SIGNLL Meeting 2

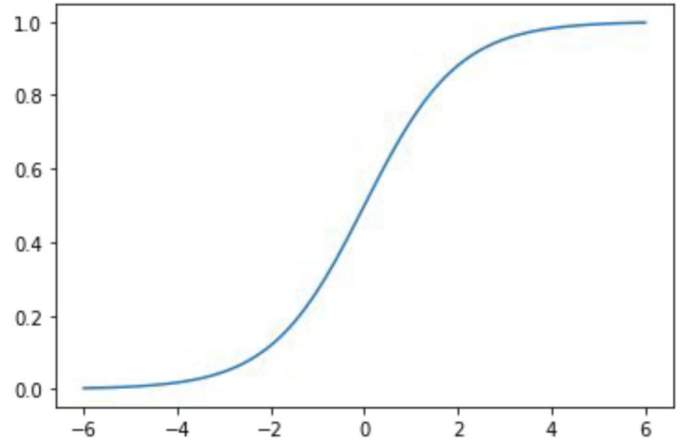
Agenda (9/18)

- Logistic Regression
- Discuss/finalize project ideas

What is Logistic Regression?

- GPT/Naive Bayes are **generative**
- Logistic Regression is **discriminative**
- For an input x, apply a weight and a bias -> wx + b
- Apply the sigmoid function to this to get a probability (between 0 and 1)
- Sigmoid function:

$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$



What about multiple inputs?

- Suppose you have a vector input **x**. Then instead of multiplying it by a scalar weight w, you multiply it by a vector of weights **W** (dot product)**.** Add the bias to the dot product as normal.

$$z = \mathbf{w} \cdot \mathbf{x} + b$$

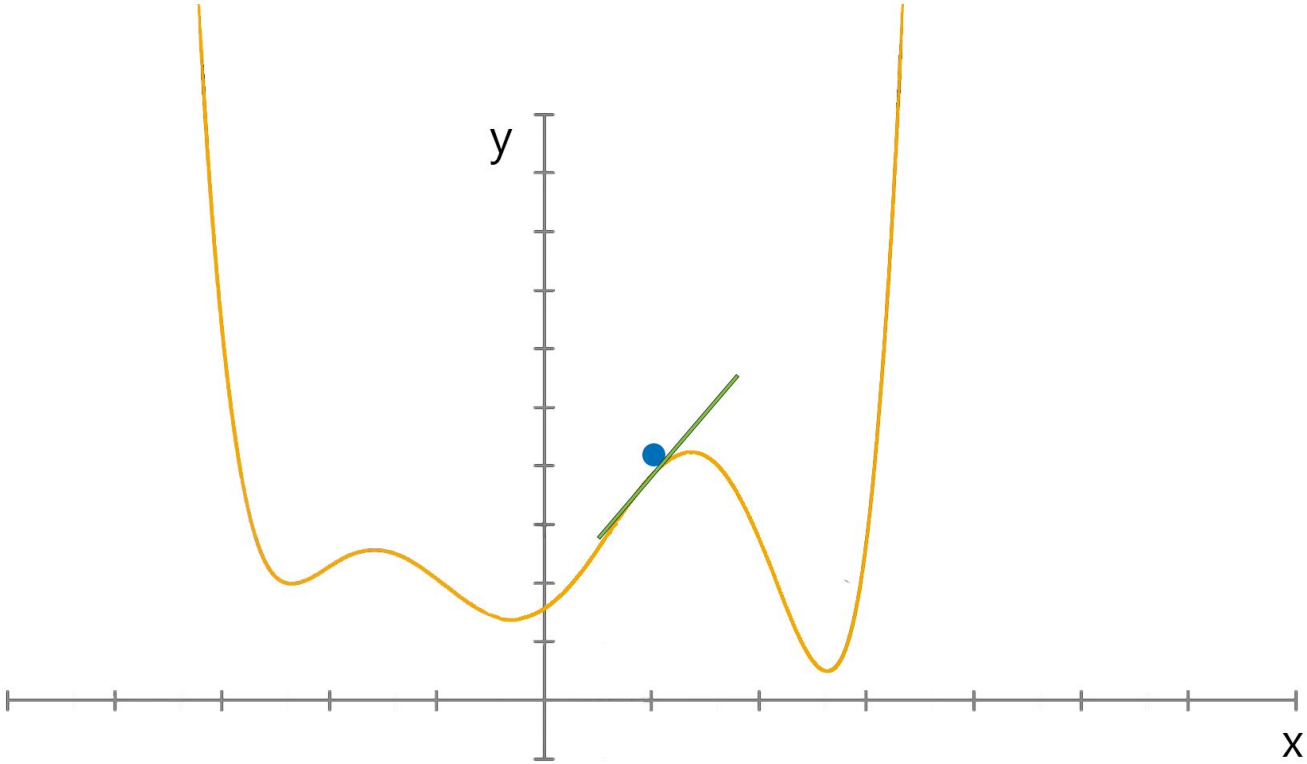
- Apply the sigmoid function to this value
- Make a decision boundary
  - E.g. If the value is > 0.5, it predicts one class. If not, it predicts another.

