Precept 2: Classification

COS 484

Simon Park, Tyler Zhu (slides borrowed heavily from Austin Wang, COS 324)
2/7/2025

Review Question

You train an n-gram model on some training corpus D using counts $P(w_n \mid w_1, \ldots, w_{n-1}) = \dfrac{c(w_1, \ldots, w_n)}{\sum_{v \in V} c(w_1, \ldots, w_{n-1}, v)}$. To prevent (possible) infinite perplexity on the test corpus D_t, you apply Laplace smoothing. Let $P(D), P(D_t)$ be the unsmoothed probabilities and $P'(D), P'(D_t)$ be the smoothed probabilities.

Review Question

You train an n-gram model on some training corpus D using counts
$P(w_n \mid w_1, \ldots, w_{n-1}) = \dfrac{c(w_1, \ldots, w_n)}{c(w_1, \ldots, w_{n-1})}$. To prevent (possible) infinite perplexity on the test corpus D_t, you apply Laplace smoothing. Let $ppl(D), ppl(D_t)$ be perplexities of the unsmoothed model and $ppl'(D), ppl'(D_t)$ of the smoothed model. Is the following T, F or undetermined (depends on model, data, n, etc)?

1. $ppl'(D) \geq ppl(D)$

2. $ppl'(D_t) < ppl(D_t)$

3. $ppl(D) < ppl(D_t)$

Review Question

You train an n-gram model on some training corpus D using counts $P(w_n \mid w_1, \ldots, w_{n-1}) = \dfrac{c(w_1, \ldots, w_n)}{c(w_1, \ldots, w_{n-1})}$. To prevent (possible) infinite perplexity on the test corpus D_t, you apply Laplace smoothing. Let $ppl(D), ppl(D_t)$ be perplexities of the unsmoothed model and $ppl'(D), ppl'(D_t)$ of the smoothed model. Is the following T, F or undetermined (depends on model, data, n, etc)?

1. $ppl'(D) \geq ppl(D)$
   This is true! Remember that setting the probability using counts (above) is the MLE estimate, which means that $P(D)$ cannot increase under any other distribution for $P(w_n \mid w_1, \ldots, w_{n-1})$

2. $ppl'(D_t) < ppl(D_t)$

3. $ppl(D) < ppl(D_t)$

Review Question

You train an n-gram model on some training corpus D using counts $P(w_n \mid w_1, \ldots, w_{n-1}) = \dfrac{c(w_1, \ldots, w_n)}{c(w_1, \ldots, w_{n-1})}$. To prevent (possible) infinite perplexity on the test corpus D_t, you apply Laplace smoothing. Let $ppl(D), ppl(D_t)$ be perplexities of the unsmoothed model and $ppl'(D), ppl'(D_t)$ of the smoothed model. Is the following T, F or undetermined (depends on model, data, n, etc)?

1. $ppl'(D) \geq ppl(D)$ - T

2. $ppl'(D_t) < ppl(D_t)$

   This is undetermined! It's not clear that the test corpus will have infinite perplexity. It is possible that the test corpus is very similar to the train corpus, and smoothing will cause its probability to drop.

3. $ppl(D) < ppl(D_t)$

Review Question

You train an n-gram model on some training corpus D using counts $P(w_n \mid w_1, \ldots, w_{n-1}) = \dfrac{c(w_1, \ldots, w_n)}{c(w_1, \ldots, w_{n-1})}$. To prevent (possible) infinite perplexity on the test corpus D_t, you apply Laplace smoothing. Let $ppl(D), ppl(D_t)$ be perplexities of the unsmoothed model and $ppl'(D), ppl'(D_t)$ of the smoothed model. Is the following T, F or undetermined (depends on model, data, n, etc)?

1. $ppl'(D) \geq ppl(D)$ - T

2. $ppl'(D_t) < ppl(D_t)$ - U

3. $ppl(D) < ppl(D_t)$
   Undetermined. A test corpus consisting solely of high-frequency n-grams might have a higher probability

Todays Topics

Given a document $d = w_1, \ldots, w_K$ and a set of classes $\mathscr{C} = \{c_1, \ldots, c_m\}$, we want to find the class c_i that maximizes $P(c \mid d)$. Two ways to do this:

Naive Bayes $<-$ Covered in lecture in great detail!

