# Lecture 2:
# Model Building and Probability Refresher

David Carlson

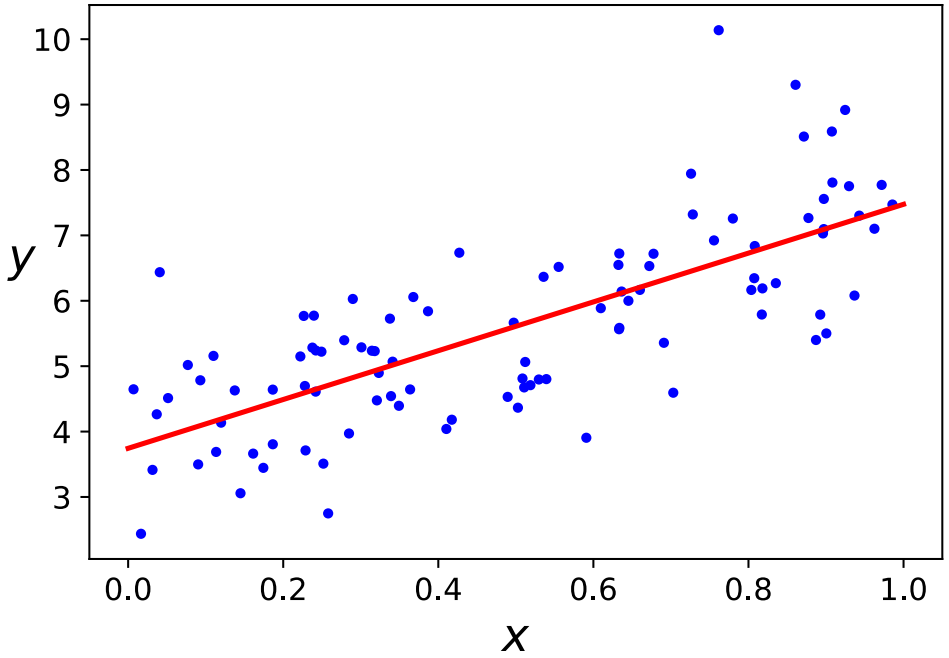
CEE 690-05: HEDS

# Refresher on Predictive Modeling

- Mathematical background necessary for talking about data, statistical modeling, and machine learning
- Approach today through thinking about linear regression
- Should be review material
  - Way I talk about statistical models is often *quite* different than a statistics class

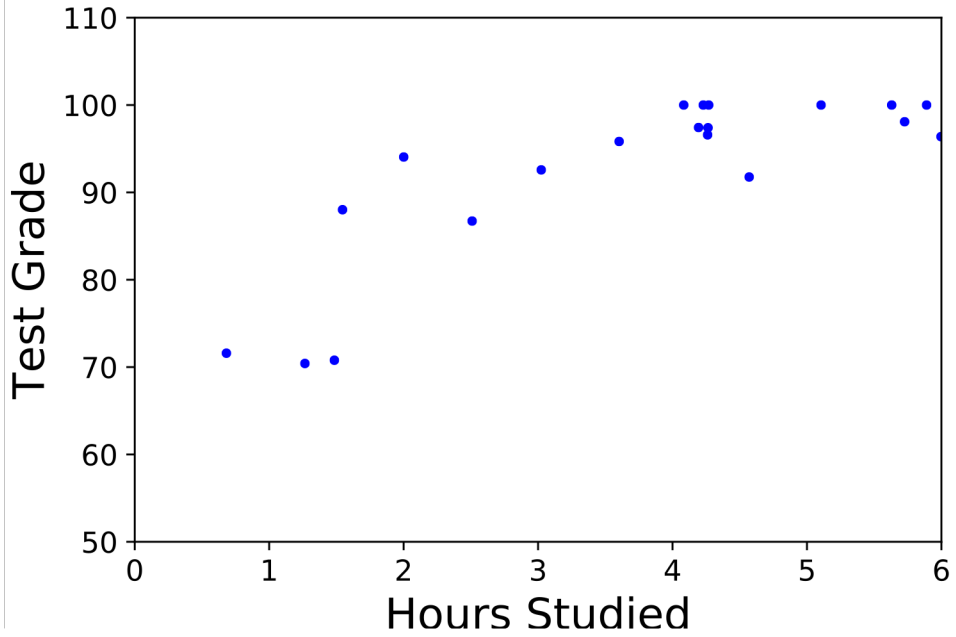
# Linear Regression

- One of the earliest (and most widely used!) statistical models.
- Use to ask a variety of questions, e.g.:
  - Is smoking related to cancer?
  - Does this gene relate to height? Or to blood pressure? Or to IQ?
  - Nearly everything…
- Basis of many idea in statistics and the *underlying* (but not complete) model in many research areas, such as genomics
- Everyone should have seen linear regression before



# Predicting Test Scores

- Suppose that we want to predict test scores from time spend studying
- How would we set up linear regression to do that?



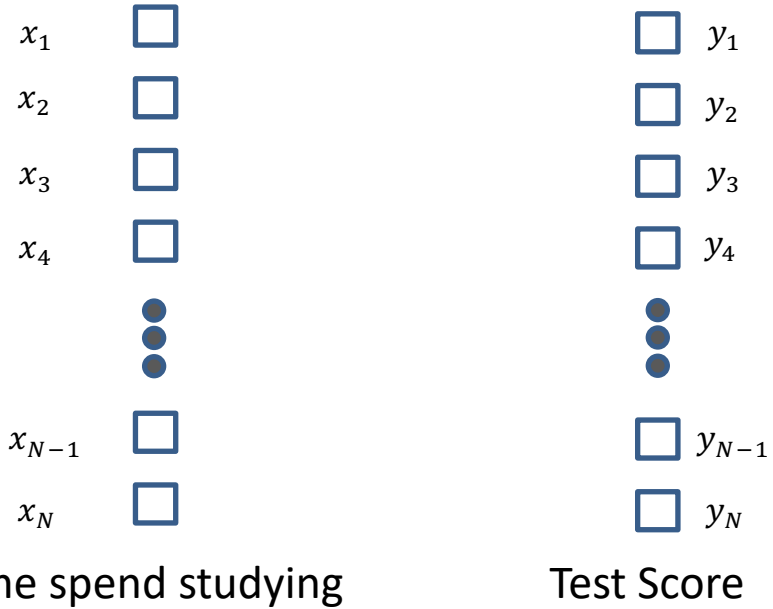
# What are we trying to do?

$x$, data for a subject (i.e. amount of time studying)

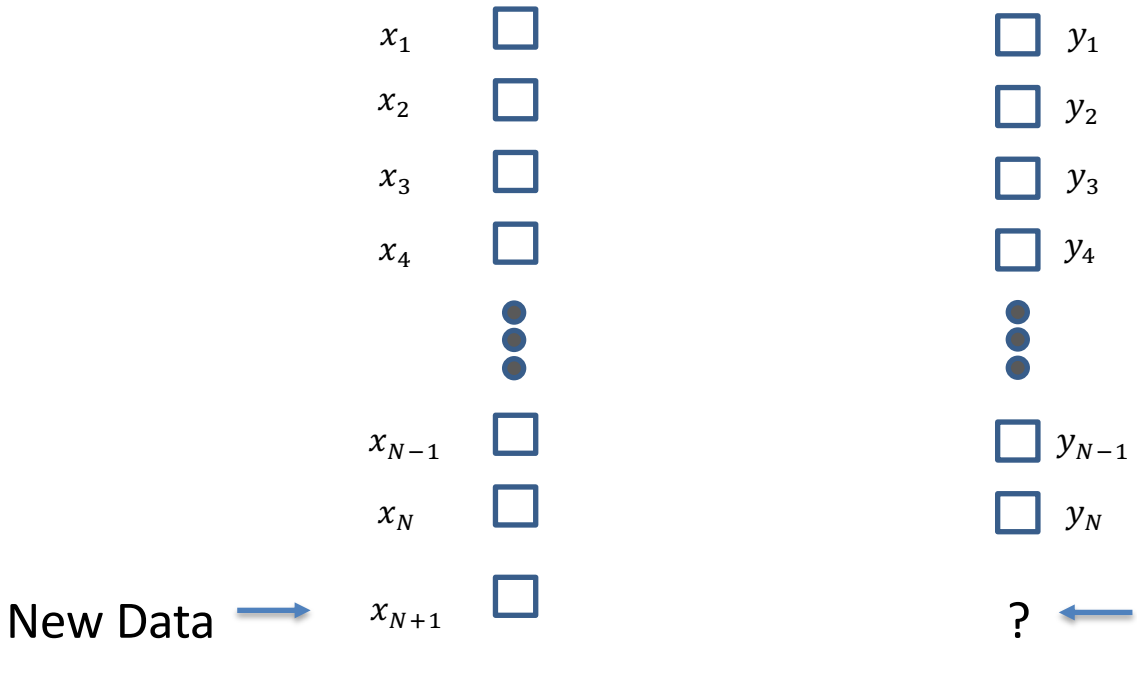
$y$, associated test score

End goal: *predict* $y$ from $x$

# Training Dataset/ Historical Examples



# Training Dataset/ Historical Examples



# Linear Regression

- Linear regression chooses the function:

$$\hat{y} = f(x; \beta_0, \beta_1) = \beta_0 + \beta_1 x,$$

- Where $\beta_0$ and $\beta_1$ are constant parameters and $t$ is a variable.

- Need to choose some way of "learning" what $\beta_0$ and $\beta_1$ are for this data problem (i.e. statistical inference)

# Minimizing (Mean) Squared Error

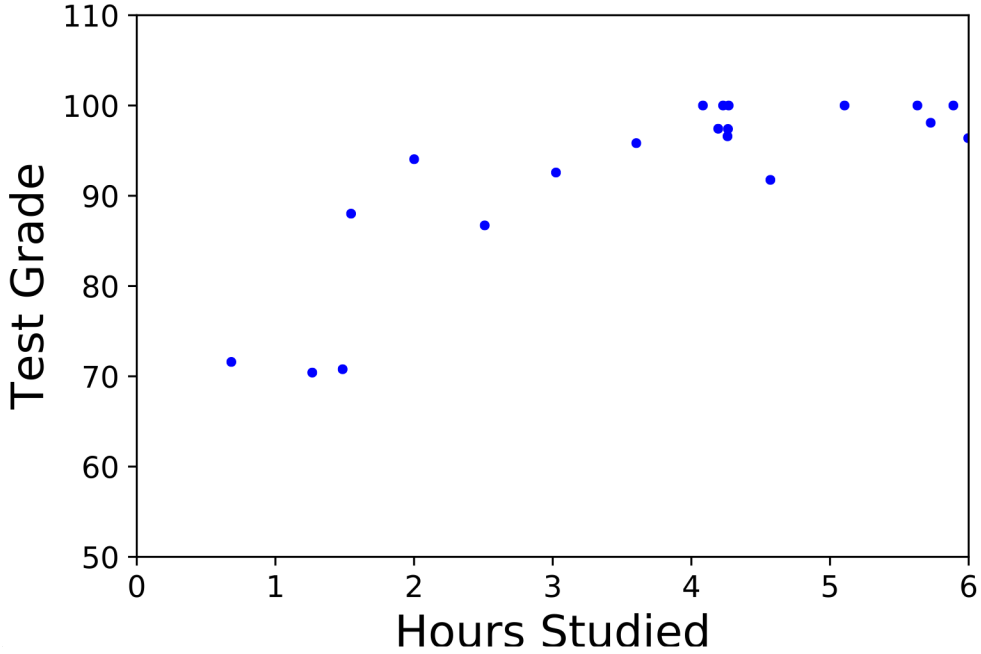
- In linear regression, we nearly* always choose to minimize the squared error over all observed data points:

$$\beta_0^*, \beta_1^* = \arg \min_{\beta_0, \beta_1, \hat{y} = \beta_0 + \beta_1 x} \sum_{i=1}^{N} (y_i - \hat{y})^2$$

