Text Analysis with Generalised Linear Models

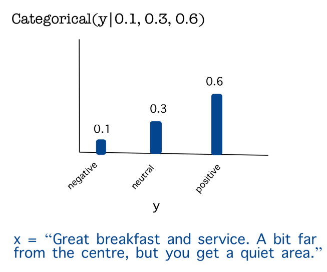
NTV 2026 / HC2b (v3)

Today's lecturer: Wilker Aziz w.aziz@uva.nl

HC2a highlights

In text classification, we want to condition on text x and predict a distribution for the *possible* values of the response variable Y.

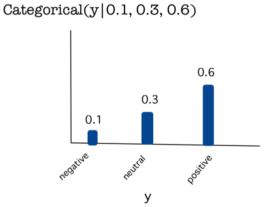
A first attempt at that proved intractable: simply hoping to estimate and store parameters for every x is impossible.



HC2a highlights

In text classification, we want to condition on text x and predict a distribution for the *possible* values of the response variable Y.

A first attempt at that proved intractable: simply hoping to estimate and store parameters for every x is impossible.



x = “Great breakfast and service. A bit far from the centre, but you get a quiet area.”

The naive Bayes approach: $P_{Y|X=x}$ is the result of probabilistic inference over a joint distribution $P_{YX}(y, w_{1:\ell}) = P_Y(y) \prod_{i=1}^{\ell} P_{W|Y}(w_i|y)$ built up from simpler CPDs (P_Y and $P_{W|Y}$).

NBC limitations: too simple view of the data

The NBC works with a rather simple view of the data (bag of words) and improving on that leads to intractabilities.

Improved NBC

We have a NB-like model of sentiment classification (with 3 classes: negative, neutral, and positive) that views documents as collections of bigrams.

Example: the datum $(Y = \text{neg}, X = \underline{\text{not very good feedback}})$ is assigned probability:

$$P_{YX}(\text{neg}, \underline{\text{not very good feedback}}) =$$

$$P_Y(\text{neg}) P_{F|Y}(\underline{\text{not very}}|\text{neg}) P_{F|Y}(\underline{\text{very good}}|\text{neg}) P_{F|Y}(\underline{\text{good feedback}}|\text{neg})$$

Here the random variable F is a pair of adjacent words (a bigram), and each word in the pair is drawn from a vocabulary of V known words.

NBC limitations: too simple view of the data

The NBC works with a rather simple view of the data (bag of words) and improving on that leads to intractabilities.

Improved NBC

We have a NB-like model of sentiment classification (with 3 classes: negative, neutral, and positive) that views documents as collections of bigrams.

Example: the datum $(Y = \text{neg}, X = \underline{\text{not very good feedback}})$ is assigned probability:

$$P_{YX}(\text{neg}, \underline{\text{not very good feedback}}) =$$

$$P_Y(\text{neg}) P_{F|Y}(\underline{\text{not very}}|\text{neg}) P_{F|Y}(\underline{\text{very good}}|\text{neg}) P_{F|Y}(\underline{\text{good feedback}}|\text{neg})$$

Here the random variable F is a pair of adjacent words (a bigram), and each word in the pair is drawn from a vocabulary of V known words.

Consider these two sentences (which classic NB cannot distinguish):

1. harsh reviewers gave us very good feedback
2. good reviewers gave us very harsh feedback

Can the 'improved NBC' make different posterior inferences for each sentence?

NBC limitations: too simple view of the data

The NBC works with a rather simple view of the data (bag of words) and improving on that leads to intractabilities.

Improved NBC

We have a NB-like model of sentiment classification (with 3 classes: negative, neutral, and positive) that views documents as collections of bigrams.

Example: the datum $(Y = \text{neg}, X = \underline{\text{not very good feedback}})$ is assigned probability:

$$P_{YX}(\text{neg}, \underline{\text{not very good feedback}}) =$$

$$P_Y(\text{neg})P_{F|Y}(\underline{\text{not very}}|\text{neg})P_{F|Y}(\underline{\text{very good}}|\text{neg})P_{F|Y}(\underline{\text{good feedback}}|\text{neg})$$

Here the random variable F is a pair of adjacent words (a bigram), and each word in the pair is drawn from a vocabulary of V known words.

Consider these two sentences (which classic NB cannot distinguish):

1. harsh reviewers gave us very good feedback
2. good reviewers gave us very harsh feedback

Can the ‘improved NBC’ make different posterior inferences for each sentence? Yes, because their ‘bags of bigrams’ are not the same!

NBC limitations: too simple view of the data

The NBC works with a rather simple view of the data (bag of words) and improving on that leads to intractabilities.

Improved NBC

We have a NB-like model of sentiment classification (with 3 classes: negative, neutral, and positive) that views documents as collections of bigrams.

Example: the datum $(Y = \text{neg}, X = \underline{\text{not very good feedback}})$ is assigned probability:

$$P_{YX}(\text{neg}, \underline{\text{not very good feedback}}) =$$

$$P_Y(\text{neg})P_{F|Y}(\underline{\text{not very}}|\text{neg})P_{F|Y}(\underline{\text{very good}}|\text{neg})P_{F|Y}(\underline{\text{good feedback}}|\text{neg})$$

Here the random variable F is a pair of adjacent words (a bigram), and each word in the pair is drawn from a vocabulary of V known words.

The change that granted this improved ability to detect differences in text has a ‘cost’ in terms of model size. Relate the size of this model (total size of its tabular CPDs) to that of the original NBC.

NBC limitations: too simple view of the data

The NBC works with a rather simple view of the data (bag of words) and improving on that leads to intractabilities.

Improved NBC

We have a NB-like model of sentiment classification (with 3 classes: negative, neutral, and positive) that views documents as collections of bigrams.

