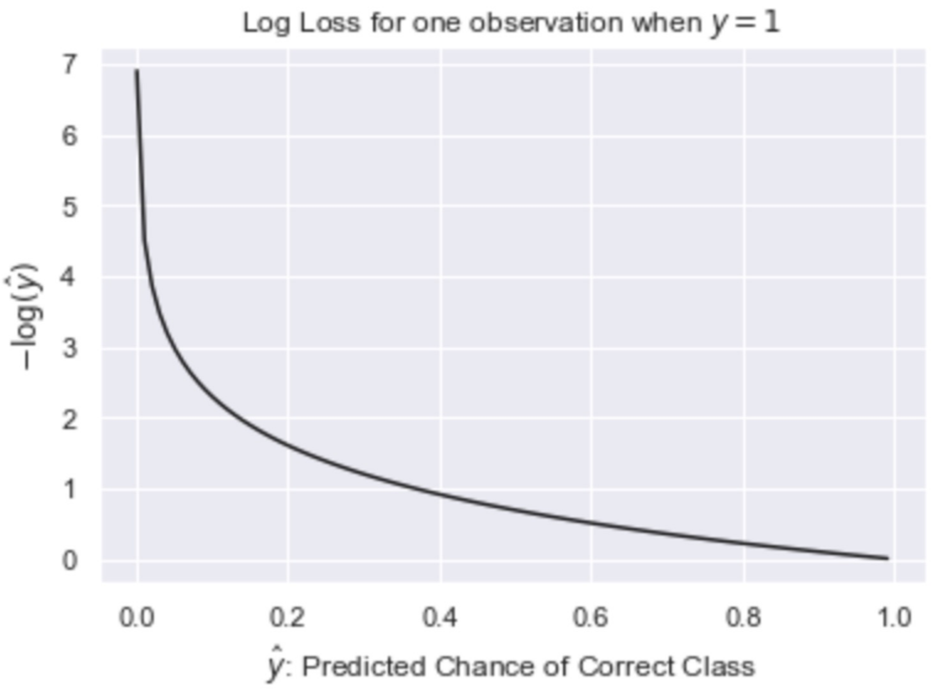
3. Fit the model

4. Evaluate model performance

# Log loss

Consider this new loss, called the (negative) **log loss**, for a single observation when the true y is equal to 1.

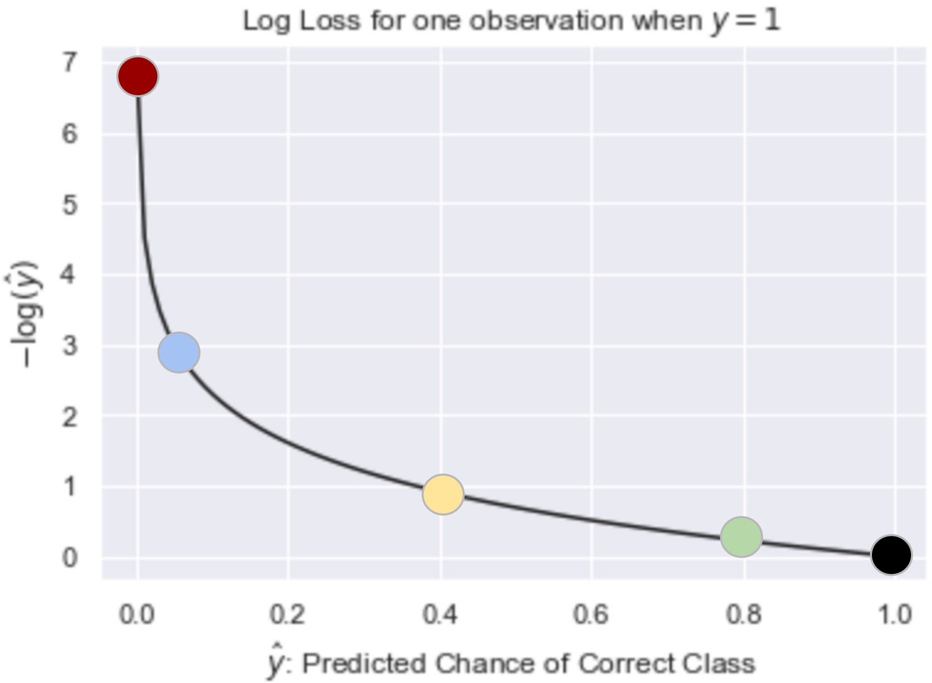
We can see that as our prediction gets further and further from 1, the loss approaches infinity (unlike squared loss, which maxed out at 1).