- Why is this the square? (will come back to)
- How do we actually solve this? (will come back to)

# Fitting a “simple” linear regression

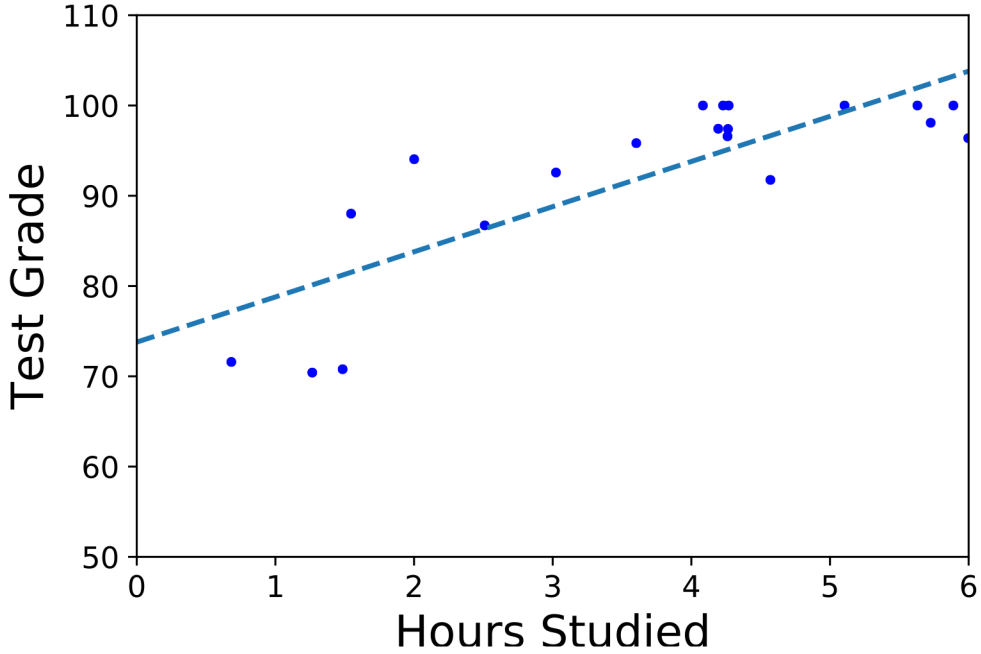
Suppose we have the data to the right, what does our linear regression look like here?



# Fitting a “simple” linear regression

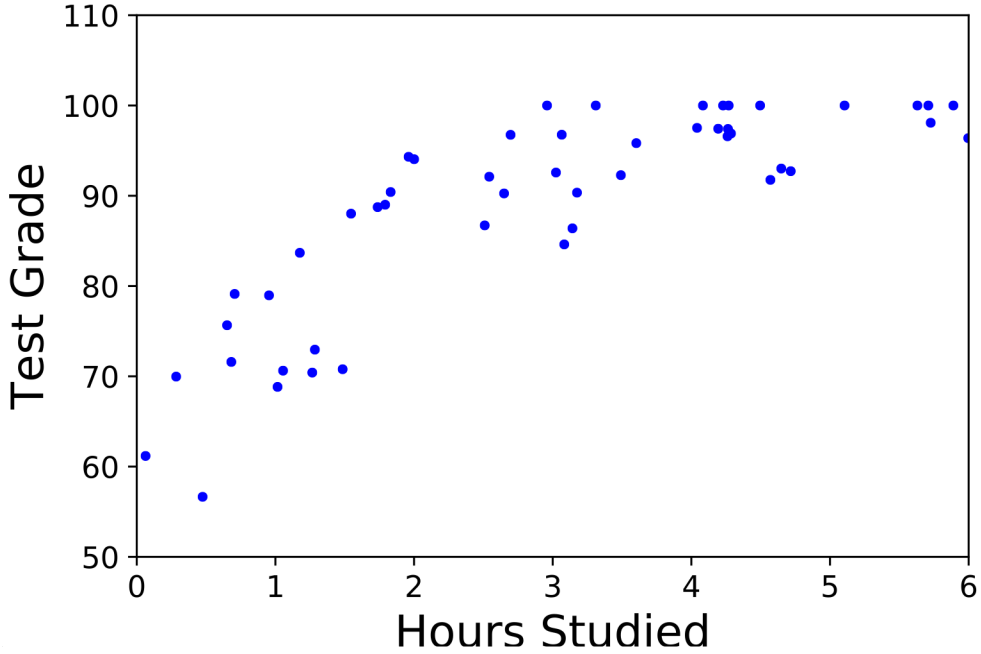
Suppose we have the data to the right, what does our linear regression look like here?

The linear regression fit does a “reasonable” job predicting scores here.



# Can keep updating our model as we see more examples

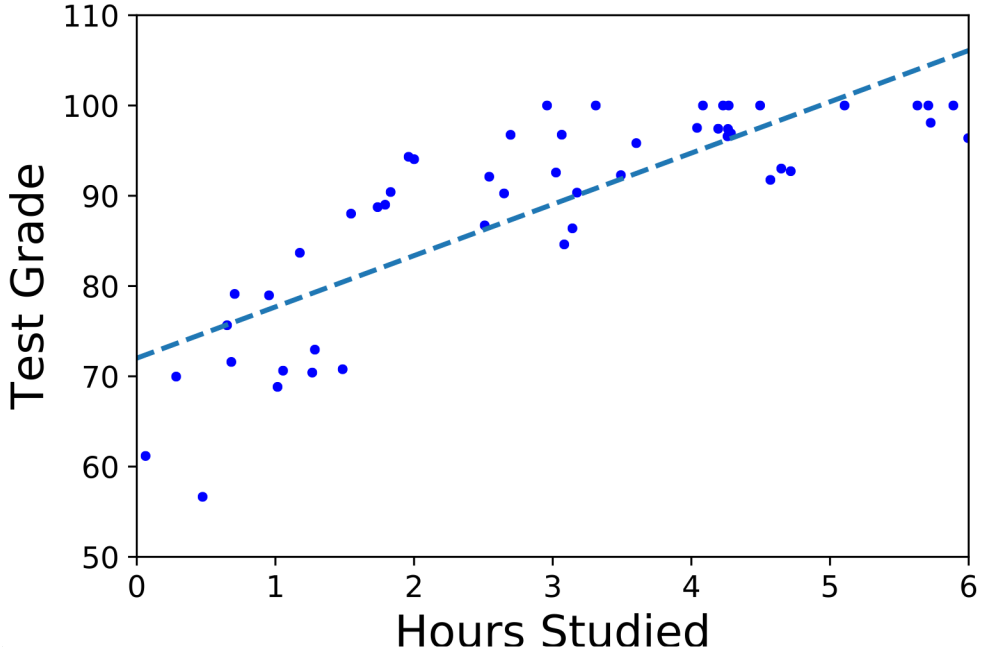
As data keeps coming in, we can use our additional examples (i.e. past experiences) to make improved predictions about the future.



# Can keep updating our model as we see more examples

As data keeps coming in, we can use our additional examples (i.e. past experiences) to make improved predictions about the future.

Fitting linear regression again…

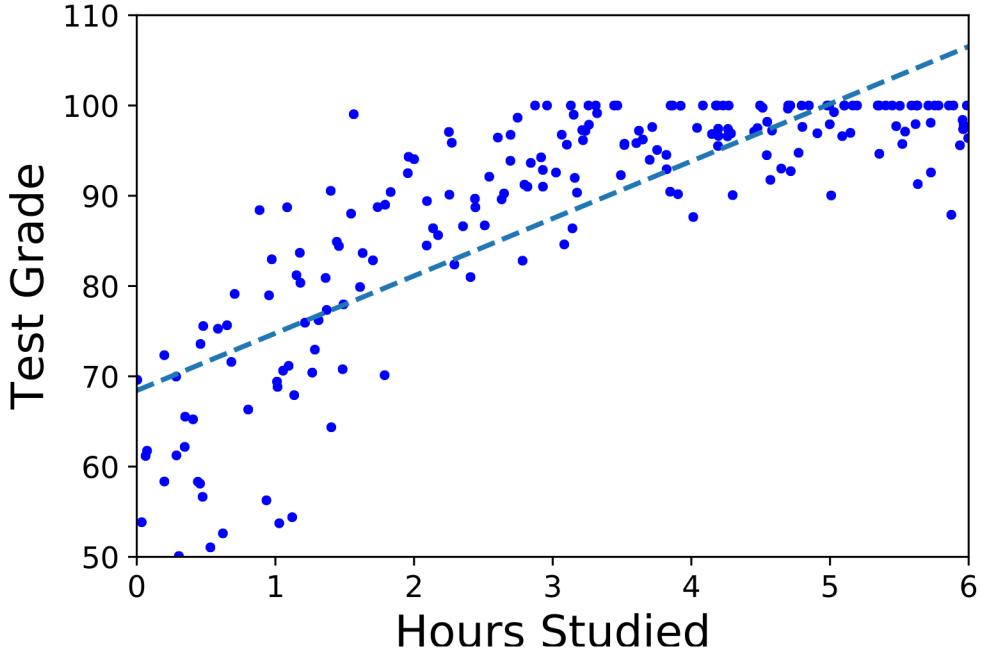


# Can keep updating our model as we see more examples

As data keeps coming in, we can use our additional examples (i.e. past experiences) to make improved predictions about the future.

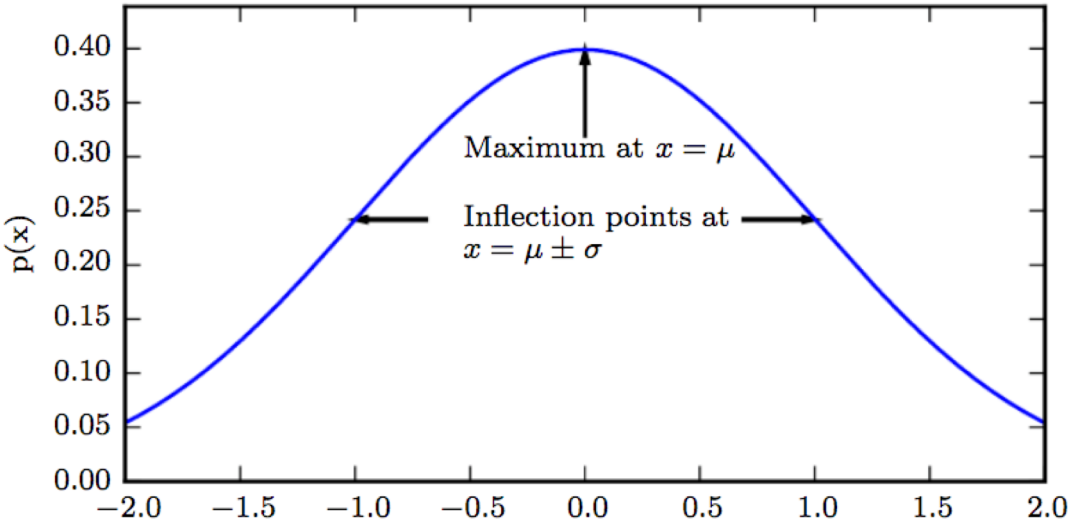
Adding much more data…

Is this still a reasonable data fit?