Example: the datum ($Y = \text{neg}, X = \underline{\text{not very good feedback}}$) is assigned probability:

$$P_{YX}(\text{neg}, \underline{\text{not very good feedback}}) =$$

$$P_Y(\text{neg})P_{F|Y}(\underline{\text{not very}}|\text{neg})P_{F|Y}(\underline{\text{very good}}|\text{neg})P_{F|Y}(\underline{\text{good feedback}}|\text{neg})$$

Here the random variable F is a pair of adjacent words (a bigram), and each word in the pair is drawn from a vocabulary of V known words.

The change that granted this improved ability to detect differences in text has a 'cost' in terms of model size. Relate the size of this model (total size of its tabular CPDs) to that of the original NBC. The original NBC has size $K + KV$, this 'improved NB' has size $K + KV^2$ because F can take on any of V^2 possible bigrams.

NBC limitations: too simple view of the data

The NBC works with a rather simple view of the data (bag of words) and improving on that leads to intractabilities.

Improved NBC

We have a NB-like model of sentiment classification (with 3 classes: negative, neutral, and positive) that views documents as collections of bigrams.

Example: the datum $(Y = \text{neg}, X = \underline{\text{not very good feedback}})$ is assigned probability:

$$P_{YX}(\text{neg}, \underline{\text{not very good feedback}}) =$$

$$P_Y(\text{neg}) P_{F|Y}(\underline{\text{not very}}|\text{neg}) P_{F|Y}(\underline{\text{very good}}|\text{neg}) P_{F|Y}(\underline{\text{good feedback}}|\text{neg})$$

Here the random variable F is a pair of adjacent words (a bigram), and each word in the pair is drawn from a vocabulary of V known words.

A growth in CPDs poses a storage challenge but also a statistical estimation challenge: some bigrams are unseen because they violate some principle of English (e.g., <u>dog exquisite</u>) others are just less frequent (e.g., <u>exquisite dog</u>), and it is hard for the model to tease these cases apart.

NBC limitations: too simple view of the data

The NBC works with a rather simple view of the data (bag of words) and improving on that leads to intractabilities.

Improved NBC

We have a NB-like model of sentiment classification (with 3 classes: negative, neutral, and positive) that views documents as collections of bigrams.

Example: the datum $(Y = \text{neg}, X = \underline{\text{not very good feedback}})$ is assigned probability:

$$P_{YX}(\text{neg}, \underline{\text{not very good feedback}}) = $$

$$P_Y(\text{neg})P_{F|Y}(\underline{\text{not very}}|\text{neg})P_{F|Y}(\underline{\text{very good}}|\text{neg})P_{F|Y}(\underline{\text{good feedback}}|\text{neg})$$

Here the random variable F is a pair of adjacent words (a bigram), and each word in the pair is drawn from a vocabulary of V known words.

Suppose we have enough data to estimate reliable CPDs and storage of large tables is not a problem for us. What problems remain?

NBC limitations: too simple view of the data

The NBC works with a rather simple view of the data (bag of words) and improving on that leads to intractabilities.

Improved NBC

We have a NB-like model of sentiment classification (with 3 classes: negative, neutral, and positive) that views documents as collections of bigrams.

Example: the datum $(Y = \text{neg}, X = \underline{\text{not very good feedback}})$ is assigned probability:

$$
\begin{aligned}
P_{YX}(\text{neg}, \underline{\text{not very good feedback}}) = \\
P_Y(\text{neg}) P_{F|Y}(\underline{\text{not very}}|\text{neg}) P_{F|Y}(\underline{\text{very good}}|\text{neg}) P_{F|Y}(\underline{\text{good feedback}}|\text{neg})
\end{aligned}
$$

Here the random variable F is a pair of adjacent words (a bigram), and each word in the pair is drawn from a vocabulary of V known words.

Suppose we have enough data to estimate reliable CPDs and storage of large tables is not a problem for us. What problems remain? Some texts might have the same bigrams in different order.
Sentiment-relevant information may stretch past 2 words (e.g., $\underline{\text{not even such expensive cast can salvage the plot}}$). And more...

NBC limitations: classification only

The response variable must be a class out of a countable and finite set of possibilities.

NBC limitations: classification only

The response variable must be a class out of a countable and finite set of possibilities.

That is because NBC uses a CPD of the kind $P_{W|Y}$.

If there are infinitely many possibilities for Y, a tabular view of $P_{W|Y}$ would grow infinitely large.

This excludes, for example, regression problems where we predict numerical variables like demand, number of thumbs-up, market value, voting intentions, etc.

So, what do we do?

NBC limitations: classification only

The response variable must be a class out of a countable and finite set of possibilities.

That is because NBC uses a CPD of the kind $P_{W|Y}$.

If there are infinitely many possibilities for Y, a tabular view of $P_{W|Y}$ would grow infinitely large.

This excludes, for example, regression problems where we predict numerical variables like demand, number of thumbs-up, market value, voting intentions, etc.

So, what do we do?

The key insight behind the NBC is this: if a direct relationship, e.g. $x \rightarrow P_{Y|X=x}$, is difficult to design directly, infer it indirectly.

Outline

This class develops tools for the analysis of text, in terms of **classification or regression**, based on a tool from statistics known as **generalised linear models** (GLMs).

Suggested reading:

- Lecture notes by Wilker
- or Chapter 4 of textbook (2026 version)

In terms of content and style, our lecture notes are the right resource. The textbook has a slower pace and has more examples, but it also goes in depth into some parts that we consider part of ML (and hence which we do not cover) and it skips over parts that are part of NTV. If you like the pace of the textbook, be careful to still reserve time for our lecture notes.