Logistic Regression

Todays Topics

Given a document $d = w_1, \ldots, w_K$ and a set of classes $\mathscr{C} = \{c_1, \ldots, c_m\}$, we want to find the class c_i that maximizes $P(c \mid d)$. Two ways to do this:

Naive Bayes

Logistic Regression $<-$ Focus for today's Precept

Logistic Regression: Intuition

Given a document $d = w_1, \ldots, w_K$ and a set of classes $\mathscr{C} = \{c_1, \ldots, c_m\}$, we want to find the class c_i that maximizes $P(c \mid d)$

Logistic Regression: Intuition

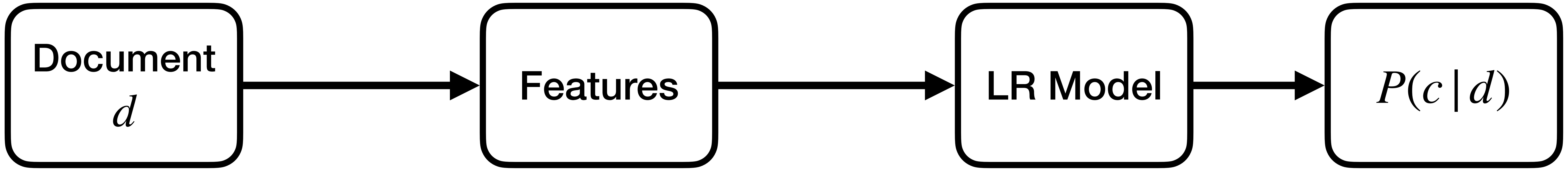
Given a document $d = w_1, \ldots, w_K$ and a set of classes $\mathscr{C} = \{c_1, \ldots, c_m\}$, we want to find the class c_i that maximizes $P(c \mid d)$

Compared to NB, with LR we take a more direct approach: directly compute $P(c \mid d)$ given a set of features constructed from the input document d.

Logistic Regression: Intuition

Given a document $d = w_1, \ldots, w_K$ and a set of classes $\mathscr{C} = \{c_1, \ldots, c_m\}$, we want to find the class c_i that maximizes $P(c \mid d)$

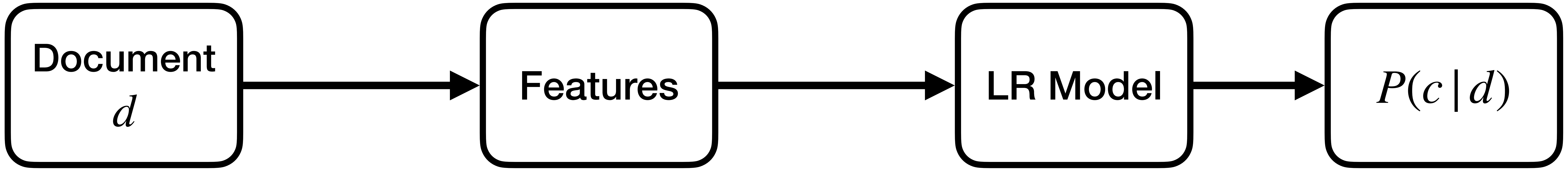
Compared to NB, with LR we take a more direct approach: directly compute $P(c \mid d)$ given a set of features constructed from the input document d.



Logistic Regression: Features

Given a document $d = w_1, \ldots, w_K$ and a set of classes $\mathscr{C} = \{c_1, \ldots, c_m\}$, we want to find the class c_i that maximizes $P(c \mid d)$

Compared to NB, with LR we take a more direct approach: directly compute $P(c \mid d)$ given a set of features constructed from the input document d.



