CSE 156 Final Project

Yaobang Deng
A13712124

Zhongyu Chen
A13801210

Rongze Yuan
A13998006

Abstract

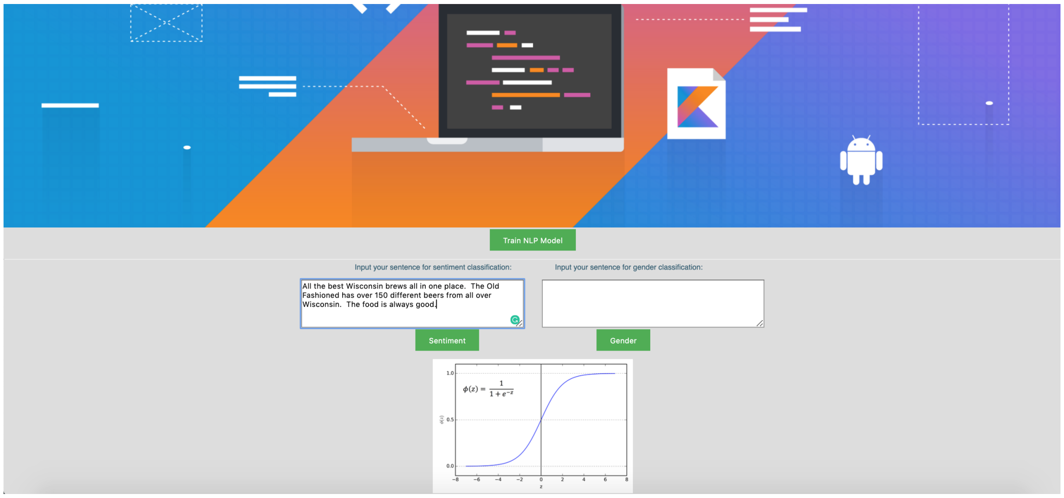
In this final project, we use several ways attempting to explain the prediction generated from our text classifier. The methods include LIME explanation, L1-norm weights extraction, and L2-norm weights extraction.

1 Part1

1.1

In Figure1, there are two input boxes. For sentiment analysis, user can input a sentence in the box corresponding to the button sentiment, and click the button 'sentiment' to generate result.

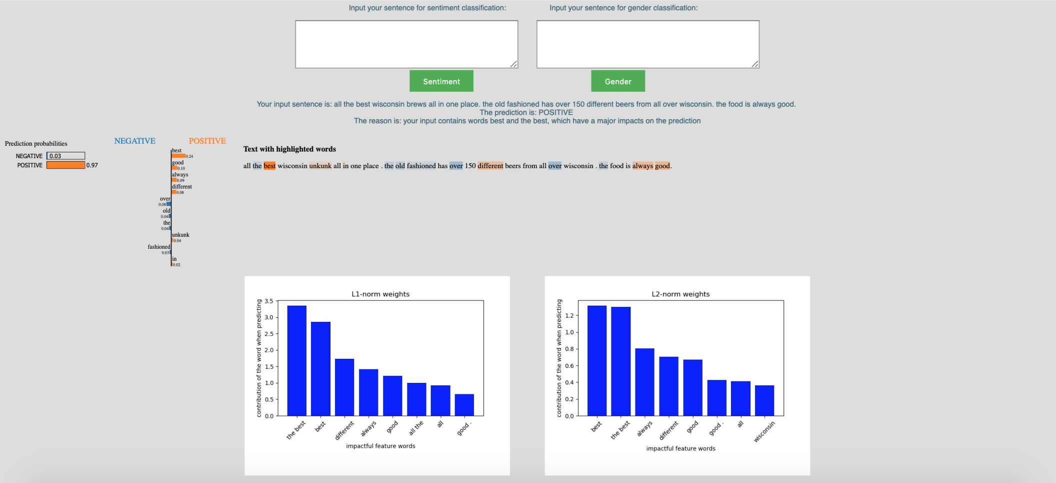
Figure 1: Input our favourite example


Train NLP Model
Input your sentence for sentiment classification:
Input your sentence for gender classification:
Sentiment
Gender
$\phi(z) = \frac{1}{1+e^{-z}}$
1.0
0.5
0.0
$\phi(z)$
−8
−6
−4
−2
0
2
4
6
8
z

All the best Wisconsin brews all in one place. The Old Fashioned has over 150 different beers from all over Wisconsin. The food is always good.