# Log loss

Let's look at some losses in particular:

| $\hat{y}$ | $-\log(\hat{y})$ |
|---|---|
| 1 | 0 |
| 0.8 | 0.25 |
| 0.4 | 1 |
| 0.05 | 3 |
| 0 | infinity! |

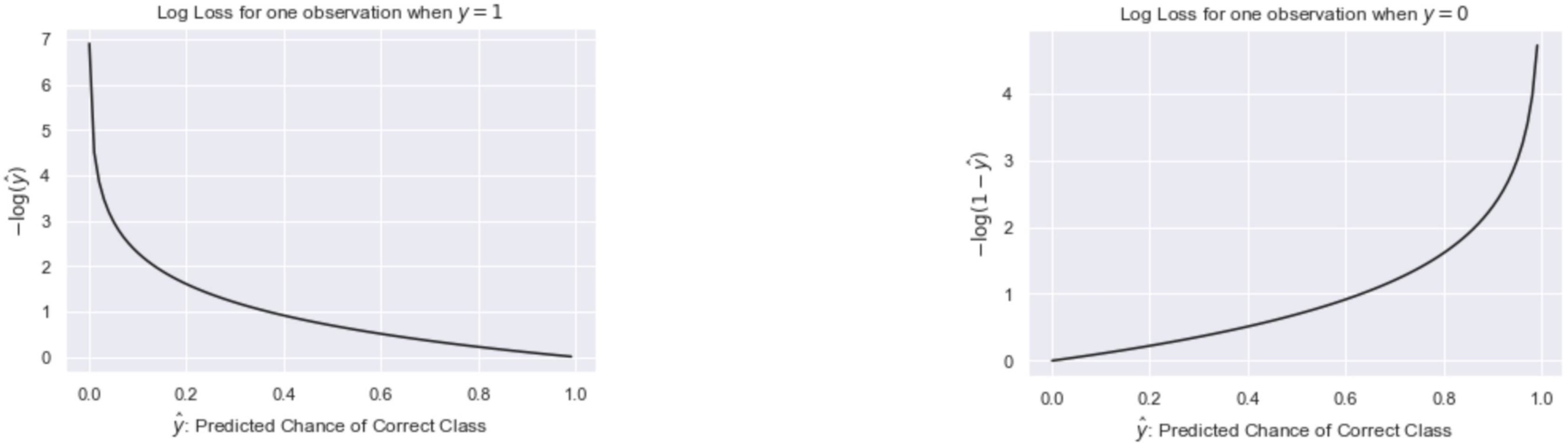



**Note:** The logistic function never outputs 0 or 1 exactly, so there's never actually 0 loss or infinite loss.

# Log loss

So far, we've only looked at log loss when the correct class was 1.

**What if our correct class is 0?**




If the correct class is 0, we want to have low loss for values of $\hat{y}$ close to 0, and high loss for values of $\hat{y}$ close to 1. This is achieved by just "flipping" the plot on the left!

# Cross-entropy loss

We can combine the two cases from the previous slide into a single loss function:

$$\text{loss} = \begin{cases} -\log(1-\hat{y}) & y = 0 \\ -\log(\hat{y}) & y = 1 \end{cases}$$

This is often written unconditionally as:

$$\boxed{\text{loss} = -y\log(\hat{y}) - (1-y)\log(1-\hat{y})}$$

*Note: Since y = 0 or 1, one of these two terms is always equal to 0, which reduces this equation to the piecewise one above.*

We call this loss function **cross-entropy** loss (or "log loss").

## Mean cross-entropy loss

The empirical risk for the logistic regression model when using cross-entropy loss is then

$$R(\theta) = -\frac{1}{n}\sum_{i=1}^{n}\left(y_i \log(\sigma(\mathbb{X}_i^T\theta)) + (1 - y_i)\log(1 - \sigma(\mathbb{X}_i^T\theta))\right)$$

Benefits over mean squared error for logistic regression:

- Loss surface is guaranteed to be convex.
- More strongly penalizes bad predictions.
- Has roots in probability and information theory (next section).