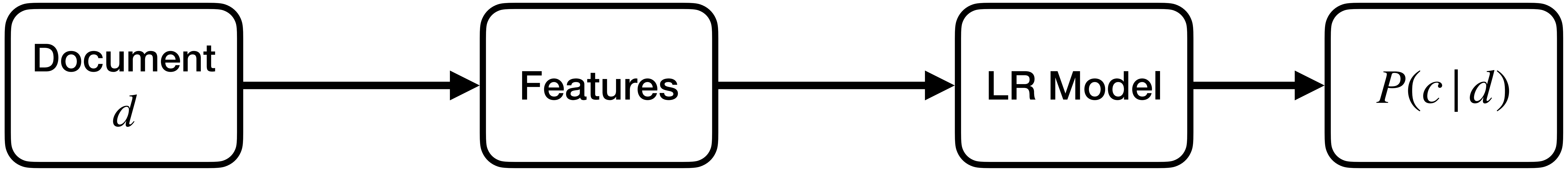
Var	Definition	Value
x_1	count(positive lexicon) $\in$ doc)	3
x_2	count(negative lexicon) $\in$ doc)	2
x_3	$\begin{cases} 1 & \text{if “no”} \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$	1
x_4	count(1st and 2nd pronouns $\in$ doc)	3
x_5	$\begin{cases} 1 & \text{if “!”} \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$	0
x_6	log(word count of doc)	$\ln(64) = 4.15$

This is the feature vector x for some input document d

Logistic Regression: Features

Given a document $d = w_1, \ldots, w_K$ and a set of classes $\mathscr{C} = \{c_1, \ldots, c_m\}$, we want to find the class c_i that maximizes $P(c \mid d)$

Compared to NB, with LR we take a more direct approach: directly compute $P(c \mid d)$ given a set of features constructed from the input document d.



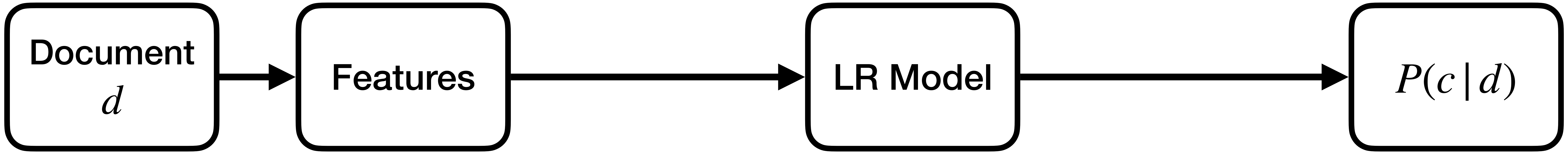
Var	Definition	Value
x_1	count(positive lexicon) $\in$ doc)	3
x_2	count(negative lexicon) $\in$ doc)	2
x_3	$\begin{cases} 1 & \text{if “no”} \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$	1
x_4	count(1st and 2nd pronouns $\in$ doc)	3
x_5	$\begin{cases} 1 & \text{if “!”} \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$	0
x_6	log(word count of doc)	$\ln(64) = 4.15$

The features to use is a design decision. A natural default is to use a vector $x \in \mathbb{R}^{|V|}$ where each dim is the counts of one word in the vocabulary. (BOW)

Logistic Regression: LR Model

Given a document $d = w_1, \ldots, w_K$ and a set of classes $\mathscr{C} = \{c_1, \ldots, c_m\}$, we want to find the class c_i that maximizes $P(c \mid d)$

Now given some feature vector x how do we turn this to a probability?