Goals

ILOs. After this class the student can

- recognise applications of text regression
- recognise the role of feature functions in countering data sparsity
- use generalised linear models to parameterise statistical models
- estimate parameters of the model via maximum likelihood estimation using a gradient-based procedure

Table of contents

1. Logistic CPDs

2. Generalised linear models

3. Text regression

4. Wrap-up

Logistic CPDs

Goal

Achieve something like the NB posterior, namely, a distribution over the response rv Y *given* an outcome x of the input rv X.

But do it **without having to prescribe** the joint distribution of X and Y.

Goal

Achieve something like the NB posterior, namely, a distribution over the response rv Y *given* an outcome x of the input rv X.

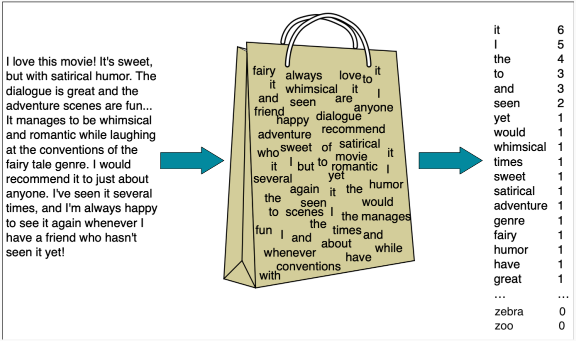
But do it **without having to prescribe** the joint distribution of X and Y.

To get there we need to develop

- a feature function
  to represent x in a numerical space
- and a parametric predictor of probability values
  to parameterise our CPD

Feature function

Define a finite set of rules to map any one given text to a point in the real coordinate space. Each dimension in this space can be thought of as a *feature* (an *aspect* or *property*) of the text that is worth representing in the context of our application.



I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!

it	6
I	5
the	4
to	3
and	3
seen	2
yet	1
would	1
whimsical	1
times	1
sweet	1
satirical	1
adventure	1
genre	1
fairy	1
humor	1
have	1
great	1
…	…
zebra	0
zoo	0

The V-dimensional **count vector** view of x represents the bag of words view of the text as a point in $\mathbb{R}^V$.

More examples

x	amazing	new	previous	terrible	version
the previous version was amazing					
the new version is terrible					

State the 5-dimensional representation of each text in the space whose coordinates capture the **number of occurrences** of the words shown in the column headers.

More examples

x	amazing	new	previous	terrible	version
the previous version was amazing	1	0	1	0	1
the new version is terrible	0	1	0	1	1

State the 5-dimensional representation of each text in the space whose coordinates capture the **number of occurrences** of the words shown in the column headers.

More examples

x	amazing	new	previous	terrible	version
the previous version was amazing	1	0	1	0	1
the new version is terrible	0	1	0	1	1

State the 5-dimensional representation of each text in the space whose coordinates capture the **number of occurrences** of the words shown in the column headers.

More on mentimeter...

Logistic CPD

Suppose we have a D-dimensional feature function $\mathsf{h}(\cdot)$ to represent texts as points in the real coordinate space. Let's exploit it to design a new type of parameterisation of the CPD $P_{Y|X}$.

We can **predict** the probability value $P(Y = y|X = x)$ of any one pair (x, y) under this CPD as follows:

- For each label $y \in \mathcal{Y}$, introduce

Logistic CPD

Suppose we have a D-dimensional feature function $\mathrm{h}(\cdot)$ to represent texts as points in the real coordinate space. Let's exploit it to design a new type of parameterisation of the CPD $P_{Y|X}$.

We can **predict** the probability value $P(Y = y|X = x)$ of any one pair (x, y) under this CPD as follows:

- For each label $y \in \mathcal{Y}$, introduce
    - a parameter vector $\mathrm{w}_y \in \mathbb{R}^D$, its coordinates capture something about the correlation between y and each feature

Logistic CPD

Suppose we have a D-dimensional feature function $\mathbf{h}(\cdot)$ to represent texts as points in the real coordinate space. Let's exploit it to design a new type of parameterisation of the CPD $P_{Y|X}$.

We can **predict** the probability value $P(Y = y|X = x)$ of any one pair (x, y) under this CPD as follows:

- For each label $y \in \mathcal{Y}$, introduce
    - a parameter vector $\mathbf{w}_y \in \mathbb{R}^D$, its coordinates capture something about the correlation between y and each feature
    - a scalar parameter $b_y \in \mathbb{R}$ to capture something about the marginal relevance of y for the analysis of our dataset

Logistic CPD

Suppose we have a D-dimensional feature function $\mathrm{h}(\cdot)$ to represent texts as points in the real coordinate space. Let's exploit it to design a new type of parameterisation of the CPD $P_{Y|X}$.

We can **predict** the probability value $P(Y = y|X = x)$ of any one pair (x, y) under this CPD as follows:

- For each label $y \in \mathcal{Y}$, introduce
  - a parameter vector $\mathrm{w}_y \in \mathbb{R}^D$, its coordinates capture something about the correlation between y and each feature
  - a scalar parameter $b_y \in \mathbb{R}$ to capture something about the marginal relevance of y for the analysis of our dataset
- Given x, let $s_y = \mathrm{w}_y^\top \mathrm{h}(x) + b_y$ associate a score with label y

Logistic CPD

Suppose we have a D-dimensional feature function $\mathbf{h}(\cdot)$ to represent texts as points in the real coordinate space. Let's exploit it to design a new type of parameterisation of the CPD $P_{Y|X}$.

