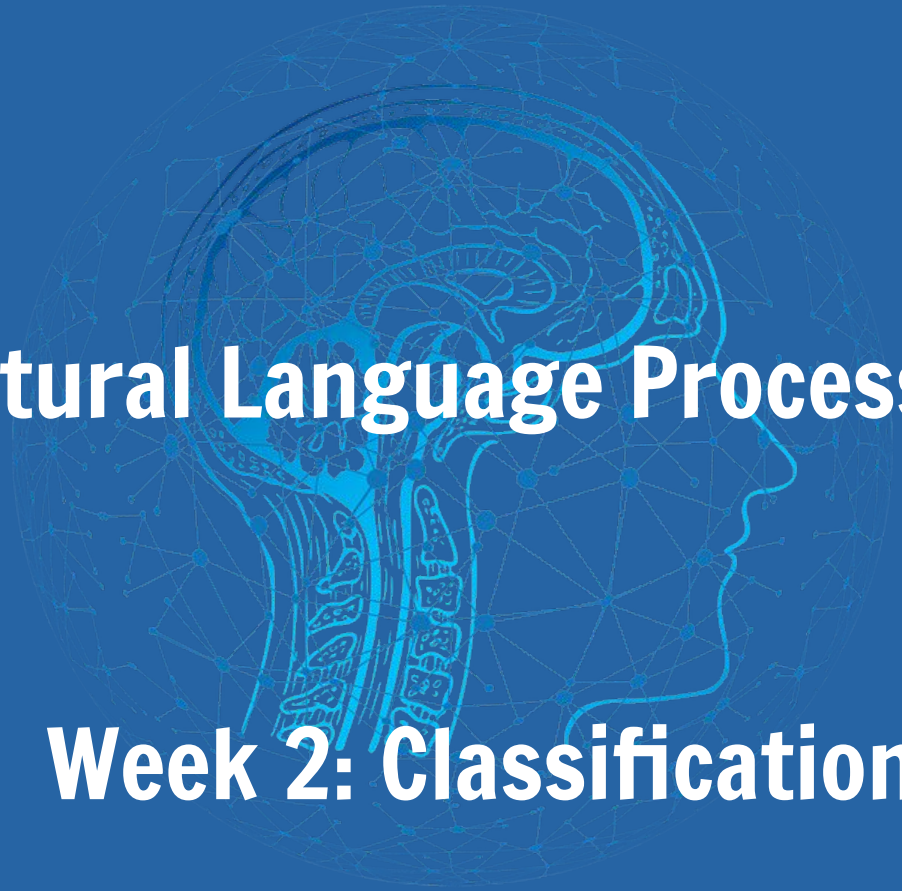
Imperial College London



Natural Language Processing

Week 2: Classification

Introduction - Joe

- NLP PhD student, supervised by Marek Rei
- My research involves making more robust and interpretable NLP models
- Recipient of the 2023 Apple Scholars in AI/ML PhD fellowship
- Previously careers:
  - Teach First teacher
  - PwC / Strategy& consultant

Topic question sheet

NLP Questions on Language Modelling and Classification (weeks 2 and 3)

Joe Stacey

January 2024

1 Naive Bayes

1.1

State the independence assumption used in Naive Bayes.

1.2

For a Bag of Words model, provide an example of two features (words) where this is not an accurate assumption.

1.3

Consider the training corpus in Table 1. Create a Binary Naive Bayes model based only on the features 'good', 'great', 'bad' and 'awful'. Use this model to predict the class of the following review: "tom cruise did it again , so great it's a masterpiece , don't listen to the bad reviews , just enjoy this great film"

Training corpus (after some pre-processing)	Class
almost as good as the first top gun	+
fun throughout , definitely recommend this great film	+
awful , tom cruise had no depth once again . why is his acting so bad	-
better than i expected , not bad at all	+
lived up to the name top gun , what a great film	+
wish i had just rewatched the original , this one was nowhere near as good	-
tom cruise should do the stunts and leave the acting to someone else	-

Table 1: Film reviews, categorised as being either positive (+) or negative (-)

- This is a good starting point for your revision (the questions here are not assessed work)
- I will say when we've covered the content for a question
- Answers will be available in February

Q.x answered

Classification

Outline

1. NLP classification tasks
2. Naive Bayes
3. Logistic Regression
4. Neural Networks (NNs)
5. Recurrent neural networks (RNNs)
6. CNNs
7. Accuracy and F1
8. Our recent research

What is classification?

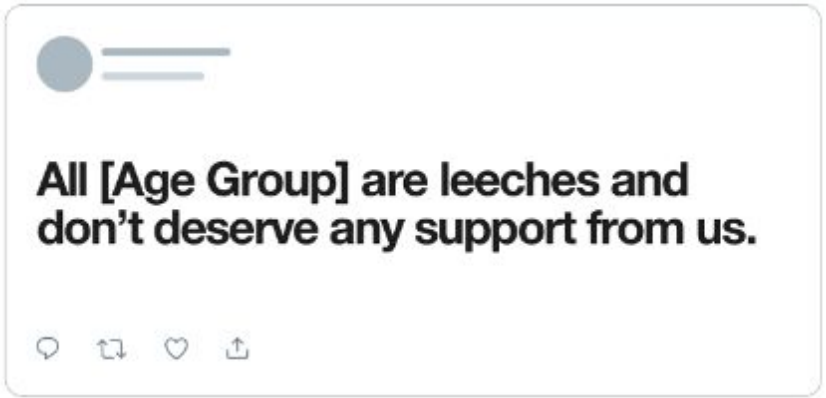
Classification:

Predicting which ‘class’ an observation belongs to.

$$\hat{y} = argmax_y P(y|x)$$

Examples of binary classification

Predicting if a text (or tweet) contains hate speech:



Hate speech

Or

Not hate speech

Examples of binary classification

Predicting if a text (or tweet) contains hate speech:

Model produces a score (logit)

Sigmoid makes this score between 0 and 1

0.5 is our decision boundary

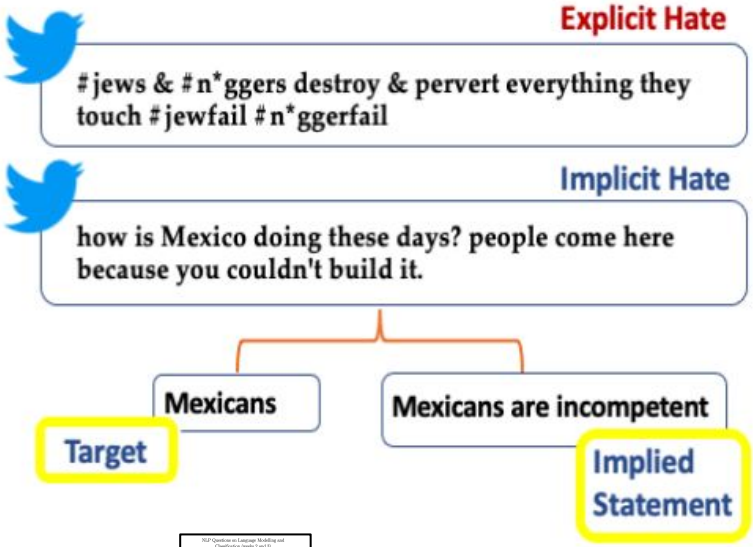
Hate speech

Or

Not hate speech

Examples of multi-class classification

Predicting if a text (or tweet) contains hate speech:



Explicit hate speech

Or

Implicit hate speech

Or

Not hate speech

Q8.1 answered (verbally)

Source: *Latent Hatred: A Benchmark for Understanding Implicit Hate Speech, EMNLP 2021*

More NLP classification tasks / uses

1. Natural Language Inference
2. Sentiment analysis
3. Paraphrase detection
4. Plagiarism detection
5. Grammatical error detection
6. Identifying presence of illness
7. Identifying children at risk (using social services data)
8. Intended sarcasm detection
9. Joke and humour detection
10. Fact verification
11. Hate speech detection
12. Propaganda detection
13. Purpose of dark web pages
14. Predicting legal judgements

And many more…

Natural Language Inference

- **Premise**: The kitten is climbing the curtains again
- **Hypothesis**: The kitten is sleeping

**

Labels:

- **Entailment**: if the hypothesis is implied by the premise
- **Contradiction**: if the hypothesis contradicts the premise
- **Neutral**: otherwise

** This is Hamish - he likes to fall asleep in very unusual positions

Questions so far?