In Figure2, we generate three different results. The first one is LIME explanation(the graph with orange and blue). The second and third one are top 8 highest weights with their features from our model with different regularization(the values shown are top 8 $w_i * x_i$).

Figure 2: Result of favourite example



Input your sentence for sentiment classification:

Input your sentence for gender classification:

Your input sentence is: all the best wisconsin brews all in one place. the old fashioned has over 150 different beers from all over wisconsin. the food is always good.
The prediction is: POSITIVE
The reason is: your input contains words best and the best, which have a major impacts on the prediction

Prediction probabilities

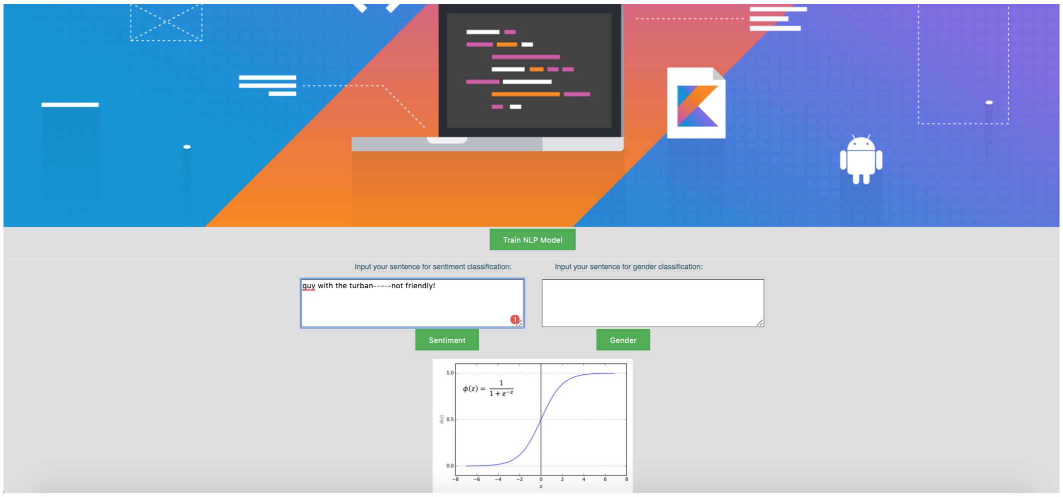
NEGATIVE	0.03
POSITIVE	0.97

Text with highlighted words

all the best wisconsin unkunk all in one place . the old fashioned has over 150 different beers from all over wisconsin . the food is always good.

1.2

Figure 3: Input overconfident example

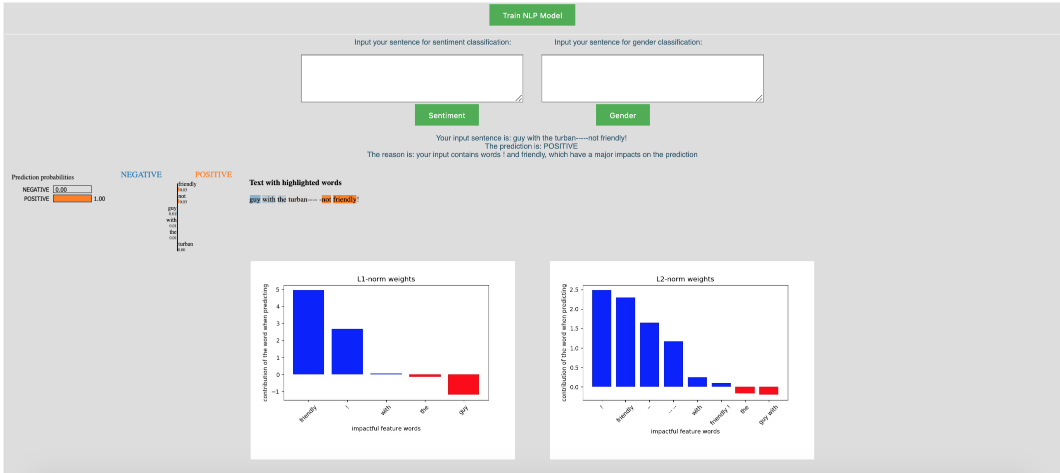


Input your sentence for sentiment classification:

guy with the turban-----not friendly!