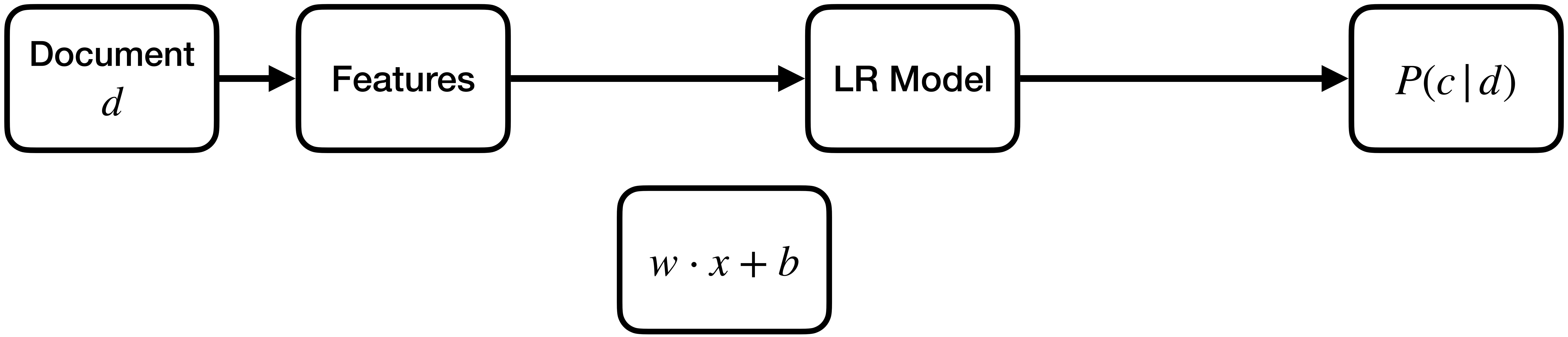


Logistic Regression: LR Model

Given a document $d = w_1, \ldots, w_K$ and a set of classes $\mathscr{C} = \{c_1, \ldots, c_m\}$, we want to find the class c_i that maximizes $P(c \mid d)$

Now given some feature vector x how do we turn this to a probability?

1. Convert the features to a number. The higher the number, the more confident we are that the document belongs to a class. We call these numbers **logits.**

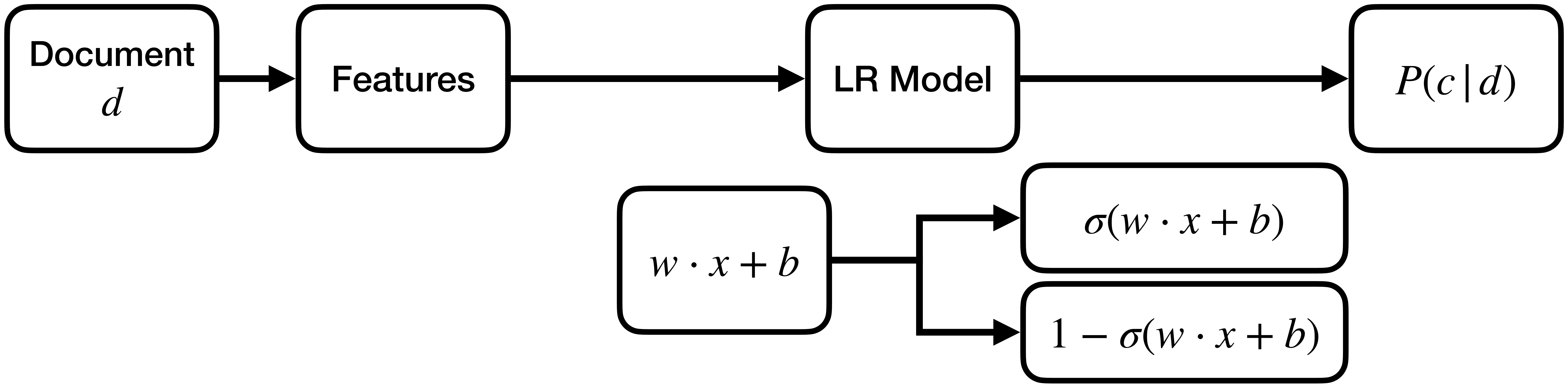


Logistic Regression: LR Model

Given a document $d = w_1, ..., w_K$ and a set of classes $\mathscr{C} = \{c_1, ..., c_m\}$, we want to find the class c_i that maximizes $P(c \mid d)$

Now given some feature vector x how do we turn this to a probability?

1. Convert the features to a number. The higher the number, the more confident we are that the document belongs to a class. We call these numbers **logits.**
2. Normalize the logits using sigmoid so we get a well-defined probability distribution.
    1. For more than 2 classes we use the softmax, which is the m > 2 generalization of sigmoid



Logistic Regression: How do we set w, b ?

Summary: we want to estimate $P(c \mid d)$ using a model $\sigma(w \cdot x + b)$

Logistic Regression: How do we set w, b ?

Summary: we want to estimate $P(c \mid d)$ using a model $\sigma(w \cdot x + b)$

We use GD. But what should we make the loss?

Logistic Regression: How do we set w, b ?

Summary: we want to estimate $P(c \mid d)$ using a model $\sigma(w \cdot x + b)$

We use GD. But what should we make the loss?

Let our train dataset be $\mathscr{D} = \{(d_1, c_1), \ldots, (d_n, c_n)\}$. Let's find the probability of seeing these documents and labels. Assume that each datapoint is independent of the other.