10 mins

Outline

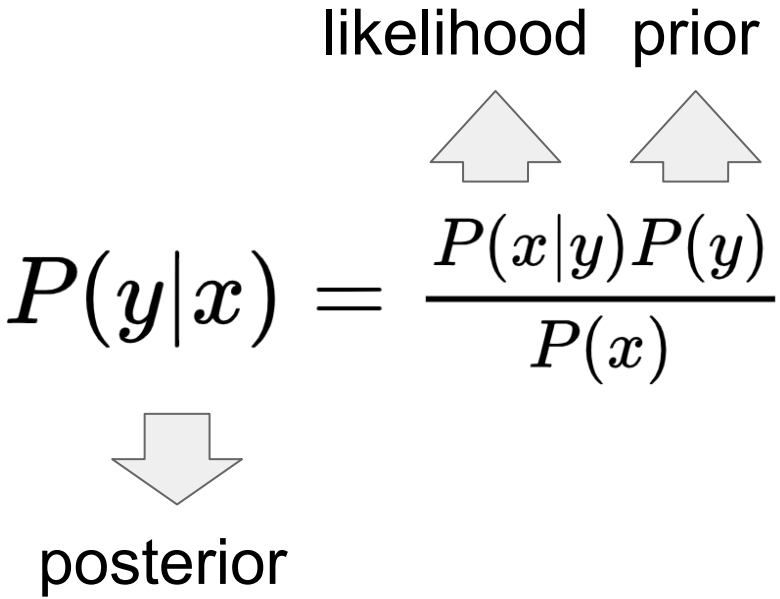
1. NLP classification tasks
2. **Naive Bayes**
3. Logistic Regression
4. Neural Networks (NNs)
5. Recurrent neural networks (RNNS)
6. CNNs
7. Accuracy and F1
8. Our recent research

Naive Bayes Classifier

Introducing Naive Bayes

Naive Bayes Classifier

Bayes' rule:



Naive Bayes Classifier

likelihood prior

$$\hat{y} = argmax_y P(y|x) = argmax_y P(x|y)P(y)$$

Naive Bayes Classifier

Q1.1 answered

Q1.2 answered

- x is a set of features $x_1, x_2, \ldots, x_I$

$$\hat{y} = argmax_y P(x_1, x_2, \ldots, x_I | y) P(y)$$

- Naive Bayes **independence assumption**:

$$P(x_1, x_2, \ldots, x_I | y) = P(x_1 | y) \cdot P(x_2 | y) \cdot \cdots \cdot P(x_I | y)$$

$$\hat{y} = argmax_y P(y) \prod_{i=1}^{I} P(x_i | y)$$

Naive Bayes Classifier

Naive Bayes in NLP

(using a sentiment analysis example - movie reviews)

Input representations

- Raw input is transformed into a numerical representation,

  i.e. each input x is represented by a feature vector:

$$[x_1, x_2, \ldots, x_I]$$

Input representations

After any pre-processing:

- We can use a Bag of Words (BoW) approach

Our review:

Another good movie for holiday watchers.... a little twist from the ordinary scrooge movie. Enjoyable.

Our review in a bag of words:

Input representations

Example Bag of Words representation:

Review #1:

This **was another** good movie for holiday watchers. There **was a** nice little twist at the end.

	a	**about**	**another**	**and**	**...**	**was**	**you**
Review #1	1	0	1	0		2	0

Naive Bayes Classifier

Collecting statistics from our training data

Naive Bayes Classifier

Training corpus	class
Another good movie for holiday watchers. A little twist from the ordinary scrooge movie. Enjoyable.	+
It seems like just about everybody has made a Christmas Carol movie. Others are just bad and the time period seems to be perfect.	+
If you're looking for the same feel good one but in a new setting, this one's for you.	+
This is a first for me, I didn't like this movie. It was really bad.	-
It was good but the Christmas Carol by Dickens was emotionally moving.	-

Naive Bayes Classifier

With some limited data processing….

Training corpus	class
another good movie for holiday watchers . a little twist from the ordinary scrooge movie . enjoyable .	+
it seems like just about everybody has made a christmas carol movie . others are just bad and the time period seems to be perfect .	+
if you're looking for the same feel good one but in a new setting , this one's for you .	+
this is a first for me , i didn't like this movie . it was really bad .	-
it was good but the christmas carol by dickens was emotionally moving .	-

Naive Bayes Classifier

Our bag of words representation….

Review	a	about	another	and	...	was	you
1	1	0	1	0		0	0
2	1	1	0	1		0	0
3	1	0	0	0		0	2
4	1	0	0	0		1	0
5	0	0	0	0		1	0

Naive Bayes Classifier

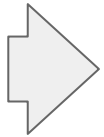
Alternatively, using some feature extraction….

Training corpus	**good**	**movie**	**bad**	class
another **good movie** for holiday watchers . a little twist from the ordinary scrooge movie . enjoyable .	1	1	0	+
it seems like just about everybody has made a christmas carol movie . others are just **bad** and the time period seems to be perfect .	0	0	1	+
if you 're looking for the same feel **good** one but in a new setting , this one 's for you .	1	0	0	+
this is a first for me , i didn 't like this **movie** . it was really **bad** .	0	1	1	-
it was **good** but the christmas carol by dickens was emotionally moving .	1	0	0	-

Naive Bayes Classifier

Training a ‘model’ just involves collecting statistics

$$P(y)$$



$$P(+) = \frac{3}{5}$$

3 positive examples

$$P(-) = \frac{2}{5}$$

2 positive examples

Naive Bayes Classifier

Frequency of the word for this class

$$P(good|+) = \frac{2}{4}$$

Total count of words for this class

Half (two out of four) of the words within the positive class are ‘good’

Naive Bayes Classifier

What happens if one of our probabilities is 0?

Naive Bayes Classifier

Add-one smoothing:

$$P(x_i|y) = \frac{count(x_i,y)+1}{\sum_{x \in V}(count(x,y)+1)} = \frac{count(x_i,y)+1}{(\sum_{x \in V} count(x,y))+|V|}$$

V is the vocabulary across both classes

Naive Bayes Classifier

Add-one smoothing:

$$P(x_i|y) = \frac{count(x_i,y)+1}{\sum_{x \in V}(count(x,y)+1)} = \frac{count(x_i,y)+1}{(\sum_{x \in V} count(x,y))+|V|}$$

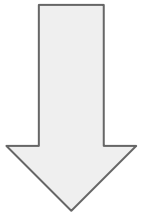
V is the vocabulary across classes

$$P(good|+) = \frac{2+1}{4+3}$$

Naive Bayes Classifier

Test example:

Not as **good** as the old **movie**, rather **bad**.



Pre-process & extract some features

good movie bad

Naive Bayes Classifier

Test example: Not as **good** as the old **movie**, rather **bad**.

$$P(good|+) = \frac{2+1}{4+3}$$

$$P(movie|+) = \text{R}$$

$$P(bad|+) = \text{L}$$

$$P(good|-) = \frac{1+1}{3+3}$$

$$P(movie|-) = \text{R}$$

$$P(bad|-) = \text{L}$$

Naive Bayes Classifier

Alternatively, using some feature extraction….

Training corpus	**good**	**movie**	**bad**	class
another **good movie** for holiday watchers . a little twist from the ordinary scrooge movie . enjoyable .	1	1	0	+
it seems like just about everybody has made a christmas carol movie . others are just **bad** and the time period seems to be perfect .	0	0	1	+
if you 're looking for the same feel **good** one but in a new setting , this one 's for you .	1	0	0	+
this is a first for me , i didn 't like this **movie** . it was really **bad** .	0	1	1	-
it was **good** but the christmas carol by dickens was emotionally moving .	1	0	0	-

Naive Bayes Classifier

Test example: Not as **good** as the old **movie**, rather **bad**.

$$P(good|+) = \frac{2+1}{4+3}$$

$$P(movie|+) = \frac{1+1}{4+3}$$

$$P(bad|+) = \frac{1+1}{4+3}$$

$$P(good|-) = \frac{1+1}{3+3}$$

$$P(movie|-) = \frac{1+1}{3+3}$$

$$P(bad|-) = \frac{1+1}{3+3}$$

Naive Bayes Classifier

Test example: Not as **good** as the old **movie**, rather **bad**.

$$P(+) = \frac{3}{5}$$

$$P(+)P(x|+) = \frac{3}{5} \times \frac{3 \times 2 \times 2}{7^3}$$

$$P(+)P(x|+) = 0.021$$

$$P(-) = \frac{2}{5}$$

$$P(-)P(x|-) = \frac{2}{5} \times \frac{2 \times 2 \times 2}{6^3}$$

$$P(-)P(x|-) = 0.014$$

Naive Bayes Classifier

Test example: Not as **good** as the old **movie**, rather **bad**.

$$P(+) = \frac{3}{5}$$

$$P(+)P(x|+) = \frac{3}{5} \times \frac{3 \times 2 \times 2}{7^3}$$

$$P(+)P(x|+) = 0.021$$

$$P(-) = \frac{2}{5}$$

$$P(-)P(x|-) = \frac{2}{5} \times \frac{2 \times 2 \times 2}{6^3}$$

$$P(-)P(x|-) = 0.014$$

Naive Bayes Classifier

Q1.3 answered

Test example: Not as **good** as the old **movie**, rather **bad**.

$$P(+) = \frac{3}{5}$$

$$P(-) = \frac{2}{5}$$

$$P(+)P(x|+) = \frac{3}{5} \times \frac{3\times2\times2}{7^3}$$

$$P(-)P(x|-) = \frac{2}{5} \times \frac{2\times2\times2}{6^3}$$

$$P(+)P(x|+) = 0.021$$

$$P(-)P(x|-) = 0.014$$

Improvements

Some improvements we can make for sentiment analysis….