# Logistic Regression with cross entropy

1. Choose a model

2. Choose a loss function

3. Fit the model

4. Evaluate model performance

**Logistic Regression** $\hat{y} = f_\theta(x) = P(Y=1|x) = \sigma(x^T\theta)$

$$R(\theta) = -\frac{1}{n}\sum_{i=1}^{n}\left(y_i \log(\sigma(\mathbb{X}_i^T\theta)) + (1-y_i)\log(1-\sigma(\mathbb{X}_i^T\theta))\right)$$

For logistic regression, we can use squared loss if we want to!

- Using squared loss and using cross-entropy loss will usually result in different $\hat{\theta}$
  - **Different optimization problems, different solutions**.
- **Cross-entropy loss is strictly better than squared loss for logistic regression.**
  - Convex, so easier to minimize using numerical techniques.
  - Better suited for modeling probabilities.

# Maximum Likelihood Estimation

## Where did log loss come from?

Log loss seemed to have come out of thin air.

- It seems to make a lot of sense for a model that predicts probabilities!
- Let’s derive where it came from.

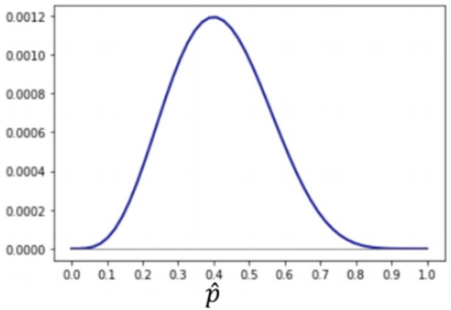
# Estimating the chance of success

Suppose I have a coin that flips heads with probability $p$. Assume each flip is independent.

- I don't know what $p$ is, but I flip the coin 10 times and I see 0001011001 (4 heads, 6 tails).
- Can model each flip with an i.i.d. Bernoulli($p$) random variable (1 for heads, 0 for tails).
- What is the most likely value of $p$?

$$L(p) = p^4(1-p)^6$$

This function is called the **likelihood** of our observed sequence.



# Estimating the chance of success

Suppose I have a coin that flips heads with probability $p$. Assume each flip is independent.

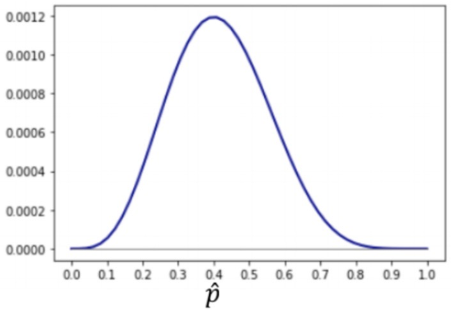
- I don't know what $p$ is, but I flip the coin 10 times and I see 0001011001 (4 heads, 6 tails).
- Can model each flip with an i.i.d. Bernoulli($p$) random variable (1 for heads, 0 for tails).
- What is the most likely value of $p$?

$$L(p) = p^4(1-p)^6$$

This function is called the **likelihood** of our observed sequence.

We'd estimate $\hat{p}$ = 0.4.

- Sample proportion of 1s.
- Maximizes likelihood function over all p.



# Two different coins

- Toss a coin that lands heads with chance $p_1$.
  - Result: $Y_1$.
- Toss a coin that lands heads with chance $p_2$.
  - Result: $Y_2$.
- What are the probabilities for all possible combinations of values?

$$P(Y_1 = 1, Y_2 = 1) = p_1 p_2$$

$$P(Y_1 = 1, Y_2 = 0) = p_1(1 - p_2)$$

$$P(Y_1 = 0, Y_2 = 1) = (1 - p_1)p_2$$

$$P(Y_1 = 0, Y_2 = 0) = (1 - p_1)(1 - p_2)$$

# PMF of the Bernoulli distribution

If $Y$ is the result of one toss of a coin that lands heads with chance $p$,

$$P(Y=y) = \begin{cases} p & \text{if } y=1 \\ 1-p & \text{if } y=0 \end{cases}$$

$$= p^y(1-p)^{1-y}$$

Then, if $Y_1$ and $Y_2$ are the results of tosses of two coins:

$$P(Y_1=y_1, Y_2=y_2) = \left(p_1^{y_1}(1-p_1)^{1-y_1}\right)\left(p_2^{y_2}(1-p_2)^{1-y_2}\right)$$

# Estimating the two probabilities

- Suppose we want to estimate the values of $p_1$ and $p_2$.
- We know what the likelihood is.

$$P(Y_1 = y_1, Y_2 = y_2) = \left(p_1^{y_1}(1-p_1)^{1-y_1}\right)\left(p_2^{y_2}(1-p_2)^{1-y_2}\right)$$

- Our goal is to find the $p_1$ and $p_2$ that **maximize** the above function, over all $p_1$ and $p_2$.
  - Maximize, because we are looking for the $p_1$ and $p_2$ that are “most likely” to have generated the data that we observed.
- As before, this involves differentiating, setting equal to 0, and solving.