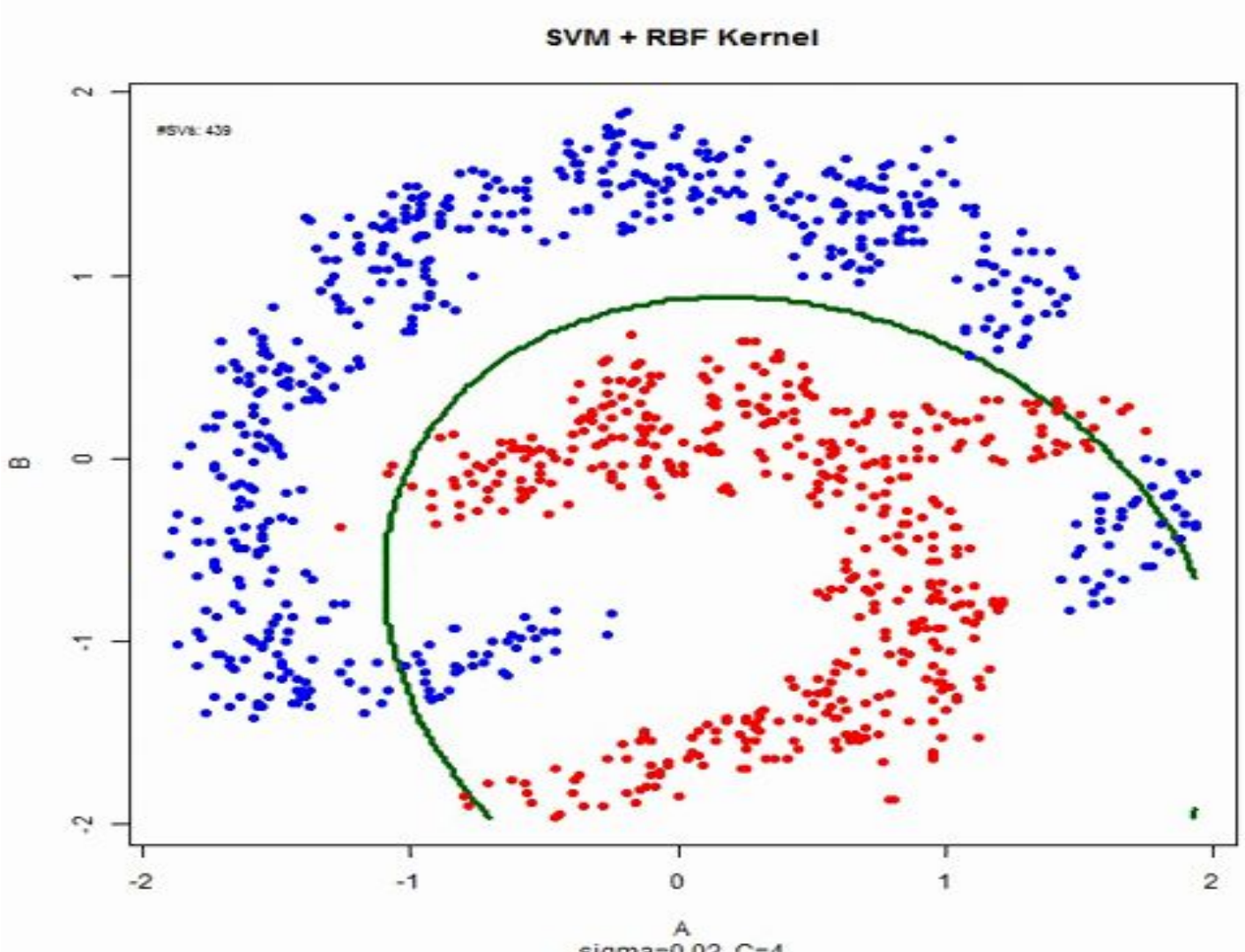
How It Is Used

- Fast and Uncomplicated Classifier
  - Mainly used for simple classification problems
- Similar to Linear Regression except CATEGORICAL
  - I.e. yes/no, true/false, etc.