# Why do we used squared error?

- It all has to do with our normal/ Gaussian/ bell curve distribution.
- Often we say that errors are “normally distributed”



# Quick Refresh on Normal Distribution

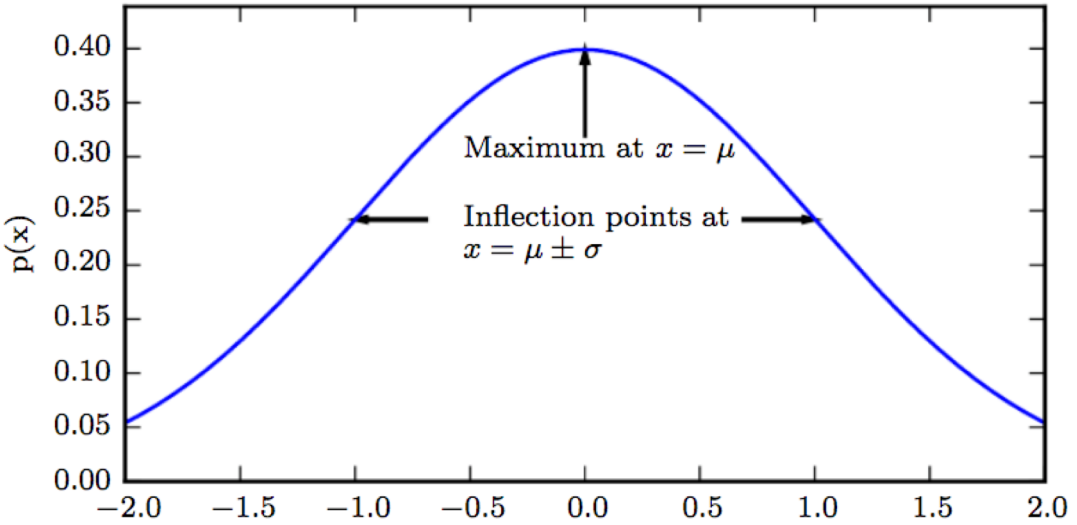
- A variable with a normal distribution is defined:

$$x \sim N(\mu, \sigma^2)$$

- With probability density function:

$$p(x; \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{1}{2\sigma^2}(x-\mu)^2\right)$$

- The “standard normal” is shown at the right with $\mu = 0$ and $\sigma^2 = 1$



# Linear Regression

- We often implicitly consider the statistical model

$$y = \beta_0 + \beta_1 x + \epsilon$$

- The "error" $\epsilon$ is normally distributed, where

$$\epsilon = y - \beta_0 + \beta_1 x$$

# Maximum Probability

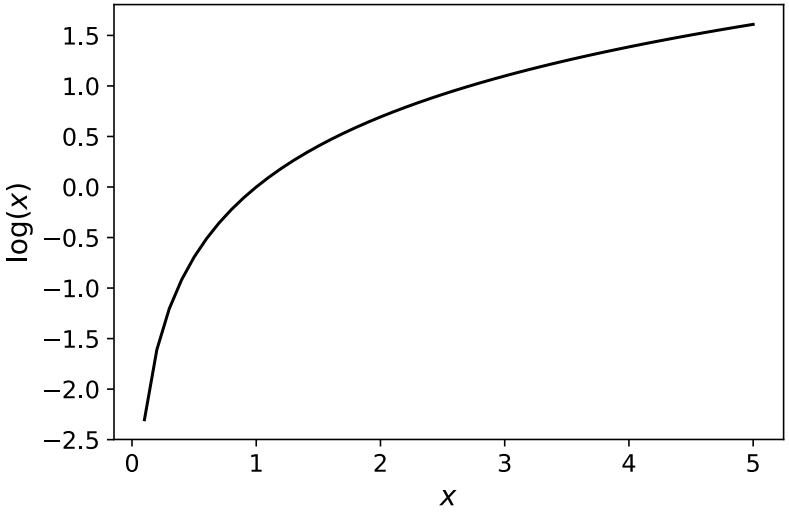
- We want to *maximize* statistical probability of our data
  - (i.e. our observations were as *likely* as possible under our model)
- This means

$$\beta_0^*, \beta_1^* = \arg \max_{\beta_0, \beta_1, \epsilon = y - \beta_0 - \beta_1 x} \prod_{i=1}^{N} p(\epsilon_i)$$

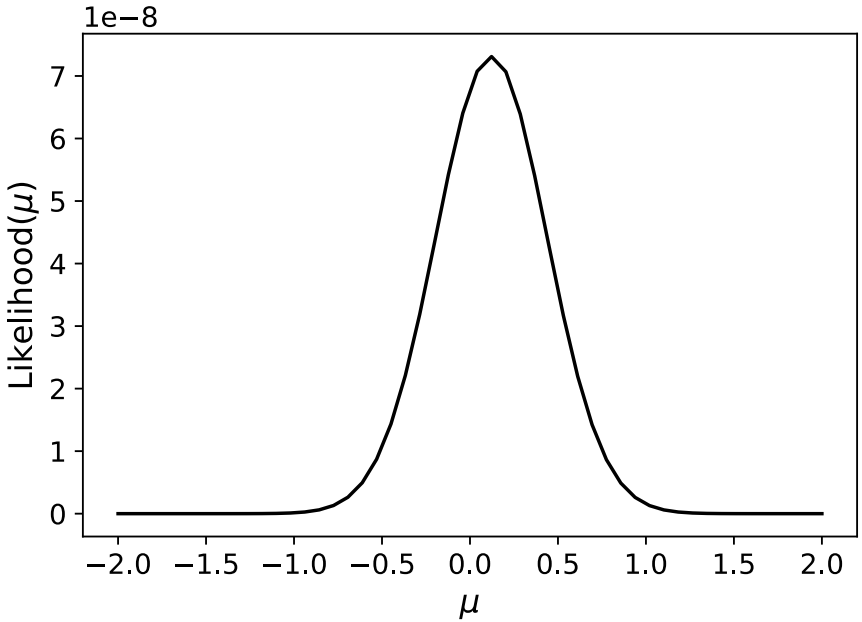
$$\beta_0^*, \beta_1^* = \arg \max_{\beta_0, \beta_1, \epsilon = y - \beta_0 - \beta_1 x} \prod_{i=1}^{N} N(\epsilon_i; 0, \sigma^2)$$

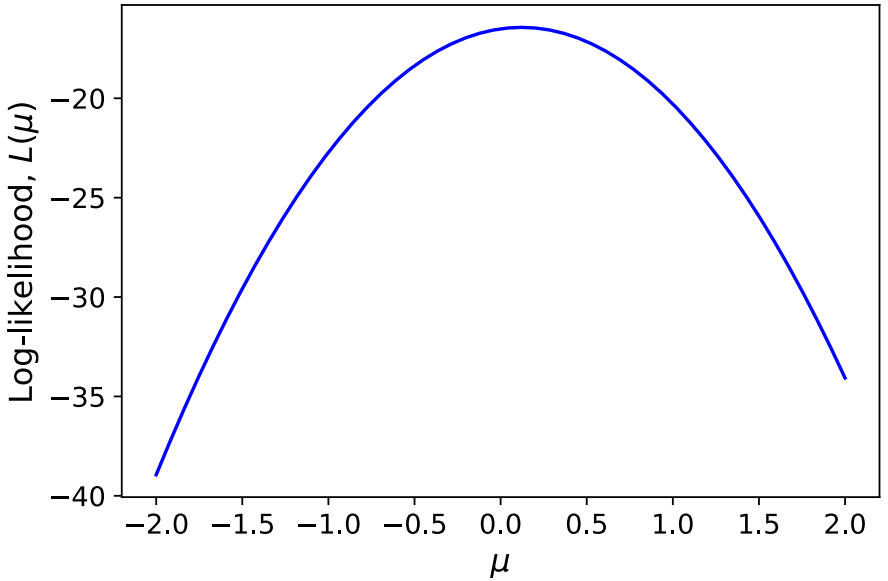
# Monotonic transformations

- Dealing with products that arise in log-likelihoods is frustrating mathematically, instead we use the $log$ function to transform the likelihood.
- Because the function is monotonic, the location of the maximum point does not change, which we'll visualize on the next slide.



# Normal (Log-)Likelihood



# Maximum Log-Likelihood

- This means

$$\beta_0^*, \beta_1^* = \arg \max_{\beta_0, \beta_1, \epsilon = y - \beta_0 - \beta_1 x} \log \prod_{i=1}^{N} p(\epsilon_i)$$

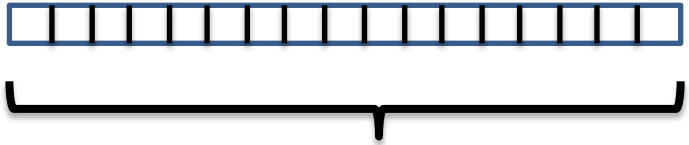
$$\beta_0^*, \beta_1^* = \arg \max_{\beta_0, \beta_1, \epsilon = y - \beta_0 - \beta_1 x} \sum_{i=1}^{N} \log p(\epsilon_i)$$

$$\beta_0^*, \beta_1^* = \arg \min_{\beta_0, \beta_1, \hat{y} = \beta_0 + \beta_1 x} const_1 + const_2 \sum_{i=1}^{N} (y_i - \hat{y})^2$$

# Some considerations

- We assume that we can solve the previous math equation
- Often, we have many features of a data (e.g hours studied, grade on the last test, grade on the last homework,…)
- Often, we care about predicting other types of outcomes, such as binary (positive/negative)