Input your sentence for gender classification:

Figure 4: Result of overconfident example



Input your sentence for sentiment classification:

Input your sentence for gender classification:

Your input sentence is: guy with the turban-----not friendly!
The prediction is: POSITIVE
The reason is: your input contains words ! and friendly, which have a major impacts on the prediction

Prediction probabilities

NEGATIVE	0.00
POSITIVE	1.00

Text with highlighted words

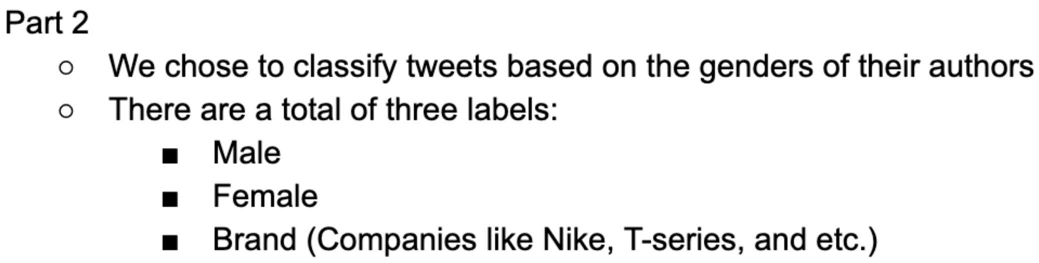
guy with the turban---- -not friendly!

2 Part2

2.1

Description of the new dataset: when user input a sentence/tweet, our classifier will determine it's either inputted by a male, female, or company(brand).

Figure 5: Description of the dataset

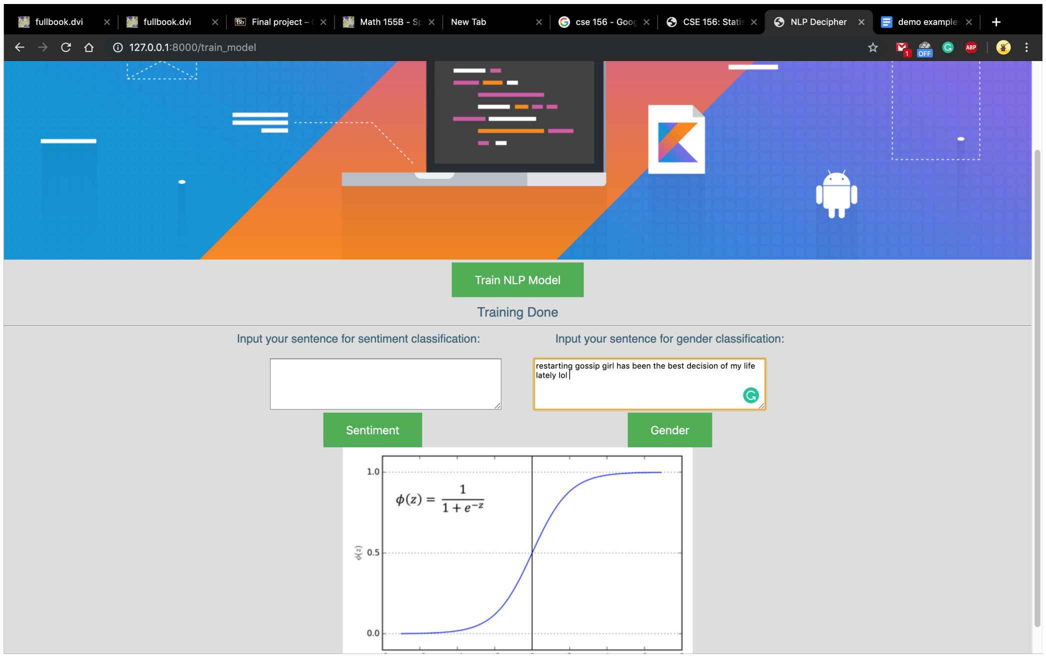


Part 2

- We chose to classify tweets based on the genders of their authors
- There are a total of three labels:
  - Male
  - Female
  - Brand (Companies like Nike, T-series, and etc.)