Introducing Gradient Descent

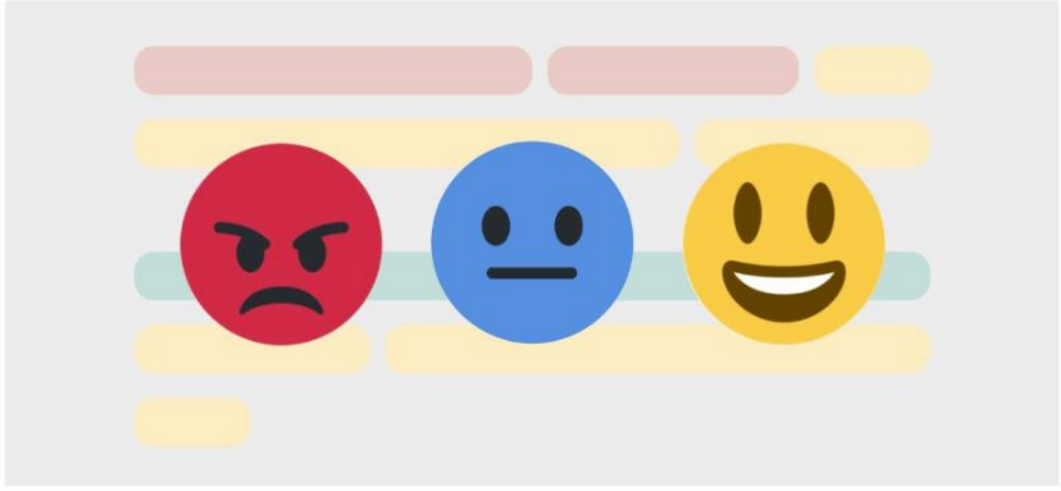
- Cost: How far is the actual value from the predicted value?
- Cost Function: A function that calculates the cost of our model
- Gradient Descent: A way to minimize the cost function on training data


y
x

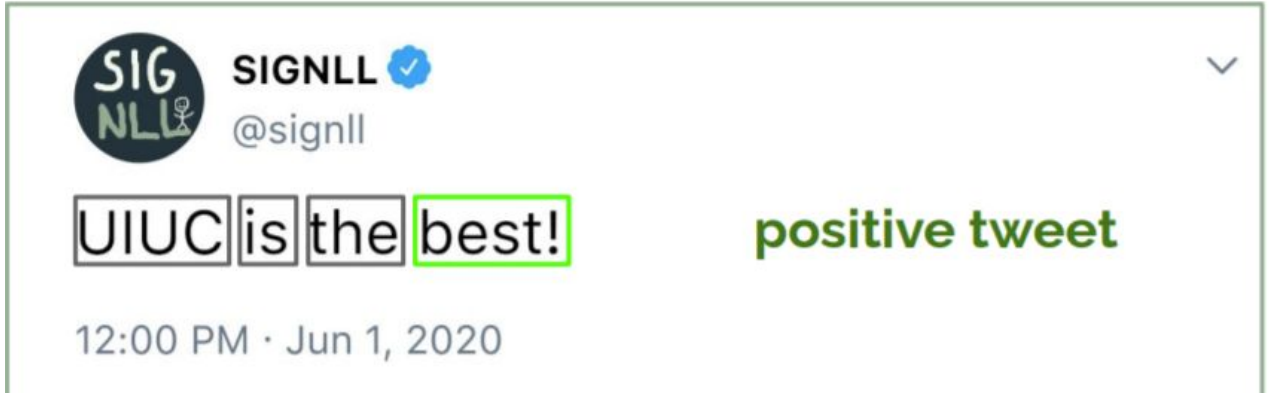

SVM + RBF Kernel
#SVs: 439
B
A
2
1
0
-1
-2
-2
-1
0
1
2

Sentiment Analysis

Sentiment analysis is the interpretation and classification of emotions (positive, negative and neutral) within text data using text analysis techniques