We can **predict** the probability value $P(Y = y|X = x)$ of any one pair (x, y) under this CPD as follows:

- For each label $y \in \mathcal{Y}$, introduce
    - a parameter vector $\mathbf{w}_y \in \mathbb{R}^D$, its coordinates capture something about the correlation between y and each feature
    - a scalar parameter $b_y \in \mathbb{R}$ to capture something about the marginal relevance of y for the analysis of our dataset
- Given x, let $s_y = \mathbf{w}_y^\top \mathbf{h}(x) + b_y$ associate a score with label y
- Then define $P(Y = y|X = x) \overset{\text{def.}}{=} \frac{\exp(s_y)}{\sum_{k=1}^{K} \exp(s_k)}$

Logistic CPD: example

Let's try this with 3 classes (negative, neutral and positive), the feature space and parameters shown below:

y	$\mathbf{w}_y$					b_y
	amazing	new	previous	terrible	version	
negative	0	0	-0.5	1	0	0
neutral	0	0.1	0.1	0	0.5	0
positive	1	0.5	0	0	0	0

x	$\mathbf{h}(x)$				
	amazing	new	previous	terrible	version
the previous version was amazing	1	0	1	0	1
the new version is terrible	0	1	0	1	1

Compute the class probabilities under $P_{Y|X=x}$ where x is the first document.

Logistic CPD: example

Let's try this with 3 classes (negative, neutral and positive), the feature space and parameters shown below:

y	$\mathbf{w}_y$					b_y
	amazing	new	previous	terrible	version	
negative	0	0	-0.5	1	0	0
neutral	0	0.1	0.1	0	0.5	0
positive	1	0.5	0	0	0	0

x	$\mathbf{h}(x)$				
	amazing	new	previous	terrible	version
the previous version was amazing	1	0	1	0	1
the new version is terrible	0	1	0	1	1

Compute the class probabilities under $P_{Y|X=x}$ where x is the first document. For negative: $\frac{\exp(-0.5)}{\exp(-0.5)+\exp(0.1+0.5)+\exp(1)} \approx 0.12$

Logistic CPD: example

Let's try this with 3 classes (negative, neutral and positive), the feature space and parameters shown below:

y	$\mathbf{w}_y$					b_y
	amazing	new	previous	terrible	version	
negative	0	0	-0.5	1	0	0
neutral	0	0.1	0.1	0	0.5	0
positive	1	0.5	0	0	0	0

x	$\mathbf{h}(x)$				
	amazing	new	previous	terrible	version
the previous version was amazing	1	0	1	0	1
the new version is terrible	0	1	0	1	1

Compute the class probabilities under $P_{Y|X=x}$ where x is the first document. For neutral: $\frac{\exp(0.6)}{\exp(-0.5)+\exp(0.1+0.5)+\exp(1)} \approx 0.35$

Logistic CPD: example

Let's try this with 3 classes (negative, neutral and positive), the feature space and parameters shown below:

y	$\mathbf{w}_y$					b_y
	amazing	new	previous	terrible	version	
negative	0	0	-0.5	1	0	0
neutral	0	0.1	0.1	0	0.5	0
positive	1	0.5	0	0	0	0

x	$\mathbf{h}(x)$				
	amazing	new	previous	terrible	version
the previous version was amazing	1	0	1	0	1
the new version is terrible	0	1	0	1	1

Compute the class probabilities under $P_{Y|X=x}$ where x is the first document. For positive: $\frac{\exp(1)}{\exp(-0.5)+\exp(0.1+0.5)+\exp(1)} \approx 0.53$

Logistic CPD: example

Let's try this with 3 classes (negative, neutral and positive), the feature space and parameters shown below:

y	$\mathbf{w}_y$					b_y
	amazing	new	previous	terrible	version	
negative	0	0	-0.5	1	0	0
neutral	0	0.1	0.1	0	0.5	0
positive	1	0.5	0	0	0	0

x	$\mathbf{h}(x)$				
	amazing	new	previous	terrible	version
the previous version was amazing	1	0	1	0	1
the new version is terrible	0	1	0	1	1

Compute the class probabilities under $P_{Y|X=x}$ where x is the first document. That is, $(0.12, 0.35, 0.53)$

What just happened?

Let’s compare what we did to the naive Bayes model from last module.

Instead of modelling the distribution of paired data $P(X, Y)$ and inferring $P_{Y|X=x}$ for a given x via Bayes rule, we created a parametric rule to map a given **vector-valued representation** of x to the *parameters* of a conditional Categorical distribution over classes given x.

The resulting model does not know how to generate any aspect of x. Models of this kind are typically called **discriminative**, whereas models like NB are typically called **generative**.

Logistic CPD $P(Y|X)$

For K possible output classes and any one input feature vector $\mathbf{h}(x)$, we perform K transformations of the input:

- $s_k = \mathbf{w}_k^\top \mathbf{h}(x) + b_k$ *the result is a scalar score per class*

with $\mathbf{w}_k \in \mathbb{R}^D$ and $b_k \in \mathbb{R}$ being class-specific parameters.

Logistic CPD $P(Y|X)$

For K possible output classes and any one input feature vector $\mathbf{h}(x)$, we perform K transformations of the input:

- $s_k = \mathbf{w}_k^\top \mathbf{h}(x) + b_k$ *the result is a scalar score per class*

with $\mathbf{w}_k \in \mathbb{R}^D$ and $b_k \in \mathbb{R}$ being class-specific parameters.

Or, equivalently, in matrix form:

- $\mathbf{s} = \mathbf{W}\mathbf{h}(x) + \mathbf{b}$ *the result is a vector of K scores*

with $\mathbf{W} \in \mathbb{R}^{K \times D}$, $\mathbf{b} \in \mathbb{R}^K$.

Logistic CPD $P(Y|X)$

For K possible output classes and any one input feature vector $\mathbf{h}(x)$, we perform K transformations of the input:

- $s_k = \mathbf{w}_k^\top \mathbf{h}(x) + b_k$ *the result is a scalar score per class*

with $\mathbf{w}_k \in \mathbb{R}^D$ and $b_k \in \mathbb{R}$ being class-specific parameters.