2.2

Figure 6: Input our favourite example



Training Done

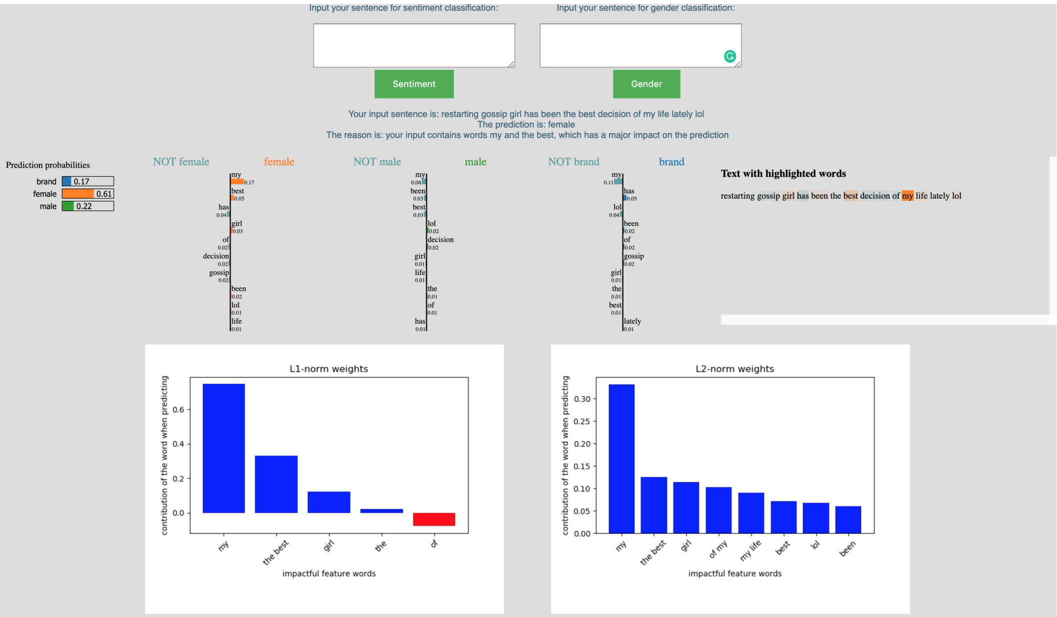
Input your sentence for sentiment classification:

Input your sentence for gender classification:

restarting gossip girl has been the best decision of my life lately lol

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

Figure 7: Result of favourite example



Input your sentence for sentiment classification:

Input your sentence for gender classification:

Your input sentence is: restarting gossip girl has been the best decision of my life lately lol
The prediction is: female
The reason is: your input contains words my and the best, which has a major impact on the prediction

Text with highlighted words

restarting gossip girl has been the best decision of my life lately lol

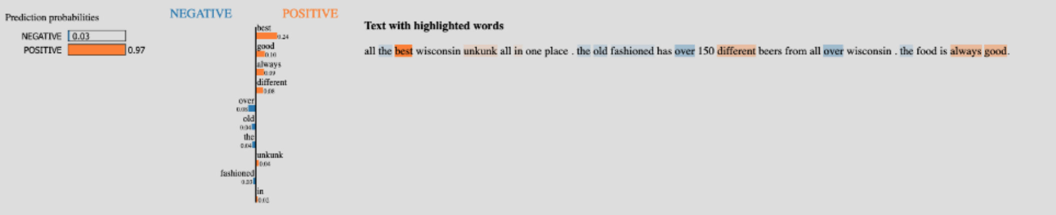
3 Part3

For creativity, we use three different ways to generate/solidify our explanation/prediction of our model.
LIME: it works as follows. For every input sentence, it first generates the probability of the sentence being in one label, then it randomly get rid of a few words in the same input sentence and generates the probability of the modified sentence again. Calculate the difference between two probability and assign that difference as how important the word is in the original sentence. For example, if a sentence contains a few stop-words and the word 'best', the probability of this sentence will be largely affected by the word 'best', in other words, the word 'best' will be assigned a higher difference. Note: LIME only consider uni-gram(single word difference).
L1-norm: as described in equation1, if we apply L1-norm to our logistic regression model, then our model will try to generate a sparse w vector, in other words, the model will pick the words(uni-gram, bi-gram, tri-gram) with most impact in the training corpus. Other words will have weight zero. For instance, if we have stop words and 'best', 'excellent', 'brilliant', then our model will assign non-zero weights to the latter three words, but zeros to the stop words.
L2-norm: as in equation2, the model will generate a w vector with similar magnitude. Then the importance of a word will depend on the Tfidf value of that word. The values shown in the figures from previous part are top 8 $w_i * x_i$.
For L2 norm, we pick the top "highest" $w_i * x_i$(larger for positive and smaller for negative) because, from logistic regression activation function 11, if the label is positive, then larger $w_i * x_i$ contributes a lot to the prediction; if the label is negative, smaller $w_i * x_i$ contributes more.
We used Django to make an interactive website that takes an input sentence and visualizes the important parts of an input that contributed to how the model made a decision. It is interesting because most of the projects we have worked on are not very user-friendly. We usually have to input through jupyter notebook or a terminal, so have a better looking interface is much more user-friendly and easier to understand how the models make decisions. It is good for engineers to debug the model and for beginners to get interested in computer science and natural language processing.

$$||w||_1 = \sum_i |w_i| \tag{1}$$

$$||w||_2 = (\sum_i w_i^2)^{1/2} \tag{2}$$

Figure 8: Example LIME



Text with highlighted words

all the best wisconsin unkunk all in one place . the old fashioned has over 150 different beers from all over wisconsin . the food is always good.