# Learning a Predictive Model with Many Features

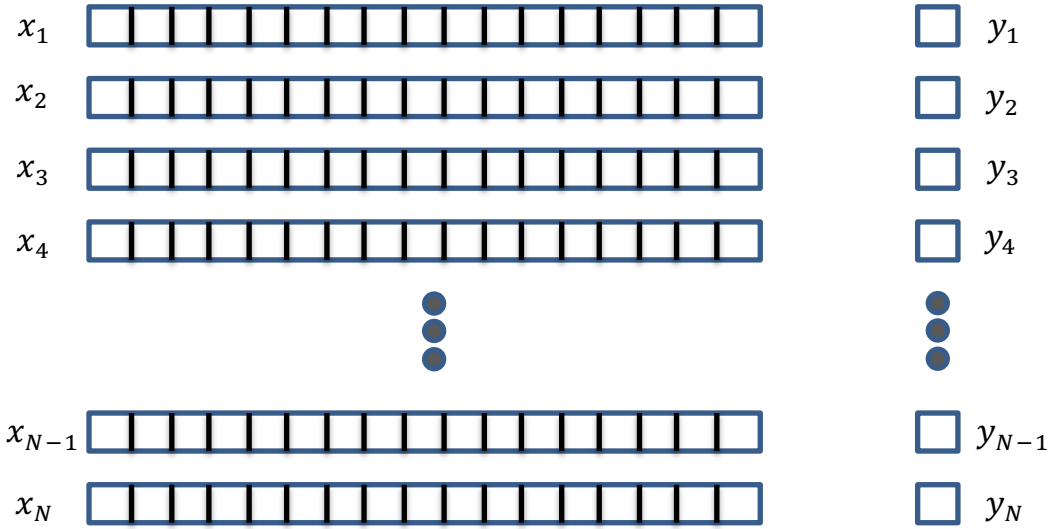


$x$, data/features for a subject

$y$, associated label 0/1

End goal: *predict* $y$ from $x$

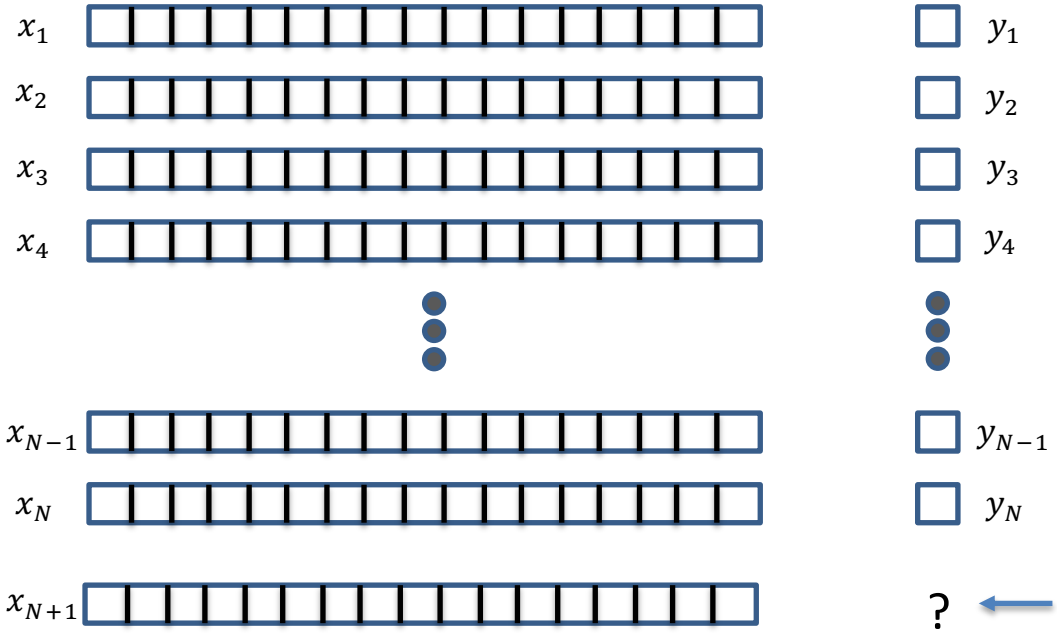
# Training Set (Historical Data)




Start by limiting $y$ to a binary outcome:

- False/True
- 0/1

# Making Predictions



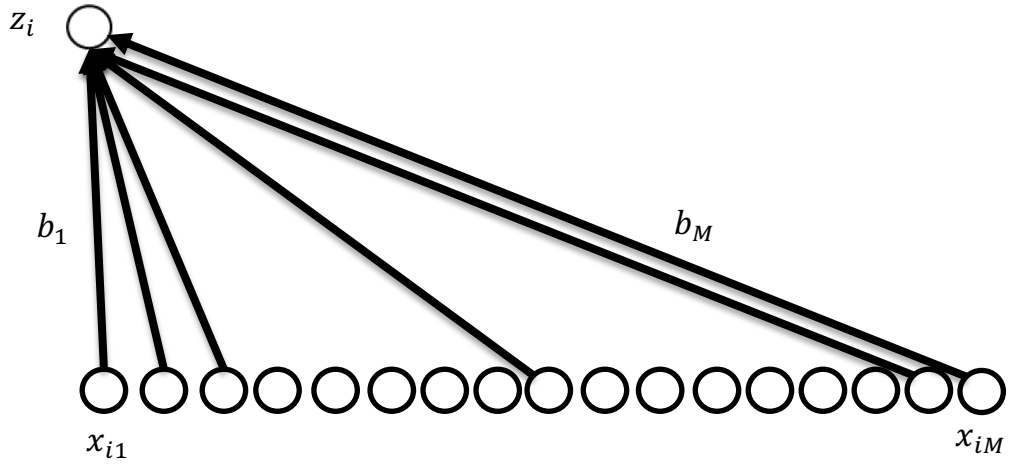

Start by limiting $y$ to a binary outcome:

- False/True
- 0/1

New Data

Want to learn how to *predict* outcome

# Linear Predictive Model



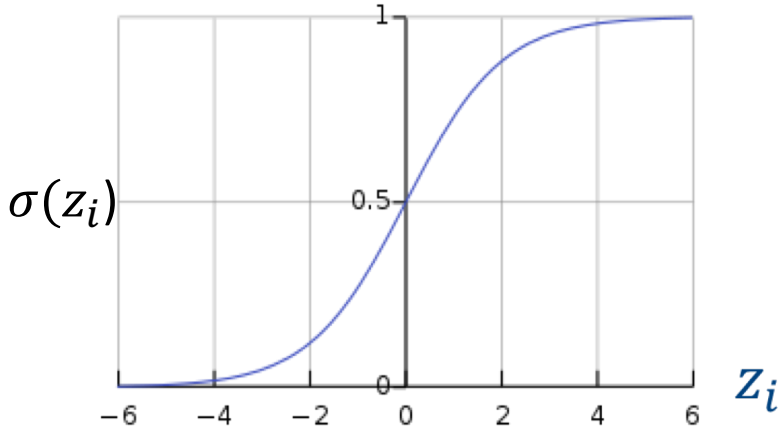

$$z_i = (b_1 \times x_{i1}) + (b_2 \times x_{i2}) + \cdots + (b_M \times x_{iM})$$

# Convert to a Probability

$$z_i = (b_1 \times x_{i1}) + (b_2 \times x_{i2}) + \cdots + (b_M \times x_{iM}) + b_0$$

Extra Constant

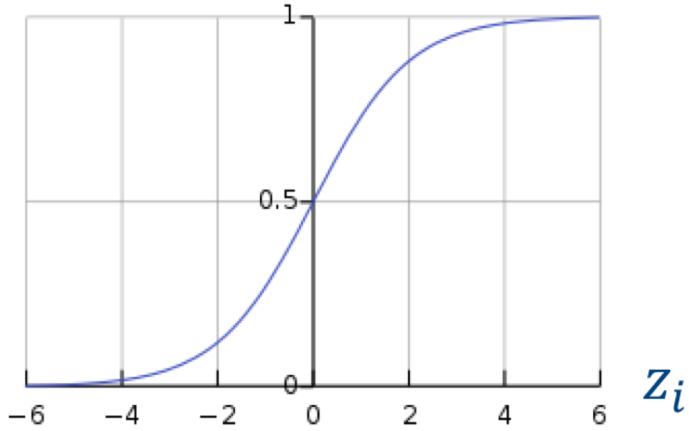
$$p(y_i = 1|x_i) = \sigma(z_i)$$



# Convert to a Probability

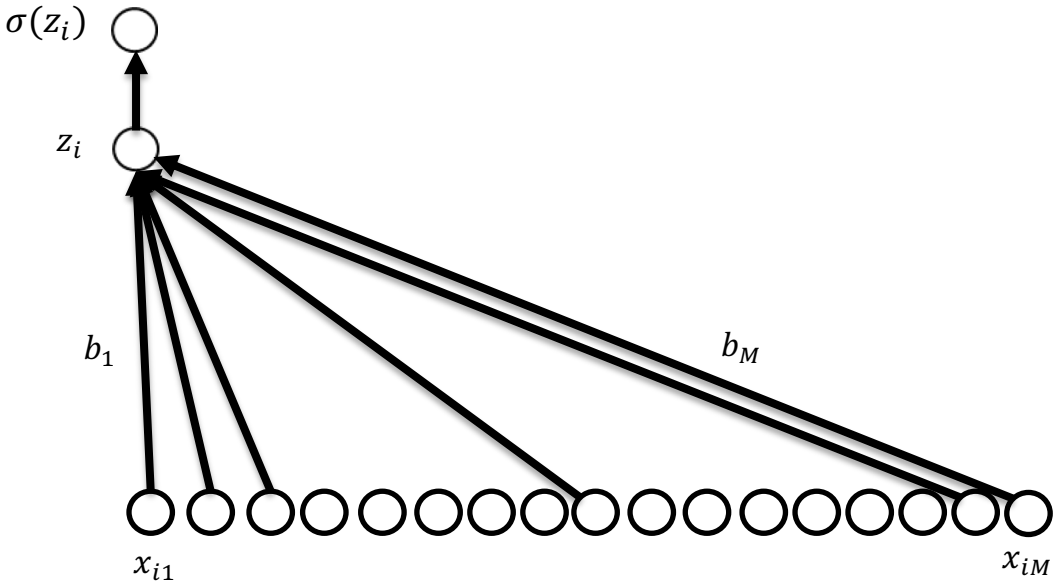
$$z_i = (b_1 \times x_{i1}) + (b_2 \times x_{i2}) + \cdots + (b_M \times x_{iM}) + b_0$$

$$p(y_i = 1|x_i) = \sigma(z_i) = \frac{\exp(z_i)}{1+\exp(z_i)}$$

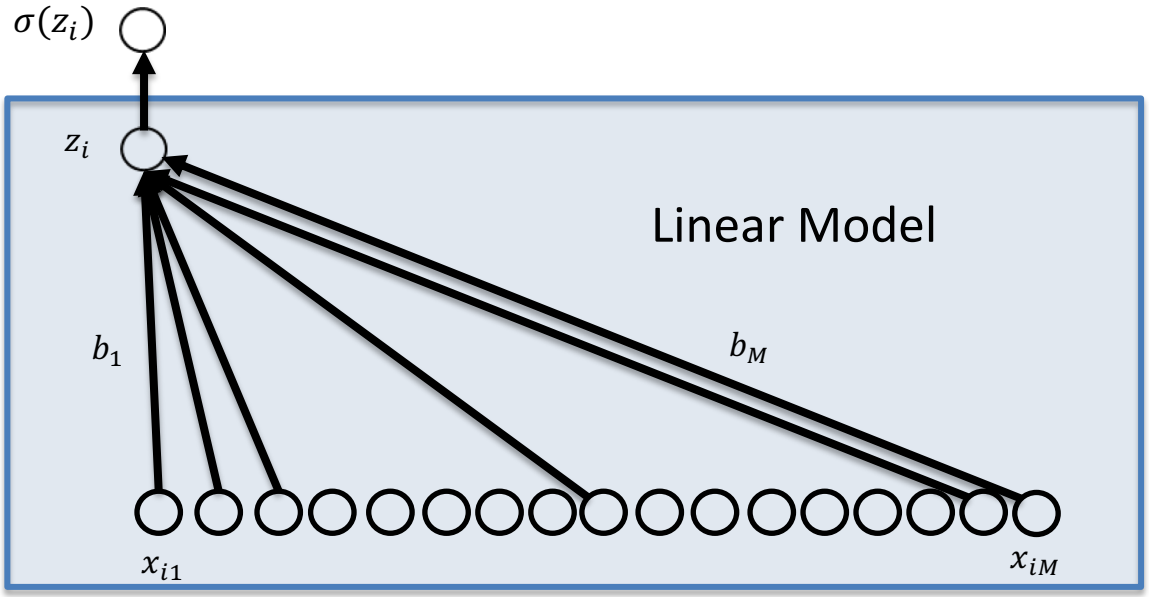



- Large and positive $z_i$ indicates that event $y_i = 1$ is likely
- Large and negative $z_i$ indicates that event $y_i = 0$ is likely