** According to Jurafsky & Martin, Speech and Language Processing*

Improvement #1: binary naive bayes

How about: Not as **good** as the old **movie**, rather **bad movie**.

$$P(+) = \frac{3}{5}$$

$$P(+)P(x|+) = \frac{3}{5} \times \frac{3 \times 2 \times 2}{7^3}$$

$$P(-) = \frac{2}{5}$$

$$P(-)P(x|-) = \frac{2}{5} \times \frac{2 \times 2 \times 2}{6^3}$$

I have added in the word ‘movie’ again, but for Binary Naive Bayes nothing changes!

With **binary naive bayes**, we only consider if a feature is present, rather than considering every time it occurs. Note, this means re-calculating the conditional probabilities from the training data.

Improvement #2: controlling for negation

Review:

I didn’t like the movie, but it was better than Top Gun

Becomes:

I didn’t NOT_like NOT_the NOT_movie, but it was better than Top Gun

We append ‘NOT_’ after any logical negation (e.g. ***n’t***, ***not***, ***no***, ***never***) until the next punctuation mark

Problems

- Conditional independence assumption
- Features considered equally important
- Context of words not taken into account
- New words (not seen at training) cannot be used

Questions so far?

Outline

1. NLP classification tasks
2. Naive Bayes
3. **Logistic Regression**
4. Neural Networks (NNs)
5. Recurrent neural networks (RNNS)
6. CNNs
7. Accuracy and F1
8. Our recent research

Logistic Regression

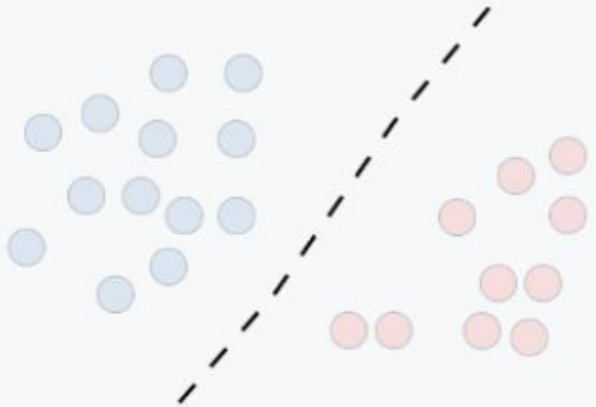
- Discriminative vs Generative algorithms
- Discriminative algorithms directly learn $P(Y \mid X)$
- They learn the input features most useful to discriminate between the different classes, without considering the likelihood of the input itself

Q2.2 answered

Generative models, on the other hand, learn:

$$P(X, Y)$$

Logistic Regression

	Discriminative model	Generative model
What's learned	Decision boundary	Probability distributions of the data
Illustration		

***Source**: Stanford CS 229 course notes*

Logistic Regression

Q2.3 answered

Logistic function

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

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

w = How important an input feature is to the classification decision

Logistic function is a linear transformation followed by a sigmoid…

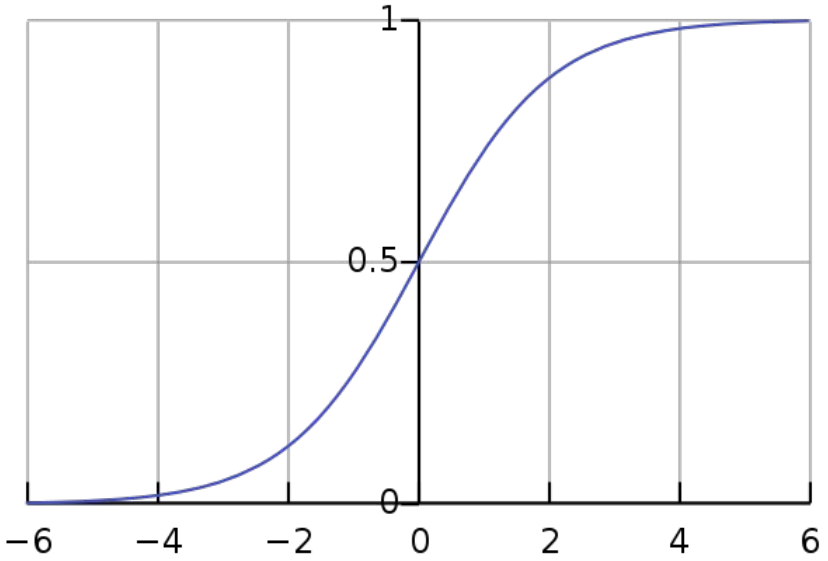
Sigmoid Function

$$P(y=1) = \frac{1}{1+e^{-(w \cdot x + b)}}$$

How do we make a decision?

$$\hat{y} = \begin{cases} 1 & \text{if } P(y=1|x) \text{ is } > 0.5 \\ 0 & \text{otherwise} \end{cases}$$

Sigmoid:



Logistic Regression

- What weights would we expect for our three features?
  - ***movie***, ***bad*** *and* ***good***

For simplicity, in our worked example we only consider the features ***bad*** and ***movie***

** We have y=1 if a movie is POSITIVE

Logistic Regression

How inference works if we have learnt w and b:

- Test example: *Not as good as the old **movie**, rather **bad**.*

x_1	count of **movie**	1
x_2	count of **bad**	1

$$x = [1.0, 1.0]$$
$$w = [-5.0, 2.5]$$
$$b = 0.1$$

Logistic Regression

Test example: *Not as good as the old **movie**, rather **bad**.*

$$x = [1.0, 1.0] \quad w = [-5.0, 2.5] \quad b = 0.1$$

$$P(y = 1|x) = g(z)$$
$$= g([-5.0, 2.5] \cdot [1.0, 1.0] + 0.1)$$
$$= g(-2.4)$$
$$= 0.08$$

$$P(y = 0|x) = 1 - g(z)$$
$$= 0.92$$

Logistic Regression

We learn parameters to make the model predictions close to our labels:

- Our loss function measures the distance between the true and predicted label
- Optimization algorithm minimises this function, usually **gradient descent**

Logistic Regression

How close is the predicted distribution Q to the true distribution P?

$$H(P,Q) = -\sum_i P(y_i) \log Q(y_i)$$

Logistic Regression

Finding the loss from our example (using log to the base e):

$$P = [1\ 0]$$

$$Q_1 = [0.92\ 0.08]$$

$$H(P, Q) = -(1\, log\, 0.92 + 0\, log\, 0.08)$$
$$= -\, log\, 0.92 = 0.08$$

Logistic Regression

Finding the loss from our example (using log to the base e):

$$P = [1\ 0]$$

$$Q_2 = [0.56\ 0.44]$$

$$H(P,Q) = -(1\, log\, 0.56 + 0\, log\, 0.44)$$

$$= -\, log\, 0.56 = 0.58$$

Logistic Regression with multiple classes

Sentiment analysis with 3 classes: +, - and neutral:

Input features:

x_1	count of **bad**	1
x_2	count of **good**	1
x_3	count of **and**	2

Logistic Regression

Weights are learnt per class

$$w_+ = \begin{bmatrix} 0.0 \\ 1.9 \\ 0.0 \end{bmatrix} \qquad w_- = \begin{bmatrix} 1.5 \\ 0.4 \\ 0.0 \end{bmatrix} \qquad w_n = \begin{bmatrix} 0.0 \\ 0.0 \\ 0.0 \end{bmatrix}$$

Logistic Regression

$$z_1 = ([1.0, 1.0, 2.0] \cdot [0.0, 1.9, 0.0]) + 0.1$$

$$z_2 = ([1.0, 1.0, 2.0] \cdot [1.5, 0.4, 0.0]) \text{ - 0.9}$$

$$z_3 = ([1.0, 1.0, 2.0] \cdot [0.0, 0.0, 0.0]) + 0.1$$

Q2.4 answered verbally

Logistic Regression

Probability distribution over classes:

$$\mathbf{z} = [2.0,\ 1.0,\ 0.1]$$

Softmax function

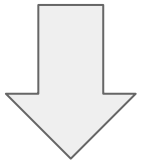
$$y = g(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}} \qquad 1 \leq i \leq k$$

Replaces our sigmoid function: $y = g(z) = \frac{1}{1+e^{-z}}$

Logistic Regression

$$y = g(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}}$$

$$\mathbf{z} = [2.0,\ 1.0,\ 0.1]$$



$$y = \left[\frac{e^{2.0}}{e^{2.0}+e^{1.0}+e^{0.1}},\ \frac{e^{1.0}}{e^{2.0}+e^{1.0}+e^{0.1}},\ \frac{e^{0.1}}{e^{2.0}+e^{1.0}+e^{0.1}}\right] = [0.66,\ 0.24,\ 0.1]$$

Logistic Regression

Q2.1 answered

Logistic Regression:

- Considers the importance of features, so better at dealing with correlated features
- Better with larger datasets

Naive Bayes:

- Very quick to train
- Some evidence it works well on very small datasets

Some problems remain, e.g. considering the interaction of different features

Speech and Language Processing. Daniel Jurafsky & James H. Martin. Chapter 5

Questions so far?

45 mins

Simple NLP baselines

Why should I care….

When might you use them?