Logistic Regression: How do we set w, b ?

Summary: we want to estimate $P(c \mid d)$ using a model $\sigma(w \cdot x + b)$

We use GD. But what should we make the loss?

Let our train dataset be $\mathscr{D} = \{(d_1, c_1), \ldots, (d_n, c_n)\}$. Let's find the probability of seeing these documents and labels. Assume that each datapoint is independent of the other.

$$P(\mathscr{D}) = P(c_1 \mid d_1) \cdots P(c_n \mid d_n) = \Pi_i P(c_i \mid d_i)$$

Logistic Regression: How do we set w, b ?

Summary: we want to estimate $P(c \mid d)$ using a model $\sigma(w \cdot x + b)$

We use GD. But what should we make the loss?

Let our train dataset be $\mathscr{D} = \{(d_1, c_1), \ldots, (d_n, c_n)\}$. Let's find the probability of seeing these documents and labels. Assume that each datapoint is independent of the other.

$$P(\mathscr{D}) = P(c_1 \mid d_1) \cdots P(c_n \mid d_n) = \Pi_i P(c_i \mid d_i)$$

How to set $\theta = (w, b)$?

Logistic Regression: How do we set w, b ?

Summary: we want to estimate $P(c \mid d)$ using a model $\sigma(w \cdot x + b)$

We use GD. But what should we make the loss?

Let our train dataset be $\mathscr{D} = \{(d_1, c_1), \ldots, (d_n, c_n)\}$. Let's find the probability of seeing these documents and labels. Assume that each datapoint is independent of the other.

$$P(\mathscr{D}) = P(c_1 \mid d_1) \cdots P(c_n \mid d_n) = \Pi_i P(c_i \mid d_i)$$

How to set $\theta = (w, b)$? Use the MLE principle! Set θ such that $P(\mathscr{D})$ is maximized. This is analogous to setting the n-gram probabilities such that the probability of the train corpus is maximal.

Logistic Regression: How do we set w, b ?

Summary: we want to estimate $P(c \mid d)$ using a model $\sigma(w \cdot x + b)$

We use GD. But what should we make the loss?

Let our train dataset be $\mathscr{D} = \{(d_1, c_1), \ldots, (d_n, c_n)\}$. Let's find the probability of seeing these documents and labels. Assume that each datapoint is independent of the other.

$$P(\mathscr{D}) = P(c_1 \mid d_1) \cdots P(c_n \mid d_n) = \Pi_i P(c_i \mid d_i)$$

How to set $\theta = (w, b)$? Use the MLE principle! Set θ such that $P(\mathscr{D})$ is maximized. This is analogous to setting the n-gram probabilities such that the probability of the train corpus is maximal.

So we can directly use GD to minimize: $-\Pi_i P(c_i \mid d_i)$

Logistic Regression: How do we set w, b ?

Summary: we want to estimate $P(c \mid d)$ using a model $\sigma(w \cdot x + b)$

We use GD. But what should we make the loss?

Let our train dataset be $\mathscr{D} = \{(d_1, c_1), \ldots, (d_n, c_n)\}$. Let's find the probability of seeing these documents and labels. Assume that each datapoint is independent of the other.

$$P(\mathscr{D}) = P(c_1 \mid d_1) \cdots P(c_n \mid d_n) = \Pi_i P(c_i \mid d_i)$$

How to set $\theta = (w, b)$? Use the MLE principle! Set θ such that $P(\mathscr{D})$ is maximized. This is analogous to setting the n-gram probabilities such that the probability of the train corpus is maximal.

So we can directly use GD to minimize: $-\Pi_i P(c_i \mid d_i)$

Since $\log$ is monotonic, this is equivalent to minimizing: $-\sum_i \log P(c_i \mid d_i)$ $\leftarrow$ this is just CE loss!

Logistic Regression: How do we set w, b ?

Summary: we want to estimate $P(c \mid d)$ using a model $\sigma(w \cdot x + b)$

Want to minimize: $-\sum \log P(c_i \mid d_i)$ $\leftarrow$ this is just CE loss!