# Logistic Regression

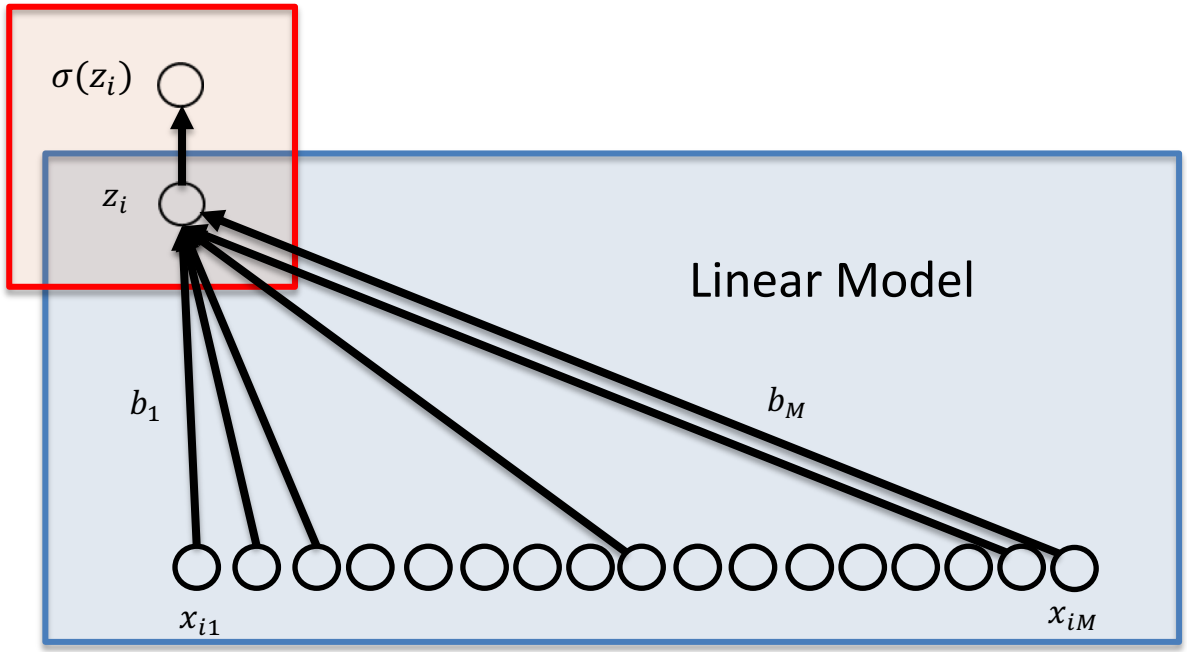


# Logistic Regression



# Logistic Regression

Convert to Probability

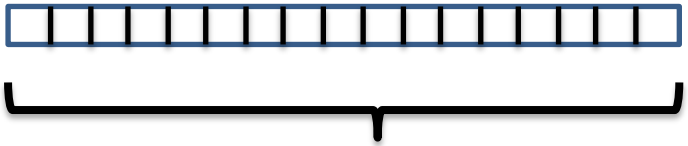


What do the parameters and model mean?

# AN EXAMPLE

# Example

- Outcome:
  - $y_i = 1$, it rains on day $i$;
  - $y_i = 0$, it does not rain on day $i$
- Features:
  - On day $i$ what is the $\{cloud\ cover, humidity, temperature, air\ pressure, \ldots\}$



$x_i$, features for day $i$

$y_i$, did it rain on day $i$

# Example

- Outcome: $y_i = 1$, it rains on day $i$; $y_i = 0$, it does not rain on day $i$
- Features: On day $i$ what is the $\{1: cloud\ cover, 2: humidity, 3: temperature, \dots\}$

$$z_i = (b_1 \times x_{i1}) + (b_2 \times x_{i2}) + \cdots + (b_M \times x_{iM}) + b_0$$

Cloud Cover  Humidity

- If cloud cover is positively related to rainfall, $b_1$ should be positive

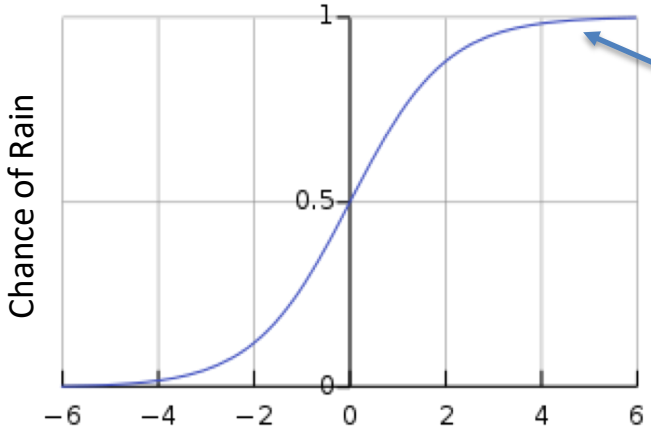
# Impact on the Sigmoid Function

$$z_i = (b_1 \times x_{i1}) + (b_2 \times x_{i2}) + \cdots + (b_M \times x_{iM}) + b_0$$

Cloud Cover

Humidity

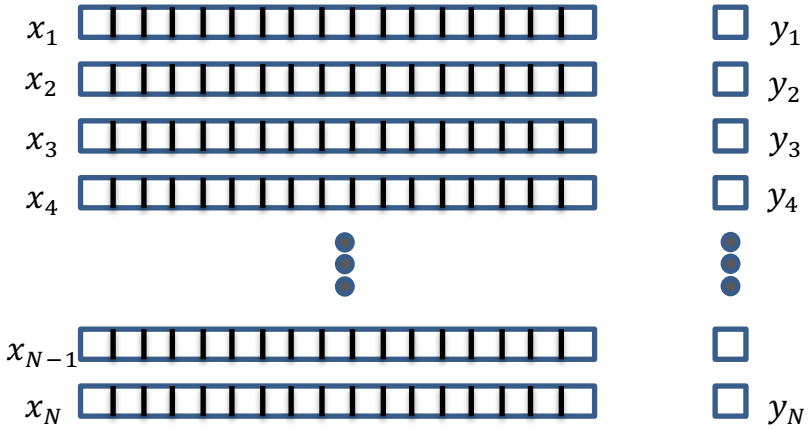
$$p(y_i = 1|x_i) = \sigma(z_i)$$



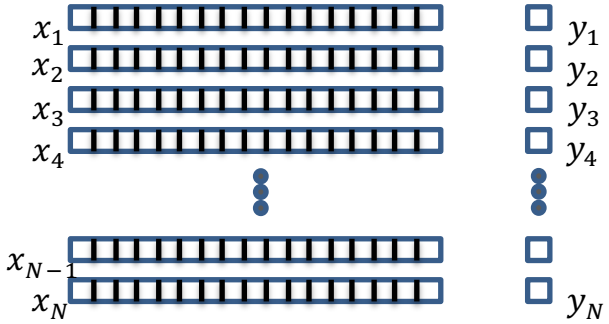

As the value $z_i$ increases, the chance of rain increases

# Building the Training Set

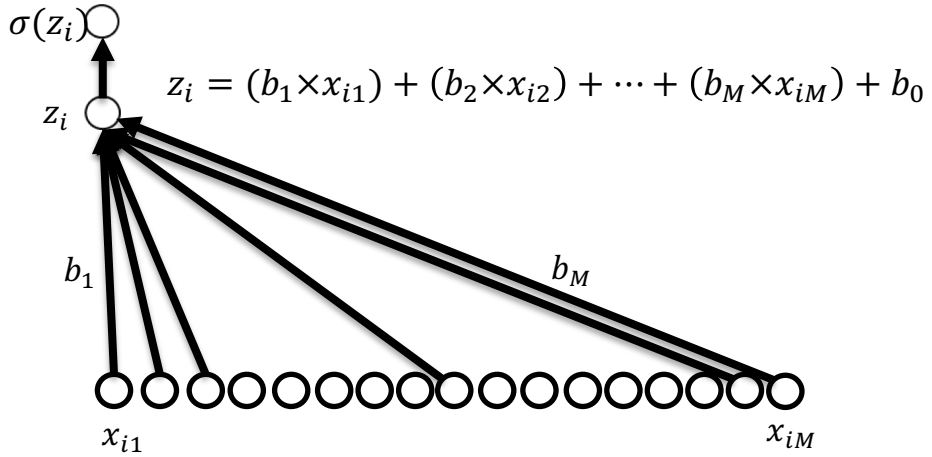
- Need to learn the parameters
- Requires *training data*
- Record data from $N$ days
  - Capture features: $\{cloud\ cover, humidity, temperature, \ldots\}$
  - Did it rain?



# Learning Model Parameters




Training Set

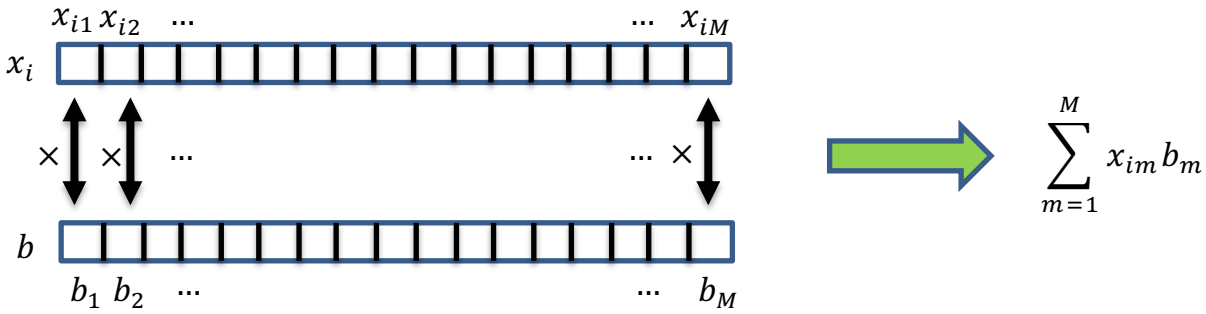



Logistic Regression Model (or "Network")

Learned Parameters $(b_0, b_1, \ldots b_N)$

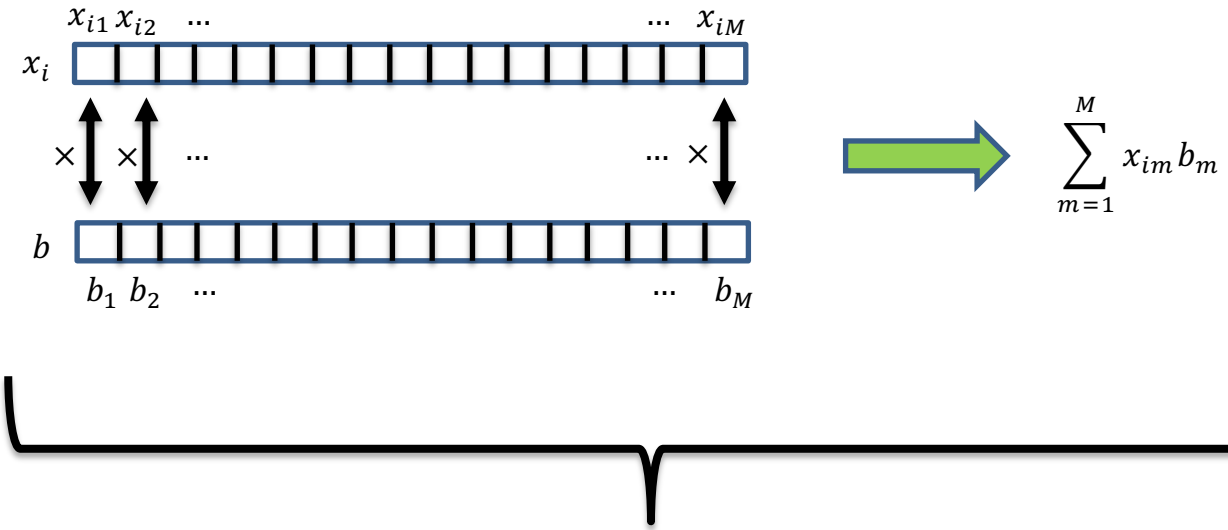
# Interpretation of Logistic Regression

$$z_i = (b_1 \times x_{i1}) + (b_2 \times x_{i2}) + \cdots + (b_M \times x_{iM})$$