They can help us understand our dataset better:

- They can help us to understand which features are influential or correlate with each class
- We can also compare the performance to to more powerful models

Natural Language Inference

Not examinable

- **Premise**: The kitten is climbing the curtains again
- **Hypothesis**: The kitten is sleeping

Label: *Contradiction*

What we can find out:

- The word *sleeping* strongly correlates with the contradiction class
- A simple model will have the capability to learn from a bias, but not from the task itself

Simple NLP baselines

Not examinable

You could even downsample training observations that a simple baseline model predicts correctly…*

*This has been done using a simple TinyBERT baseline: *Learning from others' mistakes: Avoiding dataset biases without modeling them (2021)*

Debiasing

Not examinable

- You will find results something like

	In-distribution test set	**out-of-distribution test set**
Normal training	Model performs great	Not so great
Training with debiasing		

Debiasing

Not examinable

- You will find results something like

	In-distribution test set	**out-of-distribution test set**
Normal training	Model performs great	Not so great
Training with debiasing	Little bit worse than normal training	Better than normal training

Debiasing

Not examinable

Other possible strategies:

- Augment with more data so that it ‘balances’ the bias
- Filter your data
- Prevent a classifier finding the bias in your model representations

Break

50 mins

Outline

1. NLP classification tasks
2. Naive Bayes
3. Logistic Regression
4. **Neural Networks (NNs)**
5. Recurrent neural networks (RNNS)
6. CNNs
7. Accuracy and F1
8. Our recent research

Neutral Networks

Linear layer: $z = \mathbf{w} \cdot \mathbf{x} + b = \sum_{i=0}^{I} w_i x_i + b$

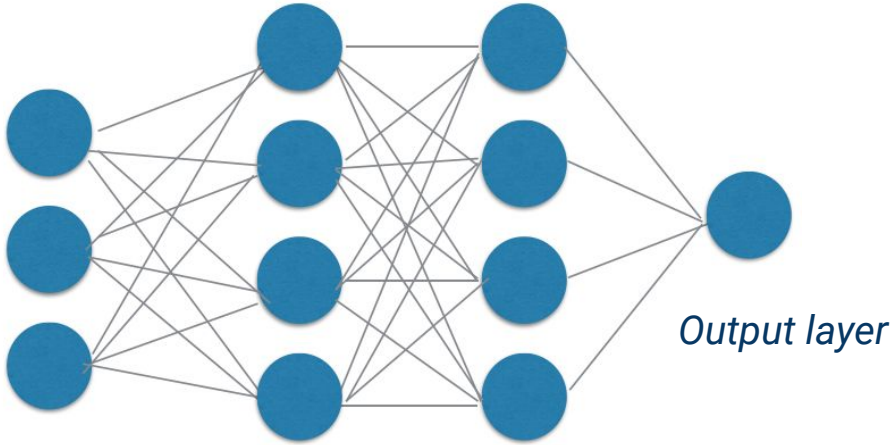
Non-linear activation function: $y = g(z)$

Neutral Networks

Q8.2 answered verbally
Q8.3 answered verbally

Fully-connected layers

$$h^1 = g^1(x\mathsf{W}^1 + \mathsf{b}^1) \quad h^2 = g^2(h^1\mathsf{W}^2 + \mathsf{b}^2) \quad y = h^2\mathsf{W}^3$$

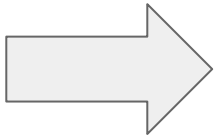


$$FFN(x) = (g^2(g^1(x\mathsf{W}^1 + \mathsf{b}^1))\mathsf{W}^2 + \mathsf{b}^2)\mathsf{W}^3$$

Learnt feature representations

Inputs (very basic model)

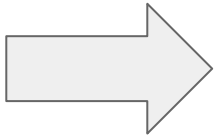
- One-hot representation of words



the	1	0	0	0
movie	0	1	0	0
is	0	0	1	0
good	0	0	0	1

Inputs (better model)

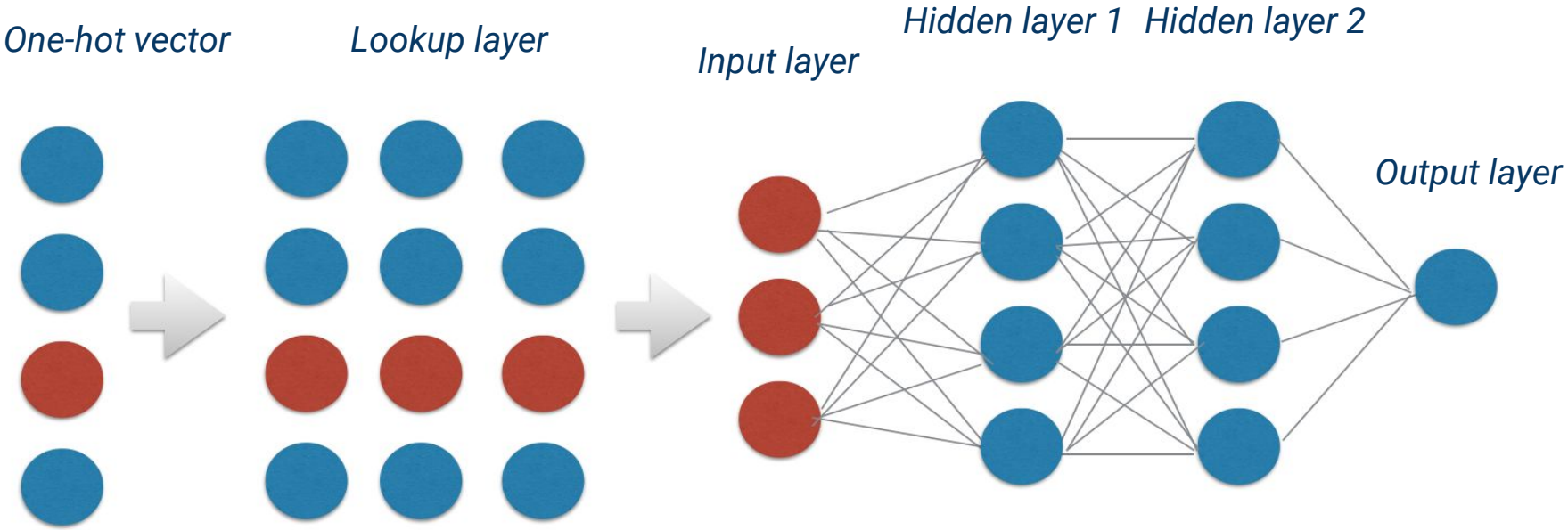
- Automatically learnt dense feature representations, or
- Pre-trained dense representations



the	0.4	0.2	-0.1
movie	0.8	-0.5	0.4
is	0.8	-0.3	0.1
good	0.2	-0.1	0.6

Neutral Networks

For a single word:



$1 \times |V|$ $|V| \times d$ $1 \times d$

Document representation

How to get a document representation of sentence of fixed dimensionality?

Document 1: l = 4

the	0.4	0.2	-0.1
movie	0.8	-0.5	0.4
is	0.8	-0.3	0.1
good	0.2	-0.1	0.6

Document 2: l = 2

excellent	0.4	0.2	-0.1
!	0.8	-0.5	0.4

Document representation

Document 1: l = 4

the	0.4	0.2	-0.1
movie	0.8	-0.5	0.4
is	0.8	-0.3	0.1
good	0.2	-0.1	0.6
average	0.55	-0.175	0.25

Document 2: l = 2

excellent	0.4	0.2	-0.1
!	0.8	-0.5	0.4
average	0.6	-0.15	0.15

Document representation

Document 1: l = 4

the	0.4	0.2	-0.1
movie	0.8	-0.5	0.4
is	0.8	-0.3	0.1
good	0.2	-0.1	0.6

Document 2: l = 2

excellent	0.4	0.2	-0.1
!	0.8	-0.5	0.4
-	0	0	0
-	0	0	0

- Could I do this?

Document representation

Document 1: l = 4

the	0.4	0.2	-0.1
movie	0.8	-0.5	0.4
is	0.8	-0.3	0.1
good	0.2	-0.1	0.6

Document 2: l = 2

excellent	0.4	0.2	-0.1
!	0.8	-0.5	0.4
-	0	0	0
-	0	0	0

- This is a really bad idea:
  - Model architecture fixed to sentence length size
  - Model weights learnt for specific word positions

Why neural networks

- Automatically learned features
- Flexibility to fit highly complex relationships in data
  - **But**: they require more data to learn more complex patterns

Questions so far?