- Loss: $-\log \prod_{i=1}^{n} P(y_i \mid x_i) = -\sum_{i=1}^{n} \log P(y_i \mid x_i)$

$$L_{CE} = -\sum_{i=1}^{n} [y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i)]$$

Logistic Regression: Gradient of Cross-entropy

Summary: in our binary logistic regression using a model $\sigma(\mathbf{w} \cdot \mathbf{x} + b)$, our cross-entropy loss is

$$\mathscr{L}_{CE}(\mathbf{w}, b) = -\frac{1}{n} \sum_{i=1}^{n} [y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i)]$$

How do we differentiate this with gradient descent?

Logistic Regression: Gradient of Cross-entropy

Summary: in our binary logistic regression using a model $\sigma(\mathbf{w} \cdot \mathbf{x} + b)$, our cross-entropy loss is

$$\mathscr{L}_{CE}(\mathbf{w}, b) = -\frac{1}{n}\sum_{i=1}^{n}[y_i \log \hat{y}_i + (1 - y_i)\log(1 - \hat{y}_i)]$$

How do we differentiate this with gradient descent? We need to determine $\frac{d\mathscr{L}}{d\mathbf{w}}, \frac{d\mathscr{L}}{db}$

Logistic Regression: Gradient of Cross-entropy

First we calculate gradient of $\mathscr{L}$ with respect to a specific $\hat{y}_i$ using the chain rule.

Logistic Regression: Gradient of Cross-entropy

First we calculate gradient of $\mathscr{L}$ with respect to a specific $\hat{y}_i$ using the chain rule.

$$\frac{d\mathscr{L}}{d\hat{y}_i} = -\frac{d}{d\hat{y}_i} \cdot \frac{1}{n} \sum_{j=1}^{n} [y_j \log \hat{y}_j + (1 - y_j)\log(1 - \hat{y}_j)]$$

Logistic Regression: Gradient of Cross-entropy

First we calculate gradient of $\mathscr{L}$ with respect to a specific $\hat{y}_i$ using the chain rule.

$$\frac{d\mathscr{L}}{d\hat{y}_i} = -\frac{d}{d\hat{y}_i} \cdot \frac{1}{n} \sum_{j=1}^{n} [y_j \log \hat{y}_j + (1 - y_j)\log(1 - \hat{y}_j)]$$

$$= -\frac{1}{n} \left[\frac{y_i}{\hat{y}_i} - \frac{1 - y_i}{1 - \hat{y}_i} \right]$$

$$\frac{d}{dx}\log(x) = \frac{1}{x}$$

Logistic Regression: Gradient of Cross-entropy

First we calculate gradient of $\mathscr{L}$ with respect to a specific $\hat{y}_i$ using the chain rule.

$$\frac{d\mathscr{L}}{d\hat{y}_i} = -\frac{d}{d\hat{y}_i} \cdot \frac{1}{n} \sum_{j=1}^{n} [y_j \log \hat{y}_j + (1 - y_j) \log(1 - \hat{y}_j)]$$

$$= -\frac{1}{n} \left[\frac{y_i}{\hat{y}_i} - \frac{1 - y_i}{1 - \hat{y}_i} \right]$$

Let $z_i = \mathbf{w} \cdot \mathbf{x}_i + b$, so $\hat{y}_i = \sigma(z_i)$. Differentiating with respect to z_i gives

Logistic Regression: Gradient of Cross-entropy

First we calculate gradient of $\mathscr{L}$ with respect to a specific $\hat{y}_i$ using the chain rule.

$$\frac{d\mathscr{L}}{d\hat{y}_i} = -\frac{d}{d\hat{y}_i} \cdot \frac{1}{n} \sum_{j=1}^{n} [y_j \log \hat{y}_j + (1 - y_j)\log(1 - \hat{y}_j)]$$

$$= -\frac{1}{n}\left[\frac{y_i}{\hat{y}_i} - \frac{1 - y_i}{1 - \hat{y}_i}\right]$$

Let $z_i = \mathbf{w} \cdot \mathbf{x}_i + b$, so $\hat{y}_i = \sigma(z_i)$. Differentiating with respect to z_i gives

$$\frac{d\hat{y}_i}{dz_i} = \sigma(z_i)(1 - \sigma(z_i)) = \hat{y}_i(1 - \hat{y}_i)$$

Logistic Regression: Gradient of Cross-entropy

First we calculate gradient of $\mathscr{L}$ with respect to a specific $\hat{y}_i$ using the chain rule.