# Interpretation of Logistic Regression

$$z_i = (b_1 \times x_{i1}) + (b_2 \times x_{i2}) + \cdots + (b_M \times x_{iM})$$

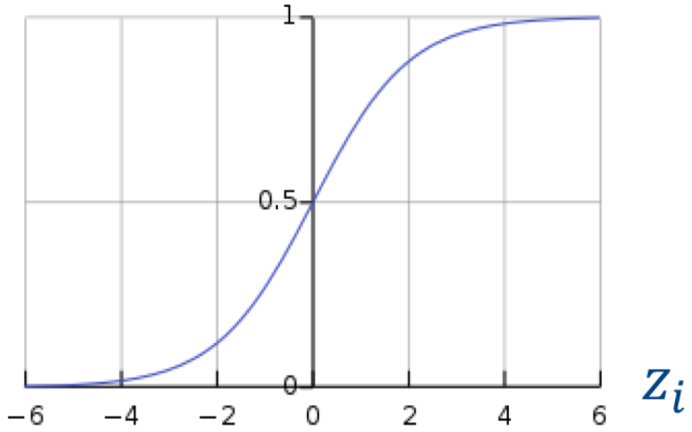


# Interpretation of Logistic Regression

$$z_i = b_0 + (b_1 \times x_{i1}) + (b_2 \times x_{i2}) + \cdots + (b_M \times x_{iM})$$

$$= b_0 + x_i \odot b$$

$$p(y_i = 1 | x_i) = \sigma(z_i)$$

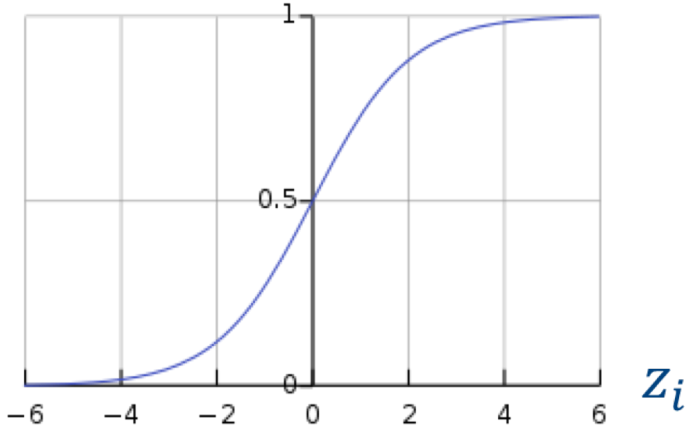


# Interpretation of Logistic Regression

$$z_i = b_0 + (b_1 \times x_{i1}) + (b_2 \times x_{i2}) + \cdots + (b_M \times x_{iM})$$

$$= b_0 + x_i \odot b$$

$$p(y_i = 1|x_i) = \sigma(z_i)$$



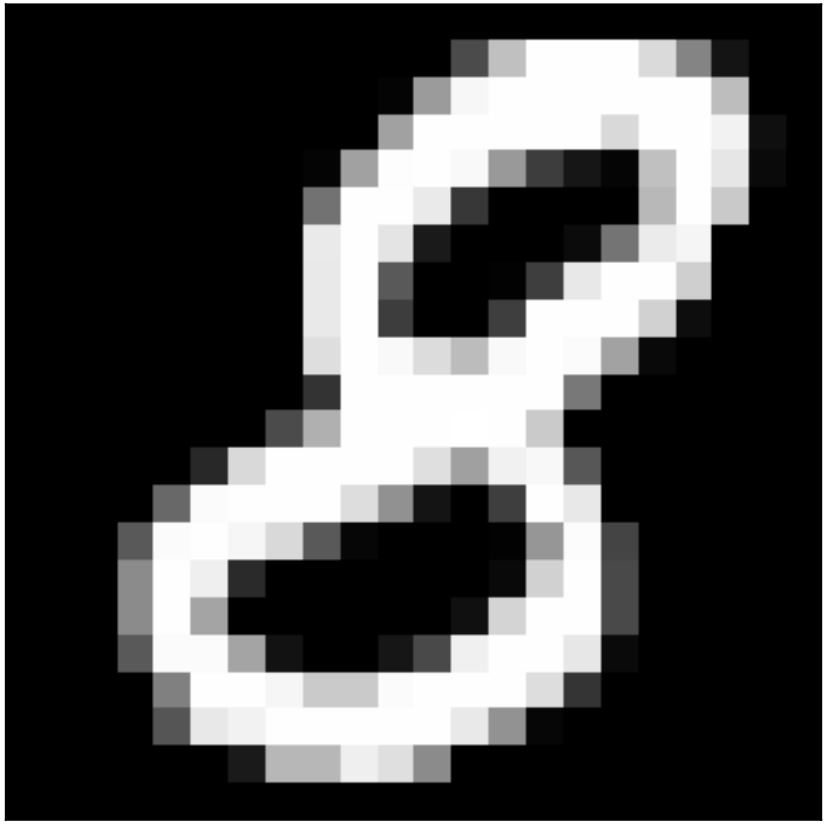

- ❑ May think of vector $b$ as a template or filter (will visualize to make clear)
- ❑ If $x_i$ is aligned/matched with $b$, then $x_i \odot b$ will be large
- ❑ The parameter $b_0$ is a bias to correct for class prevalences

# RECOGNIZING NUMERICAL DIGITS

# The MNIST Dataset

- The Modified National Institute of Standards and Technology (MNIST) contains pictures of handwritten digits (0,1,2,…)
- Want to be able to tell what digit each image is (e.g. optical character recognition)

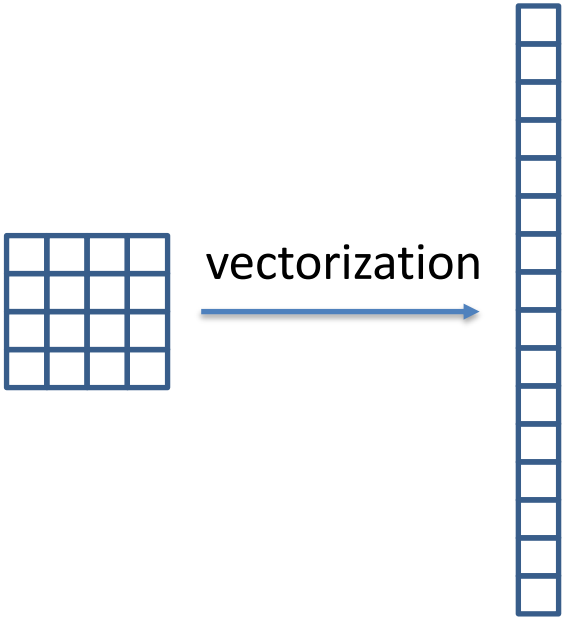
# Images are Encoded as Numbers



| | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 118 | 200 | 223 | 155 | 155 | 23 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 6 | 124 | 214 | 253 | 253 | 253 | 253 | 254 | 229 | 213 | 67 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 43 | 198 | 253 | 254 | 253 | 247 | 175 | 175 | 236 | 253 | 253 | 108 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 9 | 212 | 253 | 253 | 224 | 58 | 19 | 0 | 0 | 16 | 139 | 247 | 247 | 71 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 58 | 254 | 254 | 186 | 14 | 0 | 0 | 0 | 0 | 0 | 0 | 139 | 254 | 118 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 58 | 253 | 183 | 8 | 0 | 0 | 0 | 0 | 0 | 0 | 36 | 222 | 253 | 118 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 54 | 250 | 128 | 2 | 0 | 0 | 0 | 6 | 71 | 192 | 237 | 253 | 247 | 71 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 213 | 253 | 50 | 0 | 18 | 123 | 198 | 253 | 254 | 253 | 247 | 85 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 69 | 241 | 227 | 136 | 200 | 253 | 253 | 253 | 254 | 192 | 34 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 173 | 254 | 256 | 254 | 254 | 254 | 149 | 59 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 101 | 253 | 253 | 254 | 253 | 253 | 253 | 42 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 8 | 138 | 247 | 253 | 243 | 159 | 196 | 243 | 253 | 199 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 12 | 183 | 253 | 253 | 253 | 50 | 0 | 0 | 49 | 253 | 214 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 180 | 254 | 253 | 213 | 50 | 2 | 0 | 0 | 71 | 253 | 214 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 234 | 217 | 97 | 10 | 0 | 0 | 0 | 23 | 207 | 254 | 215 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 174 | 253 | 156 | 0 | 0 | 0 | 0 | 45 | 215 | 253 | 253 | 95 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 9 | 210 | 253 | 163 | 5 | 19 | 49 | 130 | 244 | 253 | 251 | 137 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 13 | 229 | 253 | 254 | 192 | 253 | 253 | 253 | 254 | 253 | 137 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 160 | 253 | 254 | 253 | 253 | 253 | 253 | 239 | 132 | 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 18 | 112 | 194 | 254 | 254 | 254 | 163 | 59 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

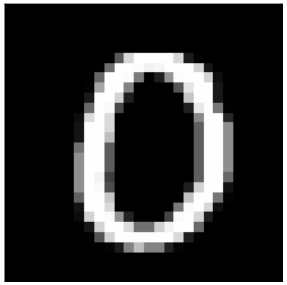
# Vectorization

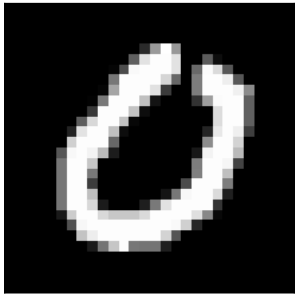
- We will start talking about deep learning *without* using the structure of the image
- We will return to this in a future lecture
- To convert an image into an unstructured set of numbers, we *vectorize it*