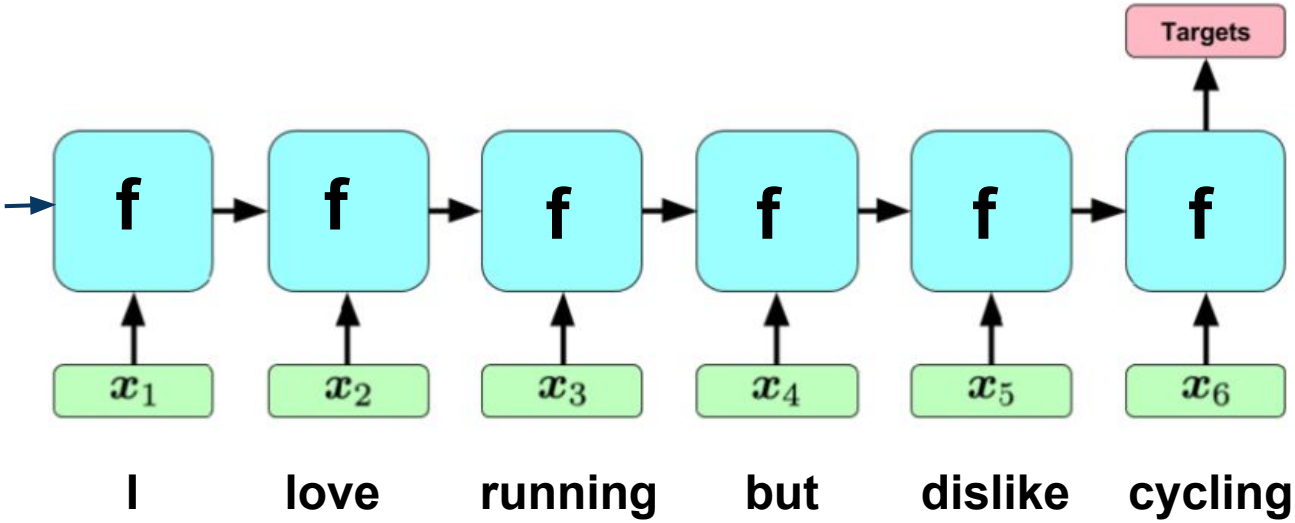
1hr 10 mins

Outline

1. NLP classification tasks
2. Naive Bayes
3. Logistic Regression
4. Neural Networks (NNs)
5. **Recurrent neural networks (RNNS)**
6. CNNs
7. Accuracy and F1
8. My recent research

Recurrent Neural Networks

- Natural language data - sequences
- Value of a unit depends on own previous outputs
- The last hidden state is the input to the output layer



Vanilla RNNs

- RNN (f) computes its next state h_{t+1} based on:
  - **Hidden state vector** and **input vector** at time t

$$h_{t+1} = f(h_t, x_t) = \tanh\left(W h_t + U x_t\right)$$

- Its hidden state is carried along
- Main parameters, matrices W and U:

$$W \in \mathbb{R}^{\text{????}} \quad \text{hidden-to-hidden}$$

$$U \in \mathbb{R}^{\text{????}} \quad \text{input-to-hidden}$$

Vanilla RNNs

- RNN (f) computes its next state h_{t+1} based on:
  - **Hidden state vector** and **input vector** at time t

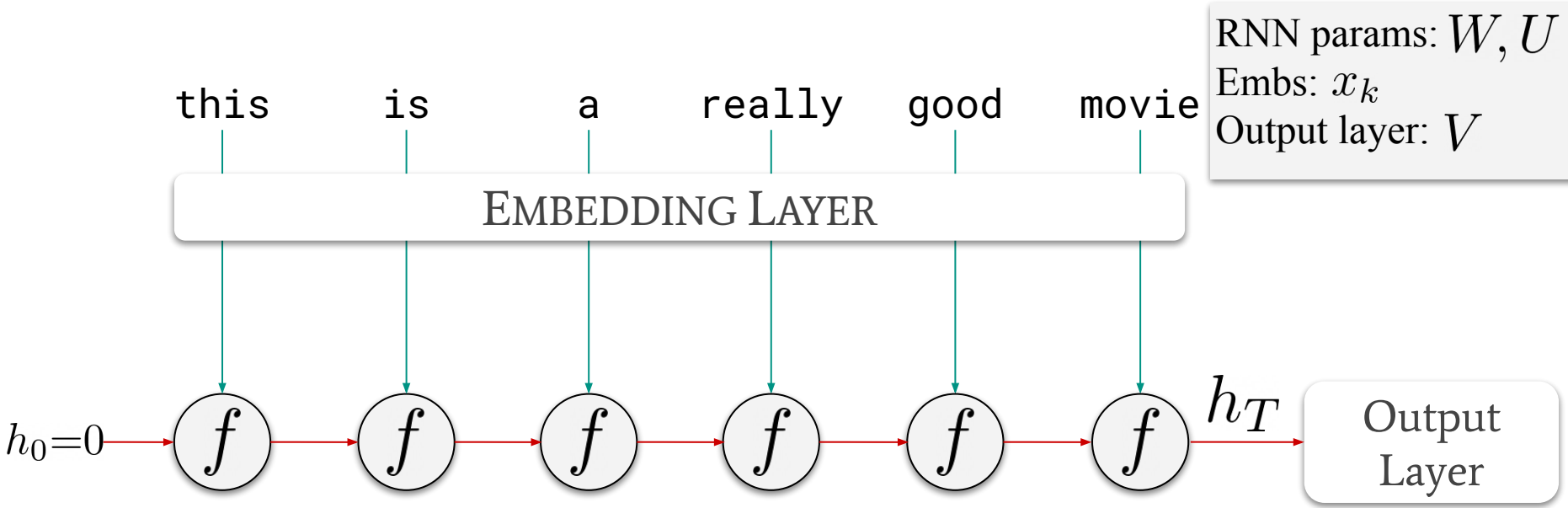
$$h_{t+1} = f(h_t, x_t) = \tanh\left(W h_t + U x_t\right)$$

- Its hidden state is carried along
- Main parameters, matrices W and U:

$$W \in \mathbb{R}^{H \times H} \quad \text{hidden-to-hidden}$$

$$U \in \mathbb{R}^{H \times E} \quad \text{input-to-hidden}$$

Vanilla RNNs

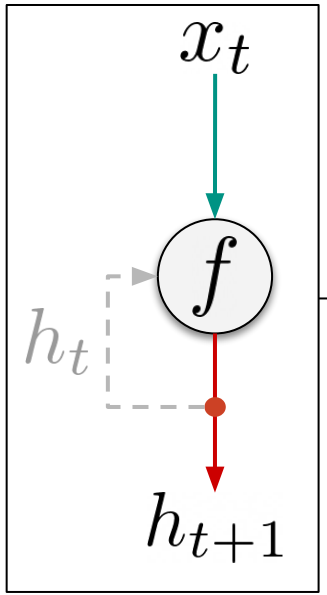


$$y_t = W_{hy} h_t + B_y$$

What is the dimensionality of W_{hy}?

t=1 t=2 t=3 t=4 ... t=T

Vanilla RNNs




```
size_H = 100, size_E = 20
E = torch.nn.Embedding(vocab_size, size_E)
U = torch.rand(size_H, size_E, requires_grad=True)
W = torch.rand(size_H, size_H, requires_grad=True)

sent = [“this”, “is”, “a”, “really”, “good”, “movie”]

# Start as zero
h_t = torch.zeros(size_H, 1)
loss = 0

for i in range(len(sent) - 1):
  x_t = E(sent[i])
  h_t = torch.tanh(W.matmul(h_t) + U.matmul(x_t))
```

Limitations - the vanishing gradient problem

- **Vanishing gradient problem**
  - The model is less able to learn from earlier inputs:
    - Tanh derivatives are between 0 and 1
    - Sigmoid derivatives are between 0 and 0.25
  - Gradient for earlier layers involves repeated multiplication of the same matrix W
    - Depending on the dominant eigenvalue this can cause gradients to either ‘vanish’ or ‘explode’

Questions so far?

1hr 25 mins

Outline

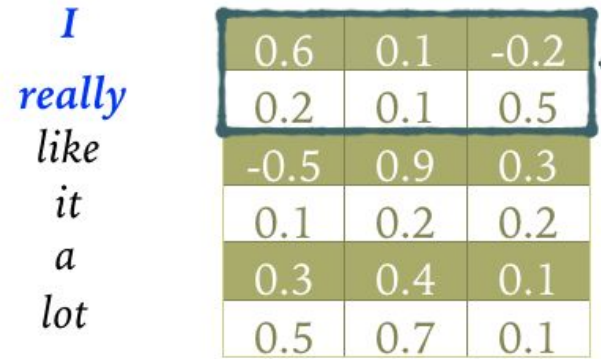
1. NLP classification tasks
2. Naive Bayes
3. Logistic Regression
4. Neural Networks (NNs)
5. Recurrent neural networks (RNNS)
6. **CNNs**
7. Accuracy and F1
8. Our recent research