$$\frac{d\mathscr{L}}{d\hat{y}_i} = -\frac{d}{d\hat{y}_i} \cdot \frac{1}{n} \sum_{j=1}^{n} [y_j \log \hat{y}_j + (1 - y_j) \log(1 - \hat{y}_j)]$$

$$= -\frac{1}{n} \left[\frac{y_i}{\hat{y}_i} - \frac{1 - y_i}{1 - \hat{y}_i} \right]$$

Let $z_i = \mathbf{w} \cdot \mathbf{x}_i + b$, so $\hat{y}_i = \sigma(z_i)$. Differentiating with respect to z_i gives

$$\frac{d\hat{y}_i}{dz_i} = \sigma(z_i)(1 - \sigma(z_i)) = \hat{y}_i(1 - \hat{y}_i)$$

All in all, the derivative with respect to z_i is

$$\frac{d\mathscr{L}}{dz_i} = \frac{d\mathscr{L}}{d\hat{y}_i} \frac{d\hat{y}_i}{dz_i} = -\frac{1}{n} \left[\frac{y_i}{\hat{y}_i} - \frac{1 - y_i}{1 - \hat{y}_i} \right] \hat{y}_i(1 - \hat{y}_i) =$$

Logistic Regression: Gradient of Cross-entropy

First we calculate gradient of $\mathscr{L}$ with respect to a specific $\hat{y}_i$ using the chain rule.

$$\begin{aligned}\frac{d\mathscr{L}}{d\hat{y}_i} &= -\frac{d}{d\hat{y}_i}\cdot\frac{1}{n}\sum_{j=1}^{n}[y_j\log\hat{y}_j + (1-y_j)\log(1-\hat{y}_j)] \\ &= -\frac{1}{n}\left[\frac{y_i}{\hat{y}_i} - \frac{1-y_i}{1-\hat{y}_i}\right]\end{aligned}$$

Let $z_i = \mathbf{w}\cdot\mathbf{x}_i + b$, so $\hat{y}_i = \sigma(z_i)$. Differentiating with respect to z_i gives

$$\frac{d\hat{y}_i}{dz_i} = \sigma(z_i)(1-\sigma(z_i)) = \hat{y}_i(1-\hat{y}_i)$$

All in all, the derivative with respect to z_i is

$$\frac{d\mathscr{L}}{dz_i} = \frac{d\mathscr{L}}{d\hat{y}_i}\frac{d\hat{y}_i}{dz_i} = -\frac{1}{n}\left[\frac{y_i}{\hat{y}_i} - \frac{1-y_i}{1-\hat{y}_i}\right]\hat{y}_i(1-\hat{y}_i) = \frac{1}{n}\left[\hat{y}_i - y_i\right]$$

Logistic Regression: Gradient of Cross-entropy

Now take derivative with respect to $\mathbf{w}$ and b for the final update equations.

Logistic Regression: Gradient of Cross-entropy

Now take derivative with respect to $\mathbf{w}$ and b for the final update equations.

$$\frac{dz_i}{d\mathbf{w}} = \frac{d}{d\mathbf{w}}(\mathbf{w} \cdot \mathbf{x}_i + b) = \mathbf{x}_i$$

Logistic Regression: Gradient of Cross-entropy

Now take derivative with respect to $\mathbf{w}$ and b for the final update equations.

$$\frac{dz_i}{d\mathbf{w}} = \frac{d}{d\mathbf{w}}(\mathbf{w} \cdot \mathbf{x}_i + b) = \mathbf{x}_i$$

$$\frac{dz_i}{db} = \frac{d}{db}(\mathbf{w} \cdot \mathbf{x}_i + b) = 1$$

Logistic Regression: Gradient of Cross-entropy

Now take derivative with respect to $\mathbf{w}$ and b for the final update equations.