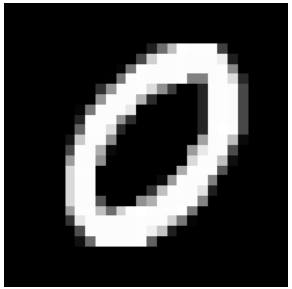


# Start With The Binary Case

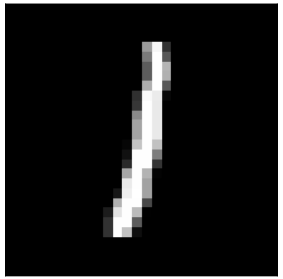
Zeros

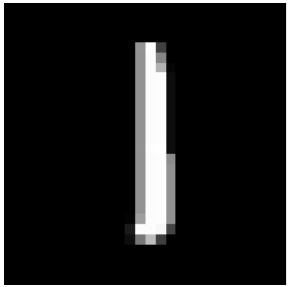


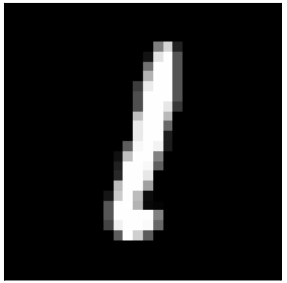




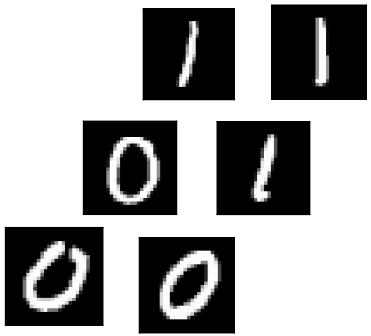
Ones



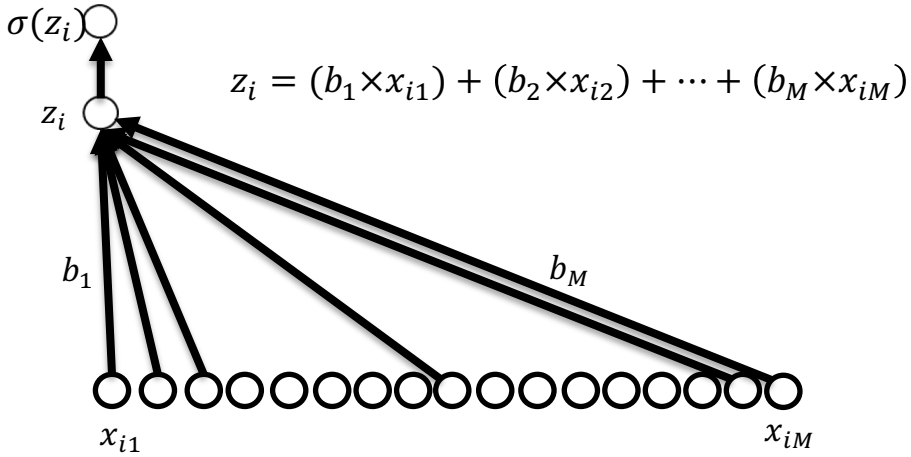




# Learning on MNIST



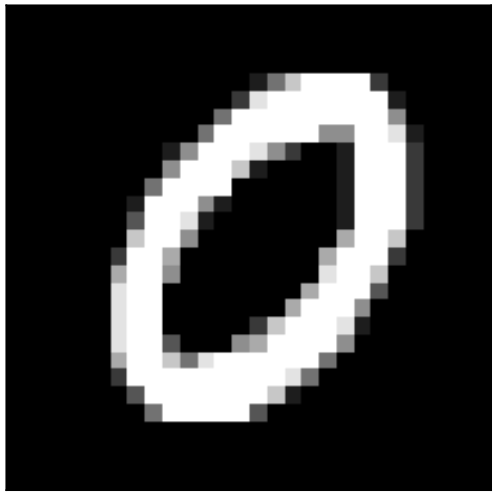
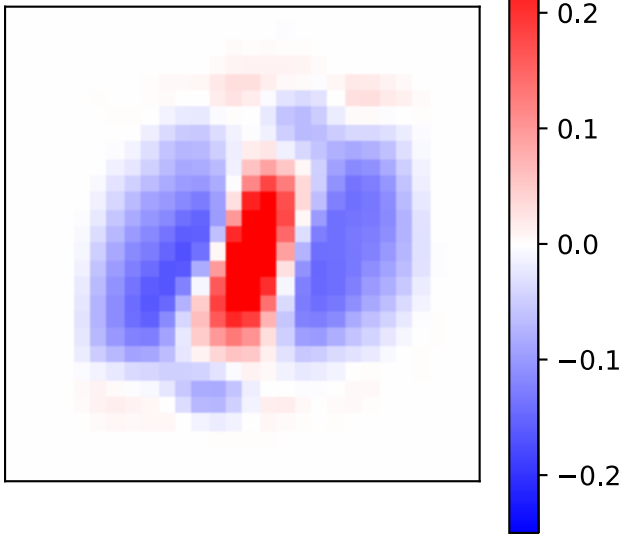
Vectorization




Logistic Regression Model (or "Network")

Learned Parameters $(b_0, b_1, \ldots b_N)$

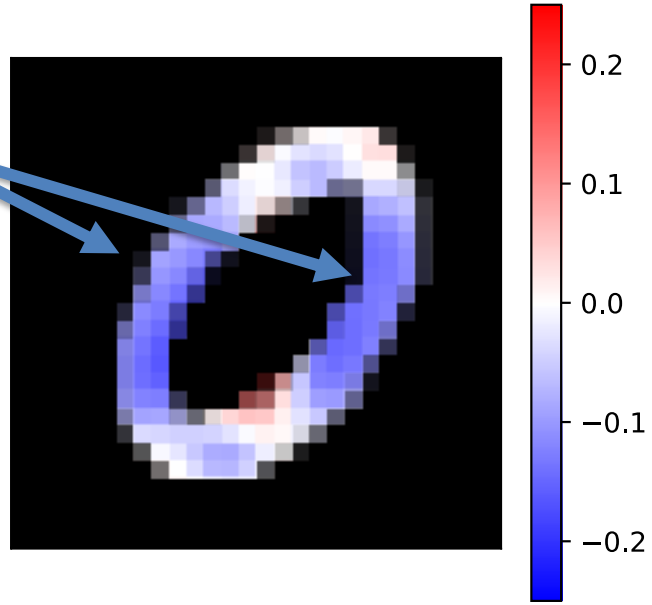
# Zooming in on 0/1

$\sigma($  $\times$ 

$)$

# Zooming in on 0/1

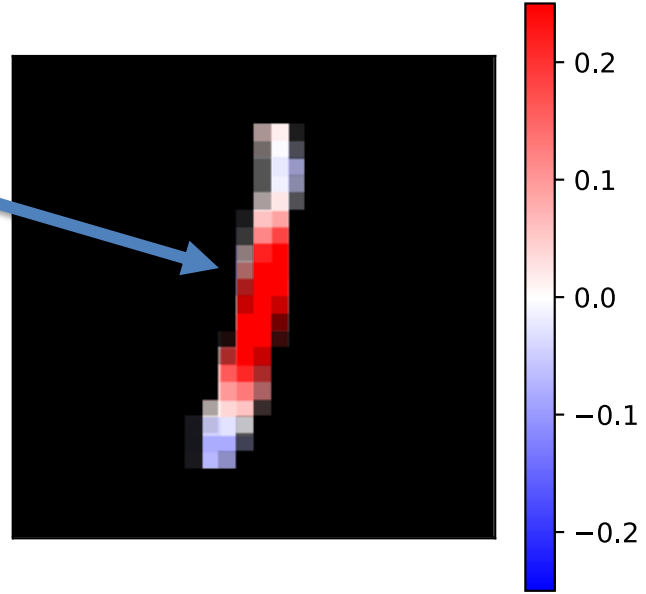
Negative Sections




$$\sigma(\quad) = 0.006$$

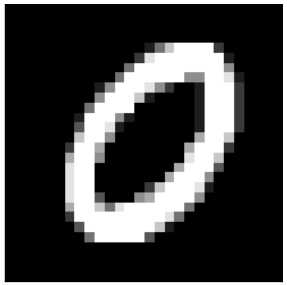
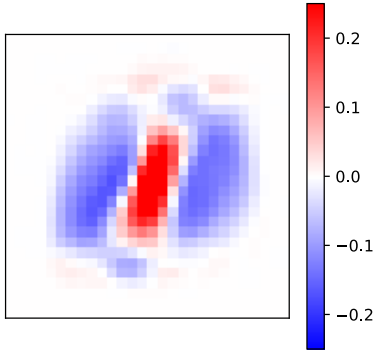
# Zooming in on 0/1

Positive Section



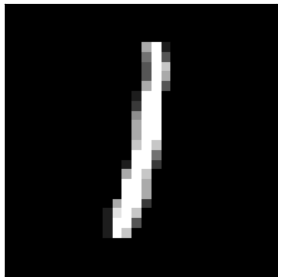
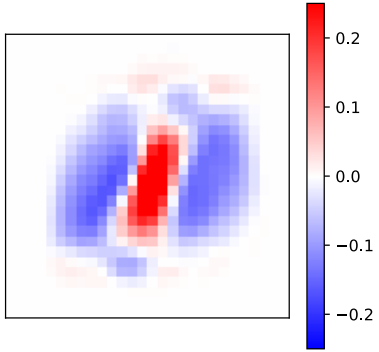

$$\sigma(\quad) = .991$$

# Learned Weights for 0/1

$\sigma($  $\times$ 

$) = .006$

We think that this is a “zero” (.6% chance it is a “one”)

$\sigma($  $\times$ 

$) = .991$

We think that this is a “one” (99.1% chance it is a “one”)

# How do we determine the parameters here?

- Same as before, we're going to use the log-likelihood
- This time, we're going to use the Bernoulli (e.g. binary, binomial)

$$p(y = 1; r) = r, p(y = 0, r) = (1 - r)$$

- This is the same as a weighted coin flip

# How do we determine the parameters here?

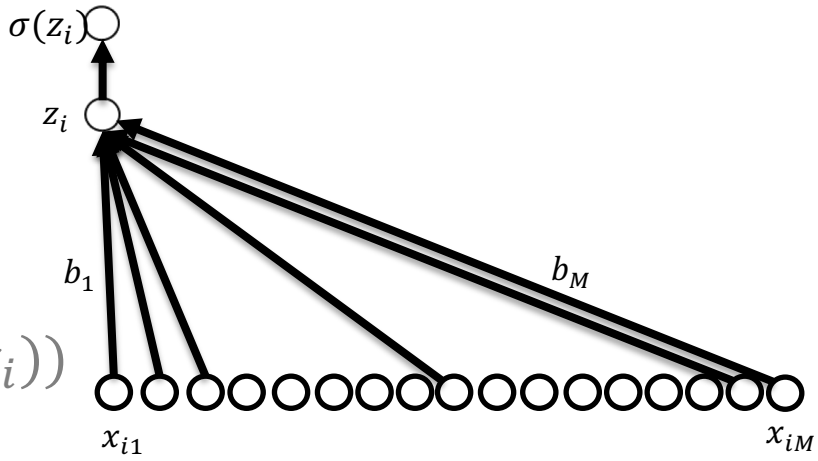
- In logistic regression, we have

$$p(y = 1; r) = r,$$
$$p(y = 0, r) = (1 - r)$$
$$r = \sigma(b_0 + x_i \odot b)$$

- Maximize:

$$b_0^*, \boldsymbol{b}^*$$

$$= \arg \max_{b_0, \boldsymbol{b}, z_i = b_0 + x_i \odot b} \sum_{i=1}^{N} \log p(y_i; \sigma(z_i))$$