Convolutional Neural Networks

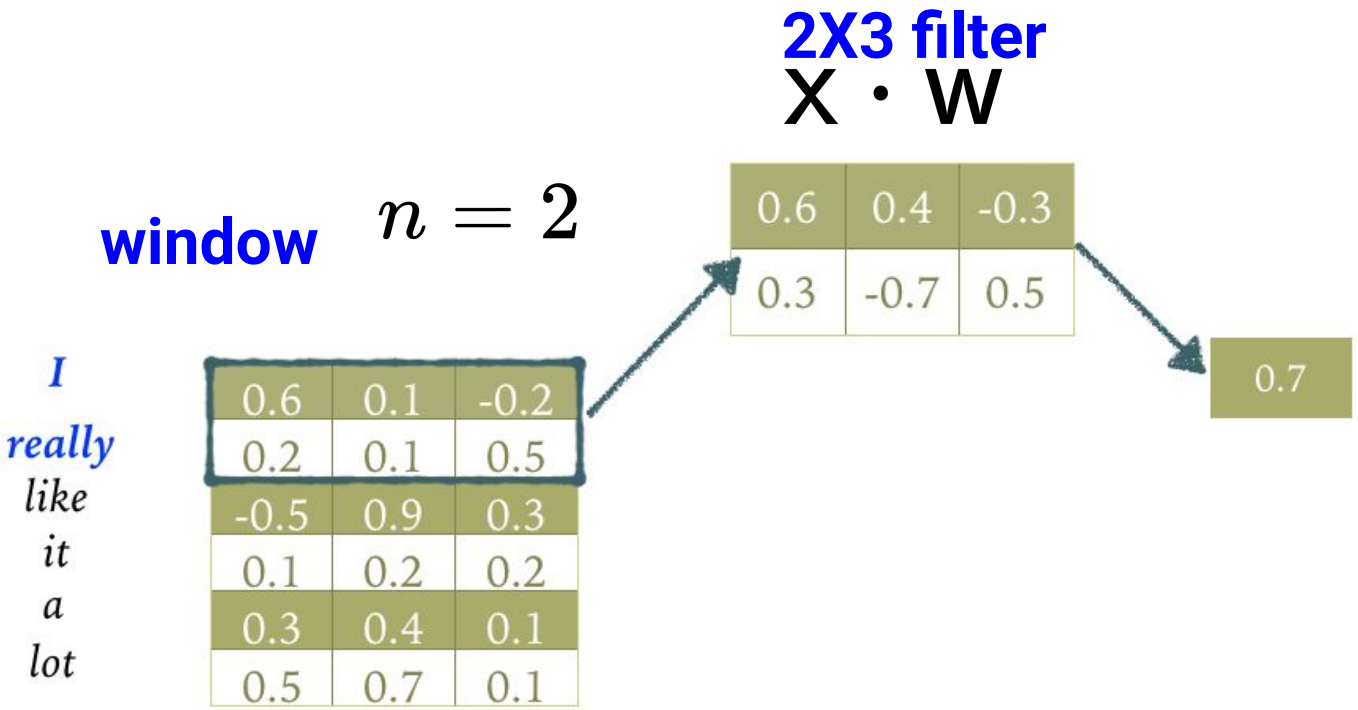
- CNNs are composed of a series of **convolution** layers, **pooling** layers and **fully connected** layers
    - Convolution layers:
        - Detect important patterns in the inputs
    - Pooling layers:
        - Reduce dimensionality of features
        - Transform them into a fixed-size
    - Fully connected layers:
        - Train weights of learned representation for a specific task

Convolutional Neural Networks

- Filter: sliding window over full rows (words) in one direction
  - **Filter width** = embedding dimension
  - **Filter height** = normally 2 to 5 (bigrams to 5-grams)