Classifying Words



Sentiment Analysis Steps

1) Process and clean up our tweets

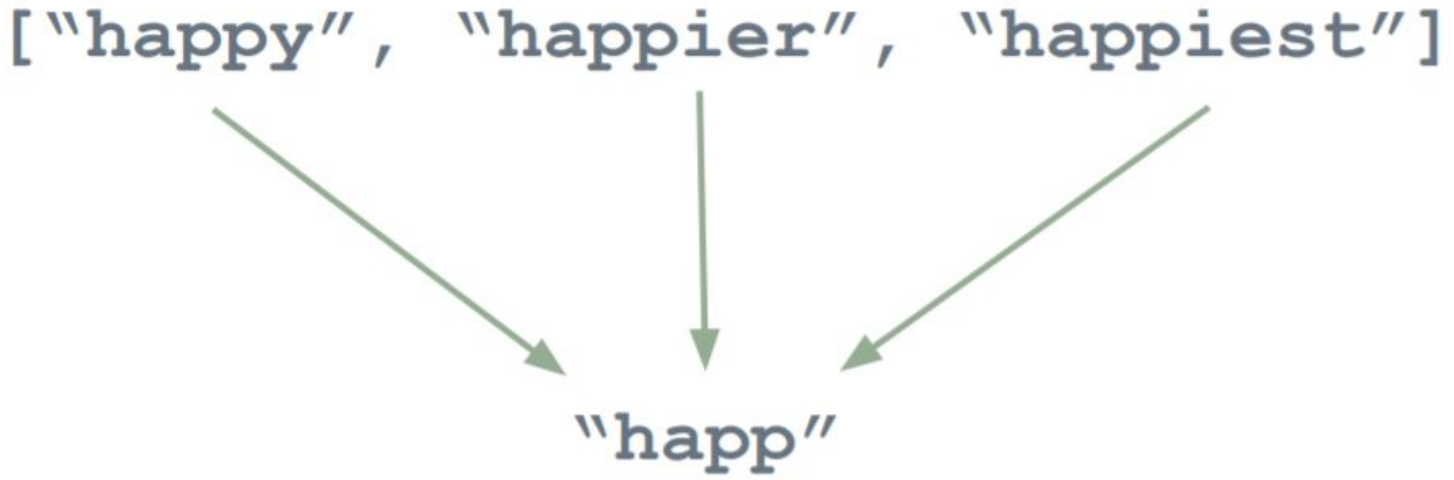
2) Organize them into a dictionary

3) Create and train a model using logistic regression

4) Verify our results with test data

Word Tokenization

- How do we deal with multiple word definitions?
- Fortunately, we can run a built-in algorithm to make sure that these words all mean the same thing



Stopwords

There are also some words that add no meaning to the sentence, such as:

[“the”, “a”, “an”, “I”, ...]

Since these words don’t help us that much, we can remove them from our tweets

Dictionary Conversion

Instead of having separate columns for positive tweet count and negative tweet count, we can represent them as a single list of length 2:

$$word : \{n_{neg} \ n_{pos}\}$$

Representing our tweets as numbers

To use logistic regression, we need to represent our tweets as numbers

We can do this by using three parameters from our dictionary:

$$\{1 \sum count_{neg} \sum count_{pos}\}$$

Tokens: ["wish", "friend", "like"]

```
  "wish" : [63, 29]
+
  "friend" : [30, 40]
+
  "like" : [182, 187]
_____________________
```

tweet_val = [1, 275, 256]

Logistic Gradient Descent

$$y = \begin{Bmatrix} 1 & neg_0 & pos_0 \\ 1 & neg_1 & pos_1 \\ \dots & \dots & \dots \\ 1 & neg_{m-1} & pos_{m-1} \end{Bmatrix} \cdot \{\Theta_0 \ \Theta_1 \ \Theta_2\}$$

$$y = sigmoid(y)$$

GitHub!

- Step 1:
  - Install Python (https://www.python.org/downloads/) and pip
  - Run the following command: pip install notebook
- Step 2:
  - Clone this semester's GitHub repository to your machine
    - https://github.com/SIGNLL-UIUC/SIGNLL-Fall-2022
  - Navigate to the "Workshops" folder and run "jupyter notebook"
    - Select the .ipynb file to edit
  - Happy logistical regressioning!

Brainstorm!

- Text summarizer
- Calendar appointment generator
- Topic autocomplete feature