# “Empirical Risk Minimization”

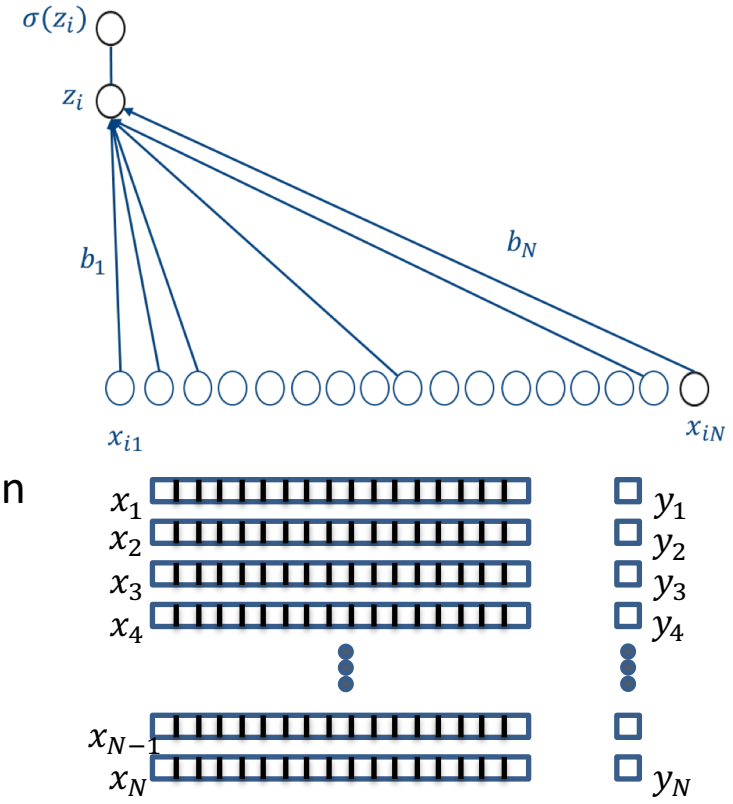
- A *loss function* $\ell(true, prediction)$ defines a penalty for poor predictions
- Want to minimize average loss
- Mathematically, this can be stated as

$$\boldsymbol{b}^* = \arg\min_{\boldsymbol{b}} \frac{1}{N} \sum_{i}^{N} \ell(y_i, \sigma(z_i))$$

Loss function

Guess

True Label



# What is the Loss Function?

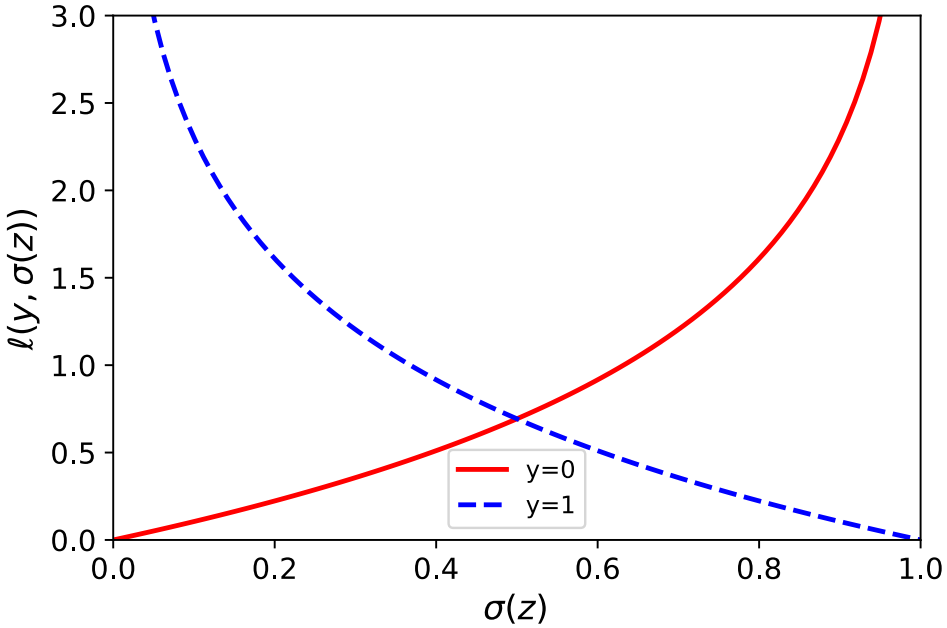
- Define $\sigma(z_i)$ as the predicted probability
- $y_i$ is our true label
- Can be viewed as the negative log-likelihood
  $\ell\big(y_i, \sigma(z_i)\big) = -\log p(y_i|\sigma(z_i))$
- Specific mathematical form is:
  $\ell\big(y, \sigma(z)\big) = -y \log \sigma(z) - (1-y) \log\big(1-\sigma(z)\big)$

# Visualization of Logistic Loss Function

- The logistic/cross-entropy loss is:

$$\ell\big(y, \sigma(z)\big) = -y \log \sigma(z) - (1-y) \log\big(1 - \sigma(z)\big)$$

- Dashed blue line shows loss when the true prediction is positive
- When we give 100% of a 1, we pay no penalty
- If we are less confident, we pay a small penalty
- If we are overconfident and wrong, we pay a large penalty




Predicted Probability of a "one"

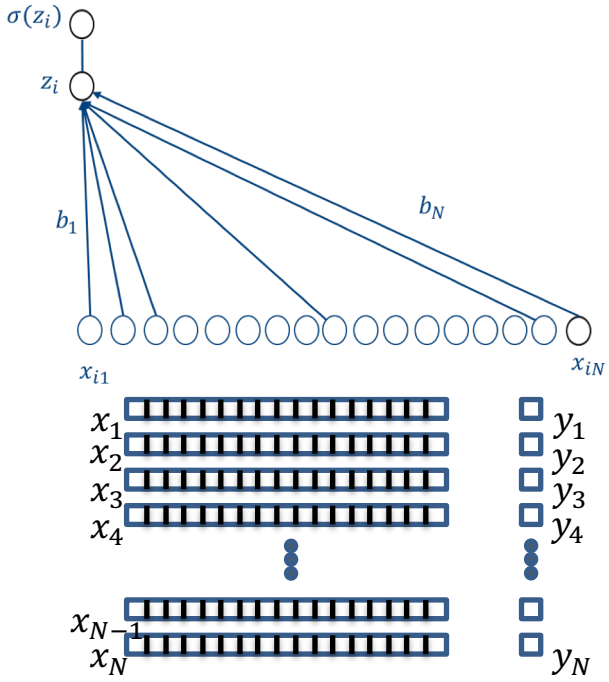
# Optimization Goal for Binary Classification

- The optimization goal is to minimize the average loss

$$\boldsymbol{b}^* = \arg\min_{\boldsymbol{b}} \frac{1}{N} \sum_{i}^{N} \ell(y_i, \sigma(z_i))$$

- For binary (0/1) problems, the logistic or cross-entropy loss is

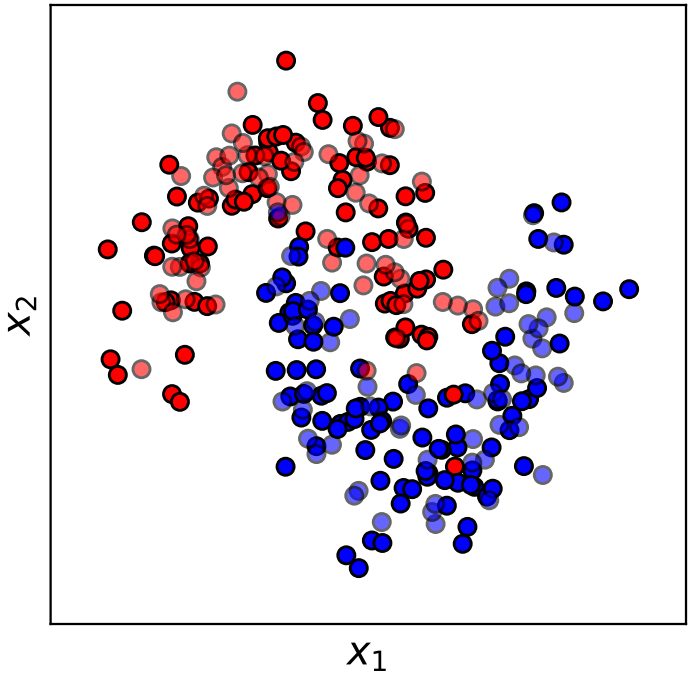
$$\ell\big(y, \sigma(z)\big) = -y \log \sigma(z) - (1-y) \log\big(1 - \sigma(z)\big)$$



# LIMITATIONS, LOOKING FORWARD

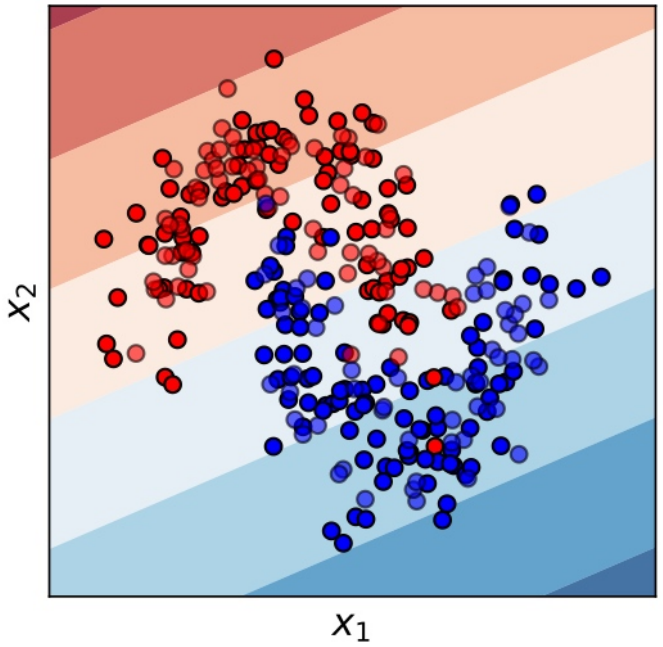
# Logistic Regression is a “Linear” Classifier

- Also referred to as a generalized linear model
- Can only split data by linear trends



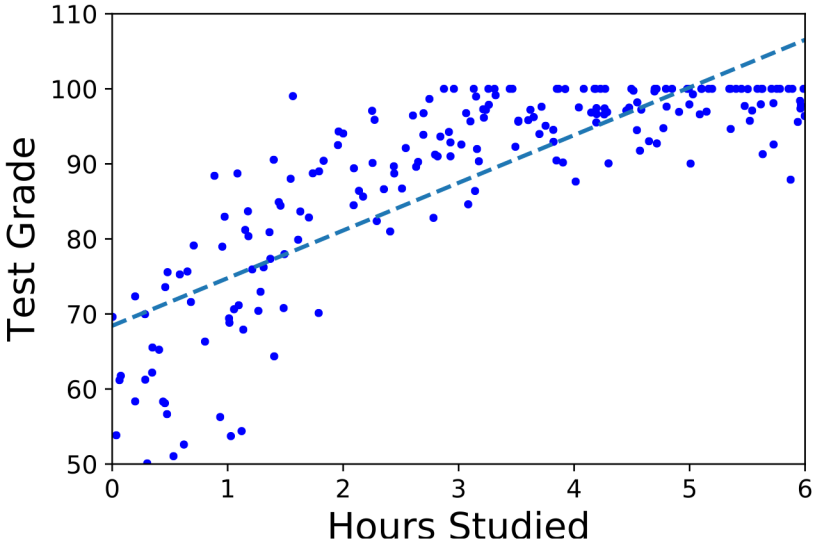
# Limitations of Linear Classifiers

- Linear classifiers can only represent limited relationships
- Often want to use a classifier that can handle non-linearities.
- Many different ways of creating non-linear classifiers!



# Can we create a non-linear model for test scores?

- Want to capture the effect of “diminishing returns” in the data
- Can we construct automatic ways of “learning” this?



# Looking Forward

- These are two initial techniques for making predictions from data
- Will start talking about machine learning algorithms for making sense of the data