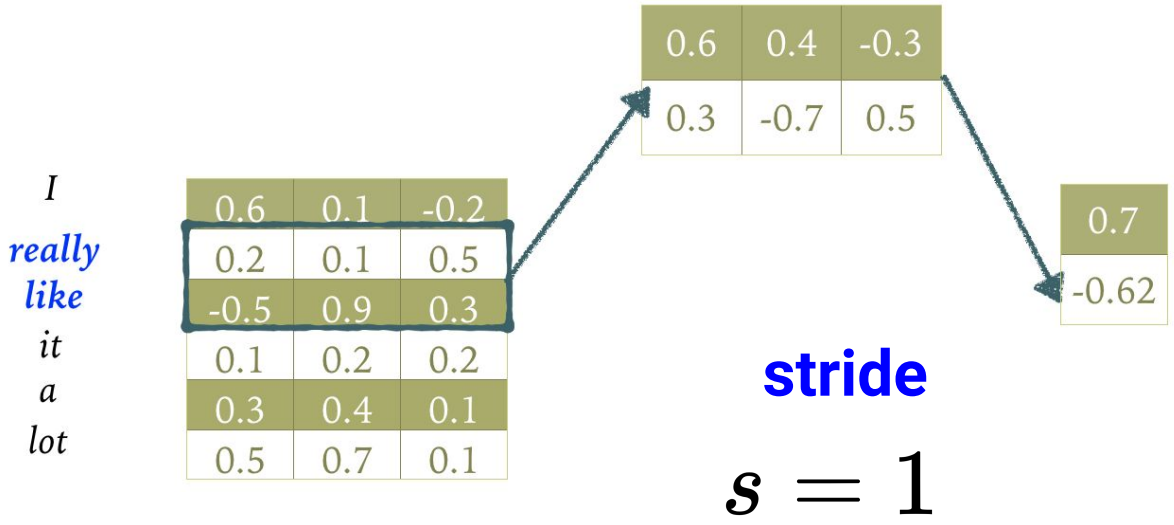
Convolutional Neural Networks



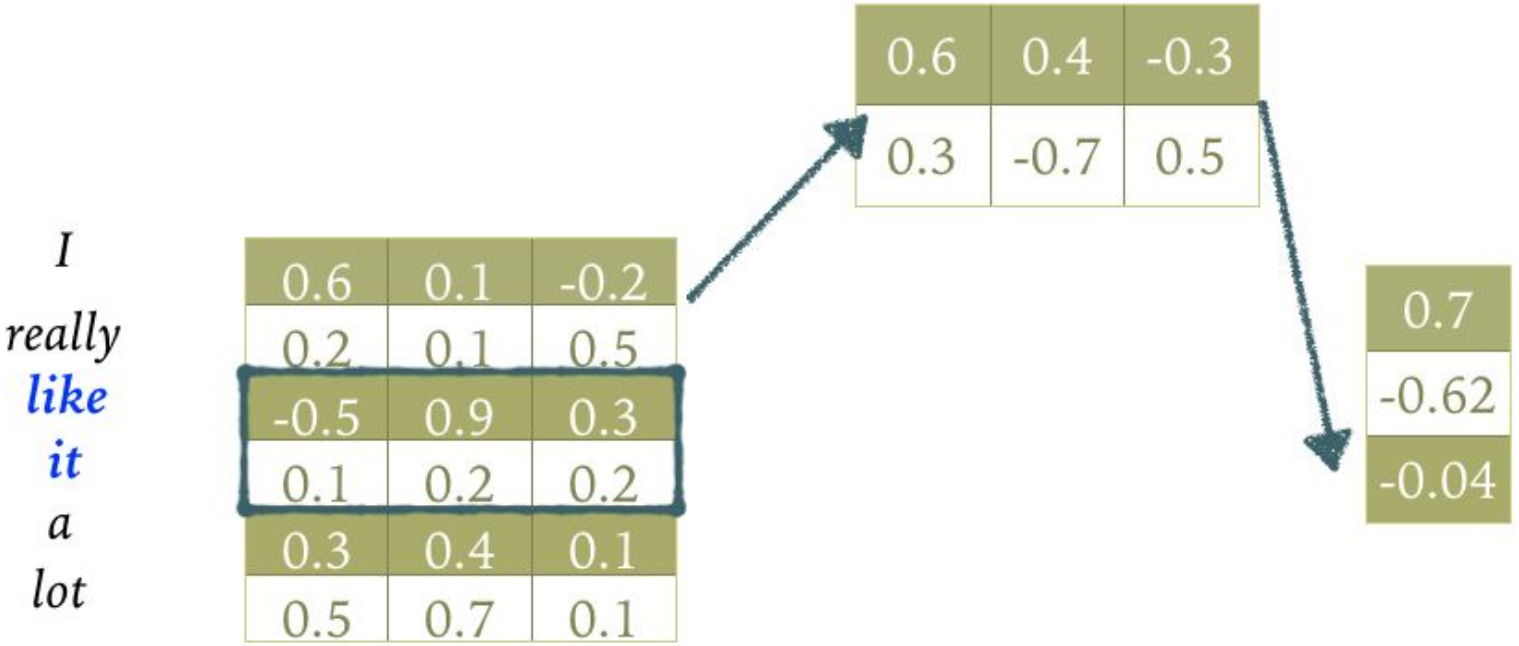
You also have padding and strides…

Convolutional Neural Networks

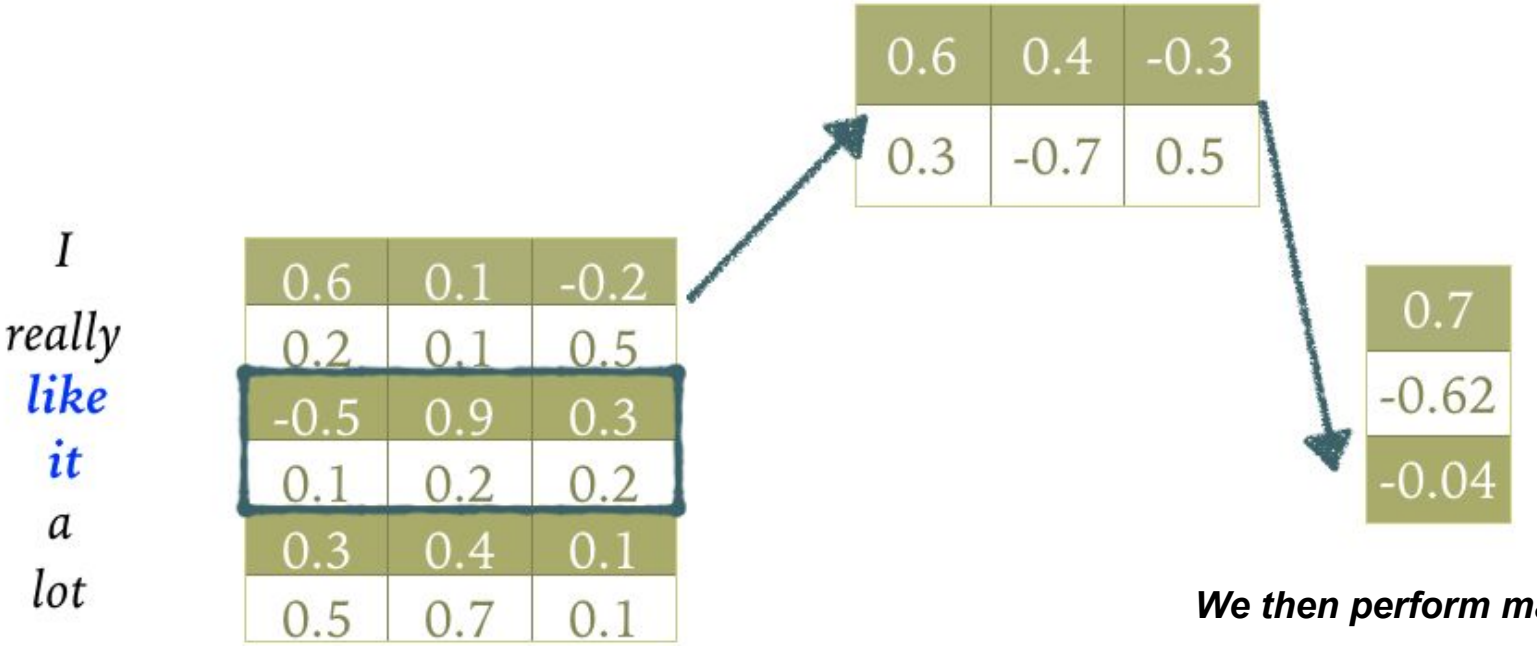
- **Stride size** - how much to shift the filter at each step



Convolutional Neural Networks



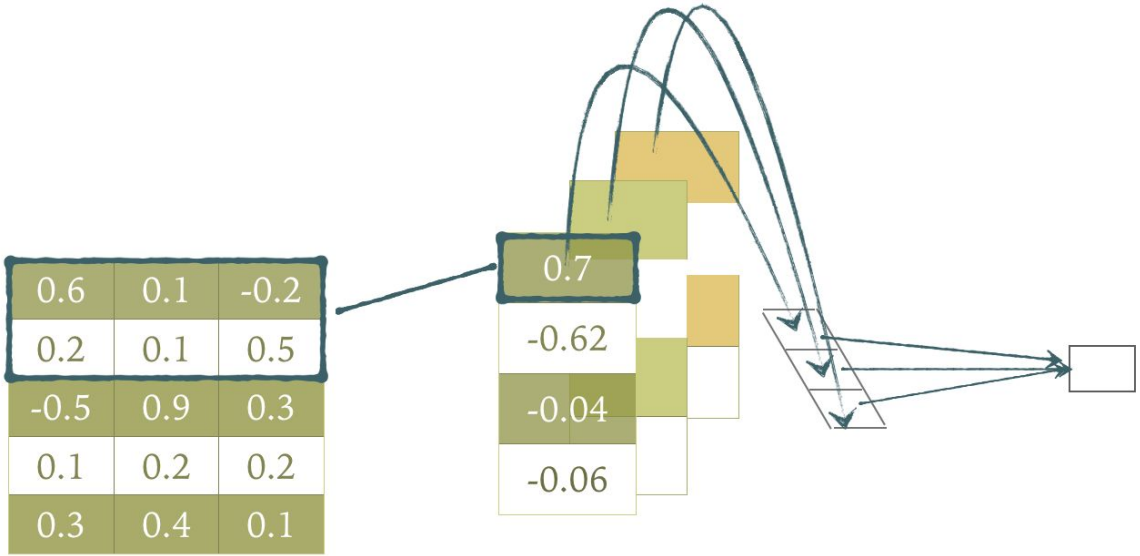
Convolutional Neural Networks



We then perform max pooling

Convolutional Neural Networks

- If we have d different parallel filters, then we have a d-dimensional representation



RNNs vs CNNs

- Understanding the strengths of **both types of models:**

- CNNs can perform well if the task involves key phrase recognition
- Whereas RNNs perform better when you need to understand longer range dependencies

Determining if an article is Fake News based on the headline:

- `“Hillary Clintons election fraud finally exposed. California stolen from Bernie Sanders!”`

- `Hillary Clinton Needed Someone to 'Sober Her Up' at 4:30 in the Afternoon`

Comparative Study of CNN and RNN for Natural Language Processing, 2017. Yin et al

Evaluation metrics for classification

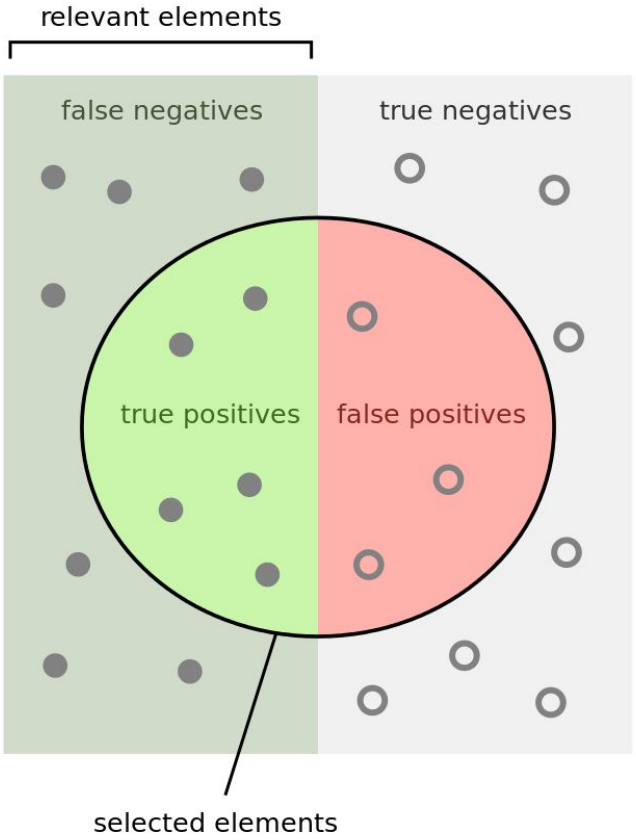
1h 35 mins

Outline

1. NLP classification tasks
2. Naive Bayes
3. Logistic Regression
4. Neural Networks (NNs)
5. Recurrent neural networks (RNNS)
6. CNNs
7. **Accuracy and F1**
8. Our recent research

Evaluation - when accuracy isn't ideal

Q3.1 answered



For two classes:

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

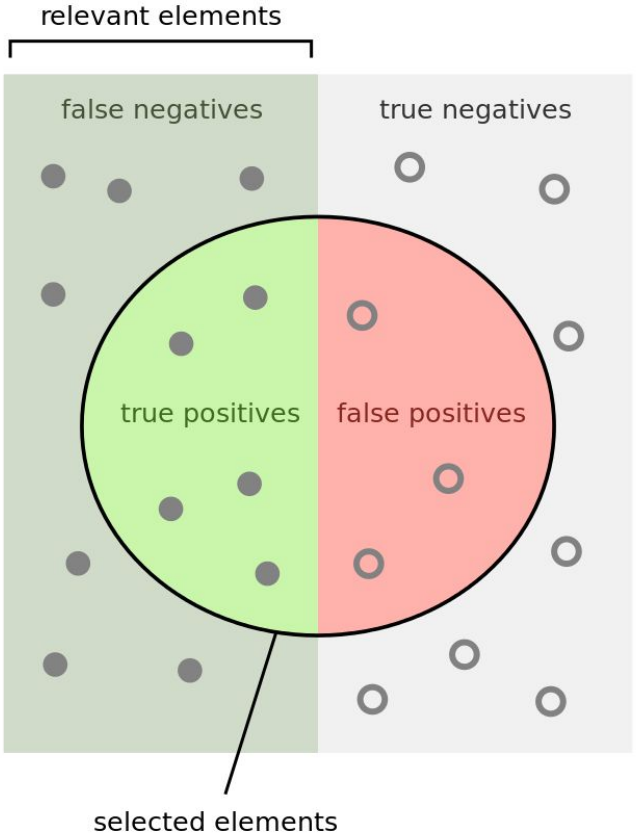
$$TN = 999900$$

$$FN = 100$$

$$accuracy = \frac{999900}{1000000} = 99.99\%$$

Evaluation - micro vs macro averaging

Q3.3 answered



F-measure:

$$F_1 = 2 \times \frac{precision \times recall}{precision + recall}$$

$$F1 = \frac{TP}{TP + \frac{1}{2}(FP + FN)}$$

Micro vs Macro F1:

- **Macro averaging** - average of each class F1 scores:
  - Increases the emphasis on less frequent classes
- **Micro averaging** - TPs, TNs, FNs, FPs are summed across each class

Let’s now see something fun

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

$$Accuracy = \frac{\sum_i^C TP_i}{|Dataset|}$$

$$F1 = \frac{TP}{TP+\frac{1}{2}(FP+FN)}$$

Micro averaged $$F1 = \frac{\sum_i^C TP_i}{\sum_i^C TP_i+\frac{1}{2}(\sum_i^C FP_i+\sum_i^C FN_i)}$$

*Assuming each observation has one label

Let's now see something fun

Q3.2 answered

$$\textit{(Micro)}\ F1 = \frac{\sum_i^C TP_i}{\sum_i^C TP_i + \frac{1}{2}(\sum_i^C FP_i + \sum_i^C FN_i)}$$

FPs

Actual \ Predicted	Airplane	Boat	Car
Airplane	2	1	0
Boat	0	1	0
Car	1	2	3

FNs

Actual \ Predicted	Airplane	Boat	Car
Airplane	2	1	0
Boat	0	1	0
Car	1	2	3

$$\textit{(Micro)}\ F1 = \frac{\sum_i^C TP_i}{\sum_i^C TP_i + \frac{1}{2}(\sum_i^C FP_i + \sum_i^C FN_i)} = \frac{\sum_i^C TP_i}{|Dataset|} = Accuracy$$

Explanation from stackexchange:
https://stats.stackexchange.com/questions/571716/why-true-positives-equals-to-true-negatives-while-calculating-micro-f1-or-accura

Questions so far?