Figure 9: Example L1-norm

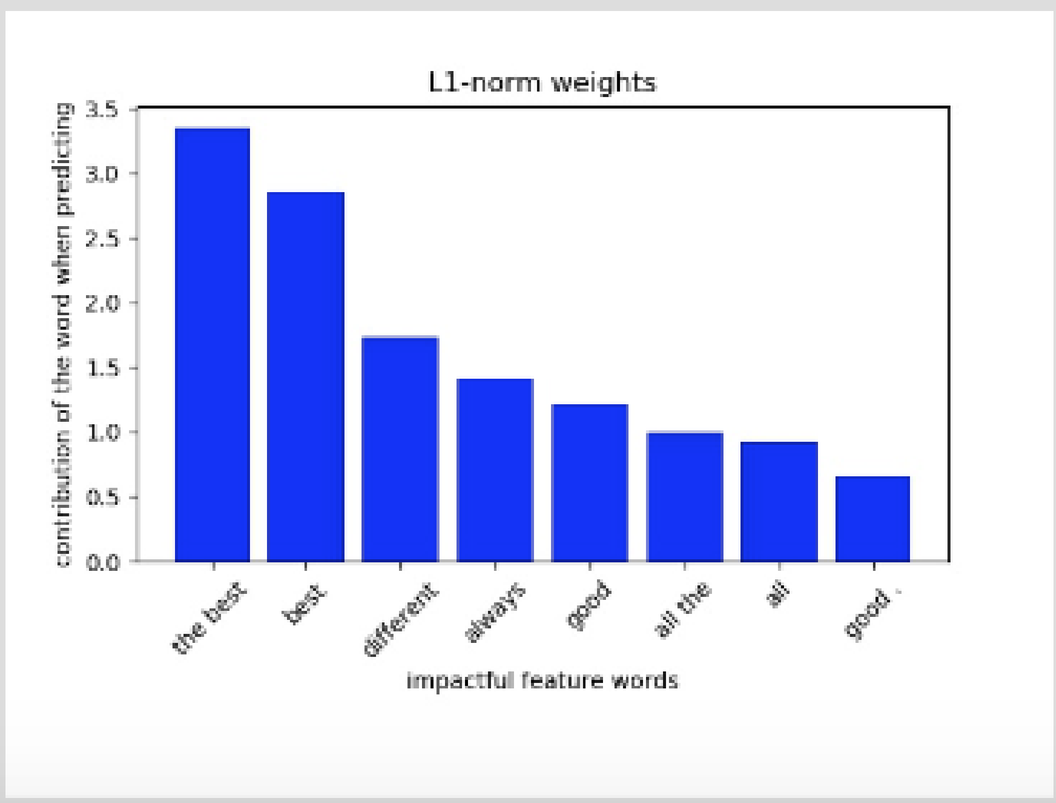


Figure 10: Example L2-norm

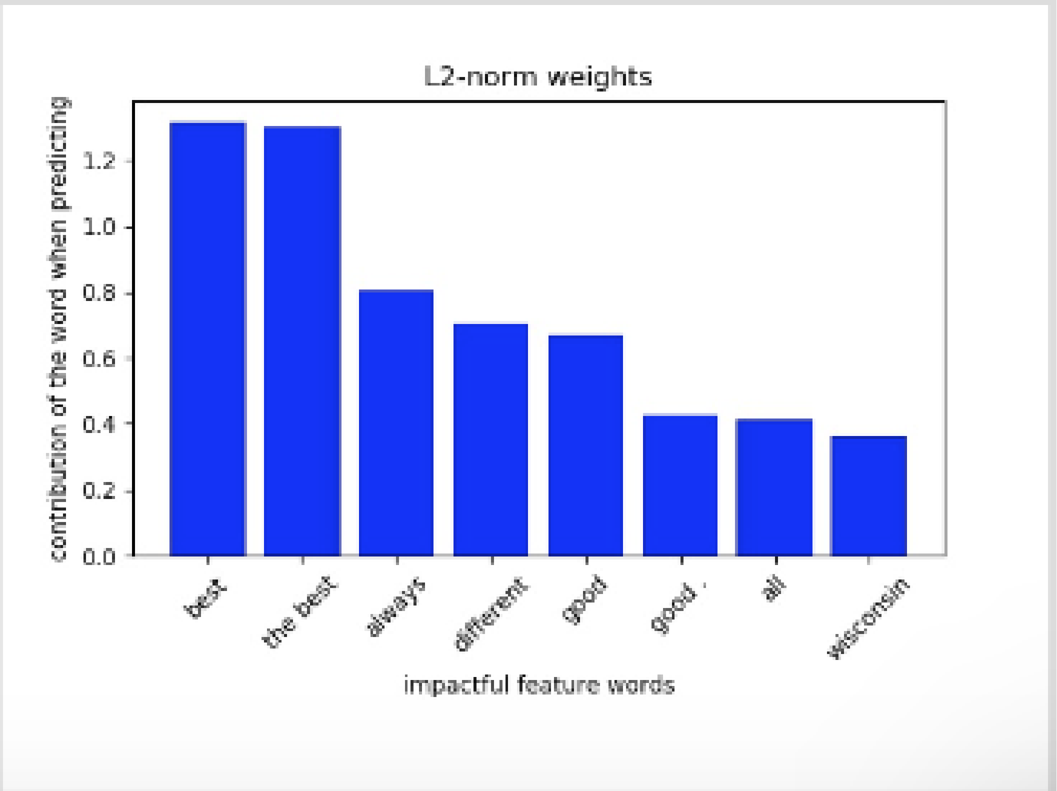
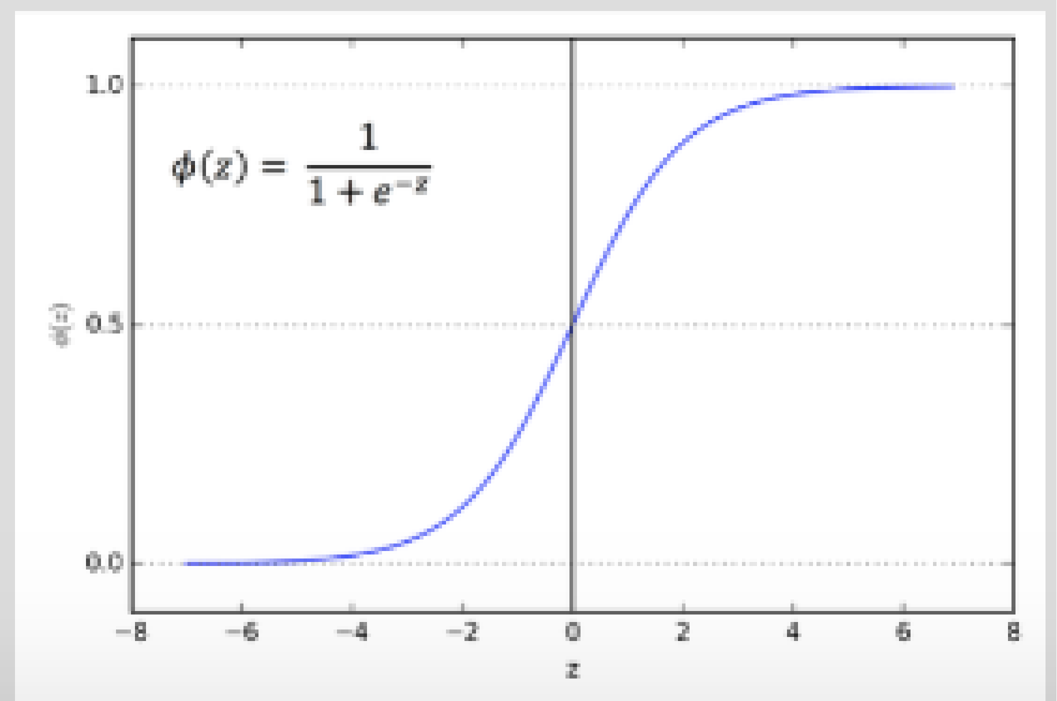


Figure 11: Logistic regression activation function



References

[1] LIME explanation package,
https://homes.cs.washington.edu/~marcotcr/blog/lime/

[2] sklearn LogisticRegression documentation,
https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

[3] Django documentation,
https://www.djangoproject.com/