Or, equivalently, in matrix form:

- $\mathbf{s} = \mathbf{W}\mathbf{h}(x) + \mathbf{b}$ *the result is a vector of K scores*

with $\mathbf{W} \in \mathbb{R}^{K \times D}$, $\mathbf{b} \in \mathbb{R}^K$.

We then turn the scores $(s_1, \ldots, s_K)$ into probabilities using the **softmax activation function**: $\boldsymbol{\pi} = \mathrm{softmax}(\mathbf{s})$ where

$$\boldsymbol{\pi} = \left(\frac{\exp(s_1)}{\sum_{k=1}^{K} \exp(s_k)}, \ldots, \frac{\exp(s_K)}{\sum_{k=1}^{K} \exp(s_k)} \right)^\top$$

Logistic CPD $P(Y|X)$

For K possible output classes and any one input feature vector $\mathbf{h}(x)$, we perform K transformations of the input:

- $s_k = \mathbf{w}_k^\top \mathbf{h}(x) + b_k$ *the result is a scalar score per class*

with $\mathbf{w}_k \in \mathbb{R}^D$ and $b_k \in \mathbb{R}$ being class-specific parameters.

Or, equivalently, in matrix form:

- $\mathbf{s} = \mathbf{W}\mathbf{h}(x) + \mathbf{b}$ *the result is a vector of K scores*

with $\mathbf{W} \in \mathbb{R}^{K \times D}$, $\mathbf{b} \in \mathbb{R}^K$.

We then turn the scores $(s_1, \ldots, s_K)$ into probabilities using the **softmax activation function**: $\boldsymbol{\pi} = \mathrm{softmax}(\mathbf{s})$ where

$$\boldsymbol{\pi} = \left(\frac{\exp(s_1)}{\sum_{k=1}^K \exp(s_k)}, \ldots, \frac{\exp(s_K)}{\sum_{k=1}^K \exp(s_k)} \right)^\top$$

The distribution of Y given $X = x$ is then $\mathrm{Categorical}(\boldsymbol{\pi})$.

Generalised linear models

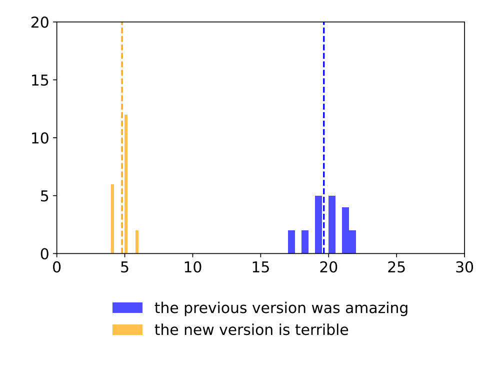
GLM

A GLM is a conditional model of $Y|X = x$ prescribed by mapping x to the K statistical parameter(s) of a pmf/pdf of choice.

- the model has its own trainable/free parameters $\boldsymbol{\theta}$ (weights and biases)
- $\boldsymbol{\theta}$ interacts linearly with $\mathrm{h}(x)$ to produce the K **linear predictor(s)**
- finally, use (non-linear) activation function(s) to make sure that for any $\mathrm{h}(x)$ and any choice of trainable parameters, the model predicts a valid parameterisation of the pmf/pdf.

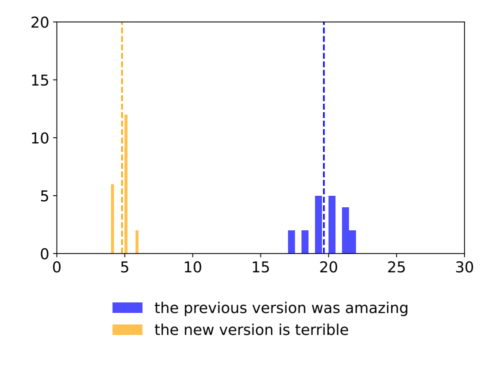
Example: numerical response

We are now working with the response variable *'number of thumbs up'*. To decide on the kind of distribution, we ran a small experiment aimed at revealing some statistical properties of this variable: we asked 20 independent raters to assign a number of thumbs up, given the review text.



Example: numerical response

We are now working with the response variable *'number of thumbs up'*. To decide on the kind of distribution, we ran a small experiment aimed at revealing some statistical properties of this variable: we asked 20 independent raters to assign a number of thumbs up, given the review text.



Ordinal data; as the mean grows, so does the variance. Let's model with Poisson.

GLM - Single parameter case

Poisson

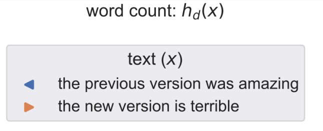
Example for ordinal Y using the Poisson distribution:

$$Y|X = x \sim \mathsf{Poisson}(\lambda(x; \boldsymbol{\theta}))$$

- Map $\mathsf{h}(x)$ to a single linear predictor
    - $s = \mathsf{w}^\top \mathsf{h}(x) + b$ with $\mathsf{w} \in \mathbb{R}^D$ and $b \in \mathbb{R}$
- Constrain s to be a valid parameter:
    - $\lambda(x; \boldsymbol{\theta}) = \exp(s)$ *other activations are also possible*
      because the Poisson rate must be strictly positive

$D + 1$ trainable parameters: $\boldsymbol{\theta} = \{\mathsf{w}, b\}$.