1hr 25 mins

My own work:

Logical reasoning in NLI (a simplified introduction)

1hr 40 mins

Outline

1. NLP classification tasks
2. Naive Bayes
3. Logistic Regression
4. Neural Networks (NNs)
5. Recurrent neural networks (RNNS)
6. CNNs
7. Accuracy and F1
8. **Our recent research**

Natural Language Inference (NLI)

- **Premise**: The person just finished running the London marathon
- **Hypothesis**: They slowed down in the second half of the marathon

Label:

- **Entailment**: if the hypothesis is implied by the premise
- **Contradiction**: if the hypothesis contradicts the premise
- **Neutral**: otherwise

What are our logical atoms?

- Statements as logical atoms:

The frog is green
Green things can jump

A frog can jump

- Relationships between people could be logical atoms:

Homer - Bart (Son)
Bart - Lisa (Sister)

Homer - Lisa (Daughter)

- But what could our atoms be for NLI?

What are our logical atoms?

- Statements as logical atoms:

The frog is green
Green things can jump

A frog can jump

- Relationships between people could be logical atoms:

Homer - Bart (Son)
Bart - Lisa (Sister)

Homer - Lisa (Daughter)

- Our idea

Premise: *An old man with a package poses infront of an advertisement.*

Hypothesis: *A man poses in front of an ad for beer.*

Or another approach, using LLMs:

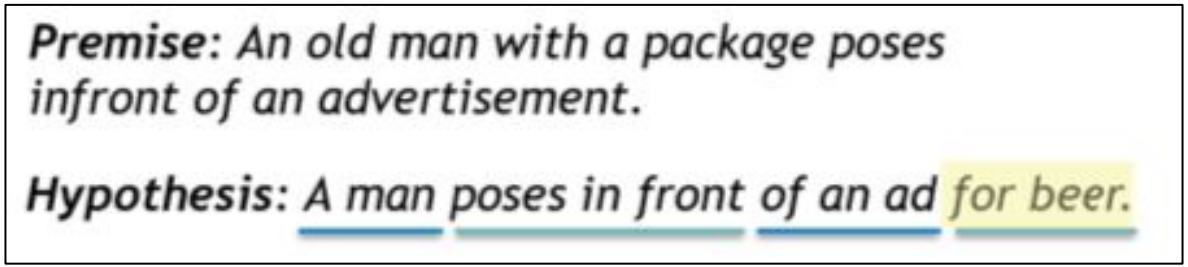
Ernest Jones is a British jeweller and watchmaker. Established in 1949, its first store was opened in Oxford Street, London. Ernest Jones specialises in diamonds and watches, stocking brands such as Gucci and Emporio Armani. Ernest Jones is part of the Signet Jewelers group.

Ask GPT-3 to decompose a text into a comprehensive set of facts

1. Ernest Jones is a British jeweller and watchmaker
2. Ernest Jones was established in 1949
3. **Ernest Jones opened its first store in Oxford Street, London in 1949**
4. Ernest Jones specialises in diamonds and watches
5. Ernest Jones is part of the Signet Jewelers group
6. Ernest Jones stocks brands such as Gucci and Emporio Armani

NLI with spans as our logical atoms

- Does it make sense to evaluate at a span-level?



- How do we train the model to have good span-level behaviour using only sentence labels?

Using spans as logical atoms...

- Does it make sense to evaluate at a span-level?

Premise: *An old man with a package poses infront of an advertisement.*

Hypothesis: *A man poses in front of an ad for beer.*

- How do we train the model to have good span-level behaviour using only sentence labels?

$$\mathcal{L}_n^{\text{Span}} = (\max_i(\widetilde{a}_{ni}) - y_n)^2$$

Logical rules underpin our system

Logical rules for evaluation:

Cont. spans:	*Neutral spans:*		*Sentence label:*
At least one		=>	Contradiction
None	At least one	=>	Neutral
None	None	=>	Entailment

Logical rules underpin our system

"It's like giving our model explainability super-powers"

Haim Dubossarsky

Premise: two woman are embracing while holding to go packages.

Hypothesis: the sisters are hugging goodbye while holding to go packages after just eating lunch.

Results

We improve interpretability
(with almost no loss in performance!)

Accuracy	SNLI	Δ
BERT (baseline)	90.77	
Feng et al. (2020)	81.2	*-9.57*
Wu et al. (2021)	84.53	*-6.24*
Feng et al. (2022)	87.8	*-2.97*
SLR-NLI	90.33	*-0.44*
SLR-NLI+esnli	**90.49**	***-0.28***

Results

We also improve model robustness when there is limited training data!

Model	In-Distribution		Out-of-Distribution				
	SNLI-dev	SNLI-test	SNLI-hard	MNLI-mis.	MNLI-mat.	SICK	HANS
Baseline	73.98	73.90	59.25	49.17	48.46	52.19	50.27
PoE	60.79	61.26	54.44	41.74	42.03	45.92	50.26
Reweight.	70.69	70.86	**59.83**	46.99	47.12	48.65	50.03
Conf Reg.	57.32	57.51	49.61	38.05	38.54	38.93	**50.84**
SLR-NLI-eSNLI	**74.22**	**74.05**	59.51	**57.05**†	**54.76**†	**52.23**	50.00

See our papers for more detail

Our paper currently under review:

Logical Reasoning for Natural Language Inference Using Generated Facts as Atoms

Joe Stacey[1], **Pasquale Minervini**[2], **Haim Dubossarsky**[3], **Oana-Maria Camburu**[4], **Marek Rei**[1]
[1]Imperial College London, [2]University of Edinburgh,
[3]Queen Mary University of London, [4]University College London
{j.stacey20, marek.rei}@imperial.ac.uk

See our EMNLP paper:

Logical Reasoning with Span-Level Predictions for Interpretable and Robust NLI Models

Joe Stacey
Imperial College London
j.stacey20@imperial.ac.uk

Pasquale Minervini
University of Edinburgh & UCL
p.minervini@ed.ac.uk

Haim Dubossarsky
Queen Mary University of London
h.dubossarsky@qmul.ac.uk

Marek Rei
Imperial College London
marek.rei@imperial.ac.uk

Questions so far?

1hr 45 mins

Appendix

1hr 50 mins

Other work I've done during my PhD:

PhD papers:

- Using human explanations to improve model robustness
- How knowledge distillation can create more robust models
- Understanding how and when bias mitigation techniques work
- Using LLM's to generate large-scale, accurate datasets
  (not yet publicly available)

Conditional independence question

- **Question 1**: Does conditional independence imply independence?
  - E.g: If P(A | C) and P(B | C) are independent,
    Are P(A) and P(B) independent?

Conditional independence question

- **Question 1**: Does conditional independence imply independence?
  - So P(A | C) and P(B | C) are independent in this example
  - But P(A) and P(B) are not

Example 1.27

A box contains two coins: a regular coin and one fake two-headed coin ($P(H) = 1$). I choose a coin at random and toss it twice. Define the following events.

- A= First coin toss results in an H.
- B= Second coin toss results in an H.
- C= Coin 1 (regular) has been selected.

Source:
https://www.probabilitycourse.com/chapter1/1_4_4_conditional_independence.php

Conditional independence question

- **Question 2**: Does independence imply conditional independence?
  - E.g: If P(A) and P(B) are independent,
    Are P(A|C) and p(B|C) independent?

Input representations

- **Question 2**: Does independence imply conditional independence?
  - E.g: If P(A) and P(B) are independent,
    Are P(A|C) and p(B|C) independent?

$$P(A \cap B) = P(A)P(B) \quad \textbf{(Eq.1)}$$

Consider rolling a dice, where:

A = {if 1 or 2 are rolled}
B = {if 2, 4, or 6 are rolled}
C = {if 1 or 4 are rolled}

Source:
https://www.probabilitycourse.com/chapter1/1_4_4_conditional_independence.php