# Log likelihoods

- Maximizing $P(Y_1 = y_1, Y_2 = y_2) = \left(p_1^{y_1}(1-p_1)^{1-y_1}\right)\left(p_2^{y_2}(1-p_2)^{1-y_2}\right)$ is annoying.
  - Products -> chain rule.
- log(x) is a **strictly increasing** function.
  - If a > b, then log(a) > log(b).
- This means, the values of $p_1$ and $p_2$ that maximize $P(Y_1 = y_1, Y_2 = y_2)$ are the same values that maximize

$$\log\left(\left(p_1^{y_1}(1-p_1)^{1-y_1}\right)\left(p_2^{y_2}(1-p_2)^{1-y_2}\right)\right)$$

$$= y_1 \log(p_1) + (1-y_1)\log(1-p_1) + y_2 \log(p_2) + (1-y_2)\log(1-p_2)$$

$$= \sum_{i=1}^{2} \left(y_i \log(p_i) + (1-y_i)\log(1-p_i)\right)$$

Starting to look familiar!

## Estimating $n$ probabilities

- For $i = 1, 2, \ldots, n$, let $Y_i$ be Bernoulli($p_i$).
  - Each $Y_i$ is independent of each other.
- To estimate $p_1, p_2, \ldots, p_n$

$$\text{Find } \hat{p}_1, \hat{p}_2, \ldots, \hat{p}_n \text{ that maximize } \sum_{i=1}^{n} \left( y_i \log(p_i) + (1 - y_i) \log(1 - p_i) \right)$$

**Equivalently:**

$$\text{Find } \hat{p}_1, \hat{p}_2, \ldots, \hat{p}_n \text{ that } \textit{minimize } -\frac{1}{n} \sum_{i=1}^{n} \left( y_i \log(p_i) + (1 - y_i) \log(1 - p_i) \right)$$

We choose this equivalent form because we are more used to minimizing loss.

# Maximum likelihood estimation

Minimizing cross-entropy loss is equivalent to maximizing the likelihood of the data.

- We are choosing the model parameters that are "most likely", given this data.
- Another perspective of fitting our model to the data.

$$R(\theta) = -\frac{1}{n}\sum_{i=1}^{n}\left(y_i \log(\sigma(\mathbb{X}_i^T\theta)) + (1-y_i)\log(1-\sigma(\mathbb{X}_i^T\theta))\right)$$

This technique is called **maximum likelihood estimation (MLE)**.

- It turns out, many of the model + loss combinations we've seen in this class can be motivated using MLE.
  - OLS, Ridge Regression.
- You will study this further in probability and ML classes. But now you know it exists.

# Solving Maximum Likelihood Estimation (Not required)

$$R(\theta) = -\frac{1}{n}\sum_{i=1}^{n}\left(y_i \log(\sigma(\mathbb{X}_i^T\theta)) + (1-y_i)\log(1-\sigma(\mathbb{X}_i^T\theta))\right)$$

- Approach 1: Gradient Descent (take larger – more certain – steps opposite the gradient)
- Approach 2: Stochastic Gradient Descent (SGD) (take many small steps opposite the gradient)
- Approach 3: Newton’s Method (use second derivatives to better follow curvature)

| | |
|---|---|
| 1. Choose a model | **Logistic Regression** $\hat{y} = f_\theta(x) = P(Y=1\|x) = \sigma(x^T\theta)$ |
| 2. Choose a loss function | $R(\theta) = -\frac{1}{n}\sum_{i=1}^{n}\left(y_i \log(\sigma(\mathbb{X}_i^T\theta)) + (1-y_i)\log(1-\sigma(\mathbb{X}_i^T\theta))\right)$ |
| 3. Fit the model | ✅ |
| 4. Evaluate model performance | Next time! |