Poisson GLM


word count: $h_d(x)$
text (x)
◀ the previous version was amazing
▶ the new version is terrible

x	h(x)				
	amazing	new	previous	terrible	version
the previous version was amazing	1	0	1	0	1
the new version is terrible	0	1	0	1	1

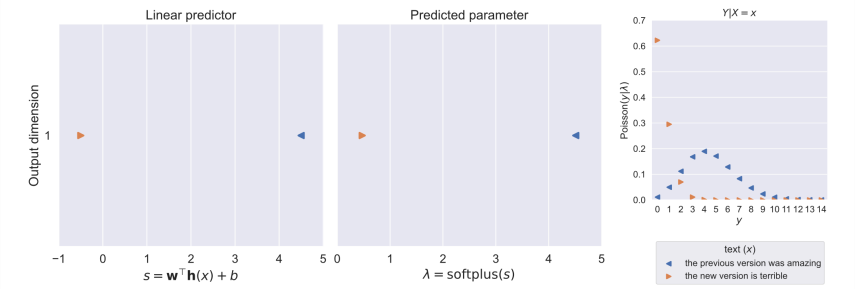
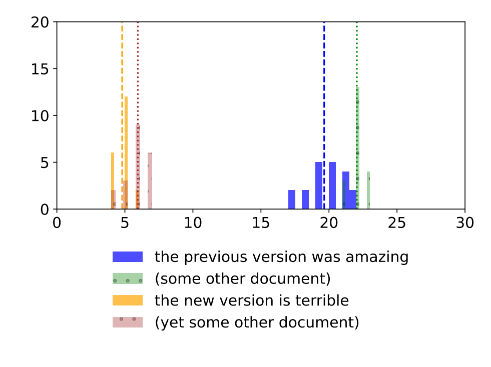


Figure 4.6: Here we show the linear predictor (left) and the Poisson rate (centre) that the GLM predicts for two pieces of text, each represented using the feature function from Figure 4.2. In this example, the GLM parameters are $\mathbf{w} = (2, -1, 0, -2, -1)^\top$ and $b = 3/2$. Using the GLM output (centre) we can prescribe the distribution of Y given $X = x^{(1)}$ or given $X = x^{(2)}$ via the Poisson pmf (right).

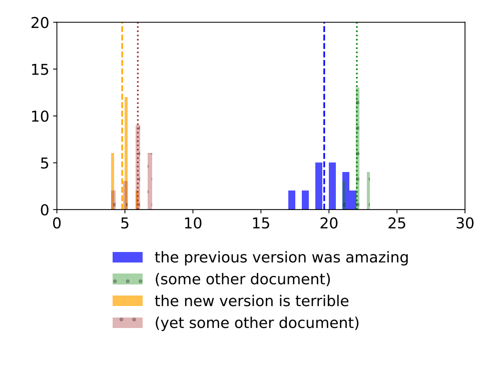
Example: numerical response (revisited)

After using the model for some time, we noticed some problems and now suspect the Poisson might not have been a good choice. To find evidence, we ran the previous experiment with a few more reviews.



Example: numerical response (revisited)

After using the model for some time, we noticed some problems and now suspect the Poisson might not have been a good choice. To find evidence, we ran the previous experiment with a few more reviews.



It looks like cases that have similar means do not always have similar variances (the Poisson cannot do that). Let’s try a two-parameter distribution.

GLM - Two parameters case

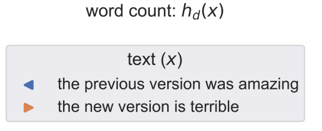
Example for real Y using the Normal/Gaussian distribution:

$$Y|X = x \sim \mathcal{N}(\mu(x;\boldsymbol{\theta}), \sigma^2(x;\boldsymbol{\theta}))$$

- Map $\mathbf{h}(x)$ to two linear predictors:
  - $s_1 = \mathbf{w}_1^\top \mathbf{h}(x) + b_1$ with $\mathbf{w}_1 \in \mathbb{R}^D$ and $b_1 \in \mathbb{R}$
  - $s_2 = \mathbf{w}_2^\top \mathbf{h}(x) + b_2$ with $\mathbf{w}_2 \in \mathbb{R}^D$ and $b_2 \in \mathbb{R}$
- Map linear predictors to valid parameters:
  - $\mu(x;\boldsymbol{\theta}) = s_1$
    because the Normal location is a real number
  - $\sigma(x;\boldsymbol{\theta}) = \exp(s_2)$ *other activations are also possible*
    because the Normal scale must be strictly positive

$2D + 2$ trainable parameters: $\boldsymbol{\theta} = \{\mathbf{w}_1, \mathbf{w}_2, b_1, b_2\}$.

Normal GLM



word count: $h_d(x)$

text (x)

◀ the previous version was amazing

▶ the new version is terrible

x	h(x)				
	amazing	new	previous	terrible	version
the previous version was amazing	1	0	1	0	1
the new version is terrible	0	1	0	1	1

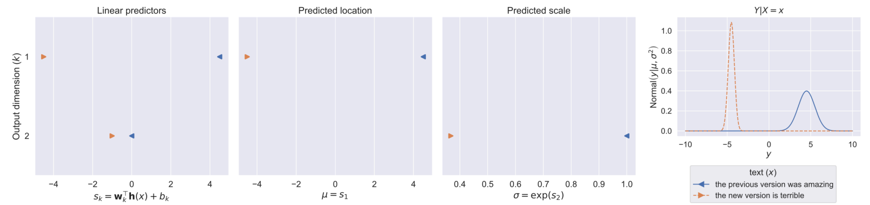


Figure 4.7: Here we show the linear predictors (left) and the Normal parameters (location and scale) that the GLM predicts for two pieces of text, each represented using the feature function from Figure 4.2. In this example, the GLM parameters are $\mathbf{W} = (\mathbf{w}_1, \mathbf{w}_2)^\top$ with $\mathbf{w}_1 = (2, -5, 0, -2, 1)^\top$, $\mathbf{w}_2 = (-1, -1, 0, -1, 1)^\top$ and $\mathbf{b} = (3/2, 0)^\top$. Using the GLM output (centre) we can prescribe the distribution of Y given $X = x^{(1)}$ or given $X = x^{(2)}$ via the Normal pdf (right).

GLM - Multiple parameters case

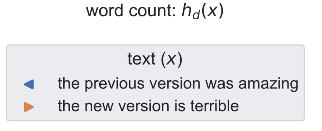
Example for nominal Y (one of K):

$$Y|X = x \sim \mathsf{Categorical}(\pi_1(x;\boldsymbol{\theta}), \ldots, \pi_K(x;\boldsymbol{\theta}))$$

- Map $\mathsf{h}(x)$ to K linear predictors:
  - $\mathsf{s} = \mathsf{W}\mathsf{h}(x) + \mathsf{b}$ with $\mathsf{W} \in \mathbb{R}^{K \times D}$ and $\mathsf{b} \in \mathbb{R}^{K}$
- Map linear predictors to valid parameters:
  - $\boldsymbol{\pi}(x;\boldsymbol{\theta}) = \mathsf{softmax}(\mathsf{s})$
    because the Categorical parameter must be a probability vector

$KD + K$ trainable parameters: $\boldsymbol{\theta} = \{\mathsf{W}, \mathsf{b}\}$.

Categorical GLM



word count: $h_d(x)$

text (x)

◀ the previous version was amazing

▶ the new version is terrible

x	h(x)				
	amazing	new	previous	terrible	version
the previous version was amazing	1	0	1	0	1
the new version is terrible	0	1	0	1	1

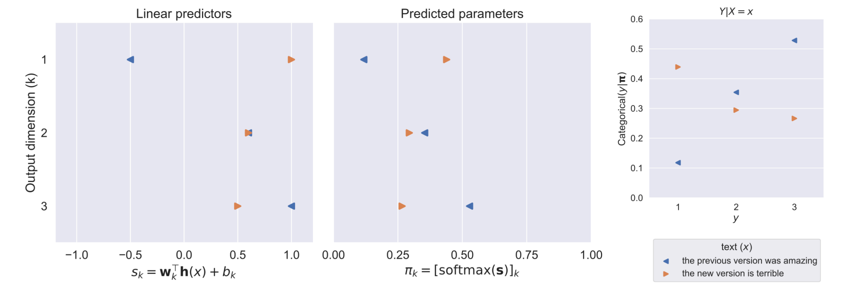


Figure 4.5: Here we show the K linear predictors (left) and the K Categorical parameters (centre) that the GLM predicts for two pieces of text, each represented using the feature function from Figure 4.2. In this example, the GLM parameters are $\mathbf{W} = (\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3)^\top$ with $\mathbf{w}_1 = (0, 0, -1/2, 1, 0)^\top$, $\mathbf{w}_2 = (0, 1/10, 1/10, 0, 1/2)^\top$, $\mathbf{w}_3 = (1, 1/2, 0, 0, 0)^\top$ and $\mathbf{b} = (0, 0, 0)^\top$. Using the GLM output (centre) we can prescribe the distribution of Y given $X = x^{(1)}$ or given $X = x^{(2)}$ via the Categorical pmf (right).

GLM - Other Distributions

- Bernoulli: for binary classification
- Skellam: sometimes right for ordinal data with varying location and scale;
- Student: sometimes right for continuous values whose distributions are unimodal, symmetric and heavier-tailed than the Normal;
- Gamma: sometimes right for non-negative continuous values, whose distributions are unimodal and show skewness;
- Beta: sometimes right for quantities between 0 and 1 (such as ratios);
- Dirichlet: sometimes right for vectors of ratios;

And the list goes on...

Making predictions

Most common strategies for **outcome prediction**

- Discrete response rv
    - most probable outcome
- Continuous response rv
    - mean (common for unimodal, symmetric distributions)
    - mode or median (common for skewed distributions)

Making predictions

Most common strategies for **outcome prediction**

- Discrete response rv
  - most probable outcome
- Continuous response rv
  - mean (common for unimodal, symmetric distributions)
  - mode or median (common for skewed distributions)

Common strategies for **uncertainty quantification**

- Discrete: Shannon entropy
- Continuous: variance

Parameter estimation

Regularised MLE[2]

$$\boldsymbol{\theta}^{\star} = \operatorname*{argmax}_{\boldsymbol{\theta}} \; \mathcal{L}(\boldsymbol{\theta}|\mathcal{D}) - \lambda \mathcal{R}(\boldsymbol{\theta})$$

via gradient-based optimisation

- Start with some randomly initialised $\boldsymbol{\theta}^{(0)}$
- Until convergence, perform iterations of the kind:

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} + \gamma_t \nabla_{\boldsymbol{\theta}^{(t)}} \mathcal{L}(\boldsymbol{\theta}^{(t)}|\mathcal{D})$$

In this course we do not compute gradients by hand, we use automatic differentiation software. Meaning, if we can compute $\mathcal{L}(\boldsymbol{\theta}|\mathcal{B})$ for a subsampled batch $\mathcal{B}$ of $\mathcal{D}$, software will give us the gradient vector for the update rule.

[2] Common regulariser: L2 penalty on parameter vector.

Text regression

Text regression

In many situations, we want to predict numerical quantities from text.

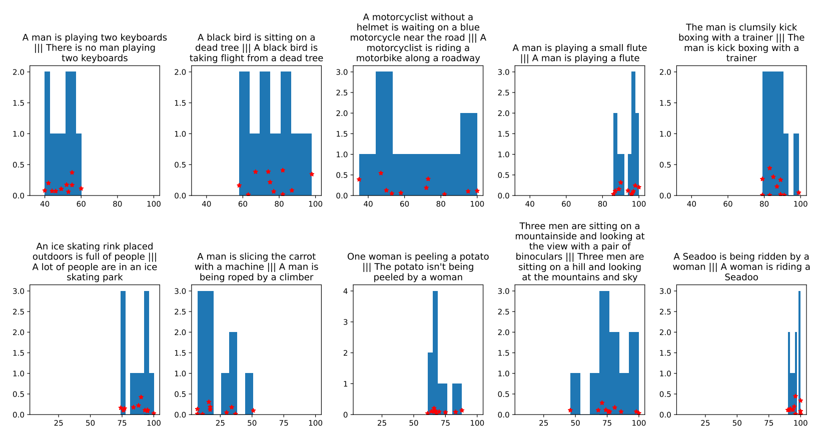
Here are some examples:

- Use social media data to predict attendance at large public event (e.g., pride in Ams);
- Use reviews to predict demand for products;
- Analyse a translation and quantify its quality (e.g., percentage of errors)
- Analyse the news and predict voting intentions

While some of these *can* be handled by some form of categorisation scheme (e.g., demand could be ‘low’, ‘regular’, ‘high’), others are more naturally expressed in numerical spaces (e.g., metrics of quality/utility, proportion of votes, etc.).

Semantic text similarity (from T2)

Semantic text similarity is a task where people are asked to assess the extent to which two texts $x = (a, b)$, text a and text b, convey the same meaning. We gather responses by asking participants to assign a score, in this example, from 0 to 100.[3]



A man is playing two keyboards ||| There is no man playing two keyboards

A black bird is sitting on a dead tree ||| A black bird is taking flight from a dead tree

A motorcyclist without a helmet is waiting on a blue motorcycle near the road ||| A motorcyclist is riding a motorbike along a roadway

A man is playing a small flute ||| A man is playing a flute

The man is clumsily kick boxing with a trainer ||| The man is kick boxing with a trainer

An ice skating rink placed outdoors is full of people ||| A lot of people are in an ice skating park

A man is slicing the carrot with a machine ||| A man is being roped by a climber

One woman is peeling a potato ||| The potato isn't being peeled by a woman

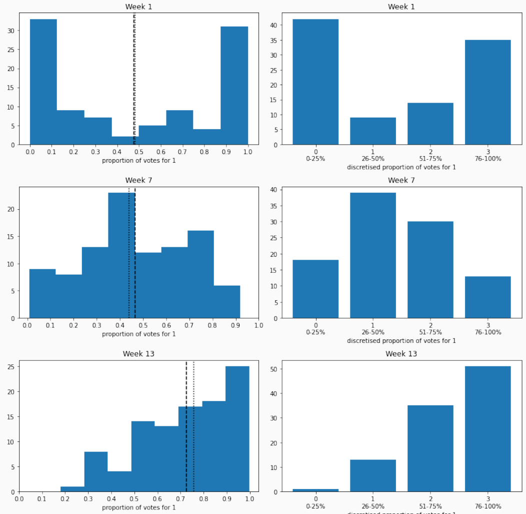
Three men are sitting on a mountainside and looking at the view with a pair of binoculars ||| Three men are sitting on a hill and looking at the mountains and sky

A Seadoo is being ridden by a woman ||| A woman is riding a Seadoo

[3] In T2, we provide data based on the SICK dataset.

Voting intention (E3)

Each week we observe the proportion of votes for candidate 1 (out of 2 candidates) in each of 100 states. We also gather relevant tweets in that same window. [Artificial data created for the exercise]



Exercise

https://github.com/probabll/ntmi-tutorials/blob/main/slides/GLMs-exercises.pdf
(or click here to download it)

Wrap-up

Limitations

The power of a GLM depends crucially on the availability of informative features.

Designing a good feature function can be difficult:

- Do we know what features matter?
- When we do, are we able to detect them automatically?
- If we can, is the resulting feature space at all manageable?

Limitations

The power of a GLM depends crucially on the availability of informative features.

Designing a good feature function can be difficult:

- Do we know what features matter?
- When we do, are we able to detect them automatically?
- If we can, is the resulting feature space at all manageable?

Often

- knowing what feature makes a difference requires expert knowledge (e.g., what aspects of language and language use influence *reading difficulty*?)
- detecting advanced features can be difficulty (e.g., how would you detect the use of *passive voice automatically?*
- features spaces can grow very large, esp when features are expressed as ‘presence’ of specific word sequences.

Nothing is *“just a tool”*

Text classification and regression are tools that allow us to analyse what people are writing (and to an extent thinking) about.

While text analysis can be used to guess categorical attributes and numerical quantities that inform human decision-making in some helpful settings (e.g., use indicators of reading complexity to improve assignments, **there is a daunting number of ways in which this can cause real harm to people**.

Tools of this kind can be used to breach the privacy of individuals, to monitor individuals and groups without their consent, to cheaply automate decisions that should arguably be made by people, etc.

Technology developers and those deploying technology for (semi-)automated decision making are responsible for the harms that afflict those who are directly or indirectly affected by the decisions.

Ethics in NLP

This course introduces you to technical devices that are part of the toolbox for text analysis.

Anything that interacts with people or human decision making **cannot** be developed or applied without *careful ethical considerations*.

Required reading: *https://aclanthology.org/2020.acl-tutorials.2.pdf*

We would like you to start thinking about broader impact of technology. Please reserve time for the reading, and check the related exercise in E2.

Summary

GLMs offer an alternative to NBC based on discriminative (rather than generative) modelling.

With GLMs we can model non-categorical responses.

GLMs make use of a feature function

- This is an opportunity, since we may be able to engineer informative features.
- But this is also rather difficult to do without expert knowledge.

What next?

T2 packs 2 GLMs, one for text classification and one for text regression.

T2 teaches you how to use autodiff to implement a simple stochastic gradient-based optimiser.

E2 has another exercise on GLMs.

Next module: **feature learning** to obtain highly effective/discriminative and yet very compact feature functions!

References i

References