$$\frac{dz_i}{d\mathbf{w}} = \frac{d}{d\mathbf{w}}(\mathbf{w} \cdot \mathbf{x}_i + b) = \mathbf{x}_i$$

$$\frac{dz_i}{db} = \frac{d}{db}(\mathbf{w} \cdot \mathbf{x}_i + b) = 1$$

Combining all together gives

$$\frac{d\mathscr{L}_{CE}}{d\mathbf{w}} = \frac{1}{n}\sum_{i=1}^{n}[\hat{y}_i - y_i]\mathbf{x}_i$$

$$\frac{d\mathscr{L}_{CE}}{db} = \frac{1}{n}\sum_{i=1}^{n}[\hat{y}_i - y_i]$$

Logistic Regression: Summary

Logistic Regression: Summary

1. Given a document $d = w_1, \ldots, w_K$ and a set of classes $\mathscr{C} = \{c_1, \ldots, c_m\}$, we want to find the class c that maximizes $P(c \mid d)$. Let's say we estimating $P(d \mid c)$ reliably is hard, we will need to estimate $P(c \mid d)$ directly.

Logistic Regression: Summary

1. Given a document $d = w_1, \ldots, w_K$ and a set of classes $\mathscr{C} = \{c_1, \ldots, c_m\}$, we want to find the class c that maximizes $P(c \mid d)$. Let's say we estimating $P(d \mid c)$ reliably is hard, we will need to estimate $P(c \mid d)$ directly.
2. Want to turn d into a vector x because then we can operate on it more conveniently.
    1. We can use a BOW, where each dim in $x \in \mathbb{R}^{|V|}$ is the # of times a word in V appears
    2. We can also be creative and add additional features we think are important (e.g. # of emojis in text)

Logistic Regression: Summary

1. Given a document $d = w_1, \ldots, w_K$ and a set of classes $\mathscr{C} = \{c_1, \ldots, c_m\}$, we want to find the class c that maximizes $P(c \mid d)$. Let's say we estimating $P(d \mid c)$ reliably is hard, we will need to estimate $P(c \mid d)$ directly.
2. Want to turn d into a vector x because then we can operate on it more conveniently.
    1. We can use a BOW, where each dim in $x \in \mathbb{R}^{|V|}$ is the # of times a word in V appears
    2. We can also be creative and add additional features we think are important (e.g. # of emojis in text)
3. Somehow we need to turn x into a single number, because $P(c \mid d)$ is a single number.
    1. Let's be as lazy as possible and just take a linear combination of the features: $w \cdot x + b$

Logistic Regression: Summary

1. Given a document $d = w_1, \ldots, w_K$ and a set of classes $\mathscr{C} = \{c_1, \ldots, c_m\}$, we want to find the class c that maximizes $P(c \mid d)$. Let's say we estimating $P(d \mid c)$ reliably is hard, we will need to estimate $P(c \mid d)$ directly.
2. Want to turn d into a vector x because then we can operate on it more conveniently.
    1. We can use a BOW, where each dim in $x \in \mathbb{R}^{|V|}$ is the # of times a word in V appears
    2. We can also be creative and add additional features we think are important (e.g. # of emojis in text)
3. Somehow we need to turn x into a single number, because $P(c \mid d)$ is a single number.
    1. Let's be as lazy as possible and just take a linear combination of the features: $w \cdot x + b$
4. Oh no! The linear combination might not be in $[0,1]$, so we normalize using sigmoid: $\sigma(x) = (1 + e^{-x})^{-1}$
    1. The probability for one class is $\sigma(w \cdot x + b)$, so the other class must have prob $1 - \sigma(w \cdot x + b)$

Logistic Regression: Summary

1. Given a document $d = w_1, \ldots, w_K$ and a set of classes $\mathscr{C} = \{c_1, \ldots, c_m\}$, we want to find the class c that maximizes $P(c \mid d)$. Let's say we estimating $P(d \mid c)$ reliably is hard, we will need to estimate $P(c \mid d)$ directly.
2. Want to turn d into a vector x because then we can operate on it more conveniently.
    1. We can use a BOW, where each dim in $x \in \mathbb{R}^{|V|}$ is the # of times a word in V appears
    2. We can also be creative and add additional features we think are important (e.g. # of emojis in text)
3. Somehow we need to turn x into a single number, because $P(c \mid d)$ is a single number.
    1. Let's be as lazy as possible and just take a linear combination of the features: $w \cdot x + b$
4. Oh no! The linear combination might not be in $[0,1]$, so we normalize using sigmoid: $\sigma(x) = (1 + e^{-x})^{-1}$
    1. The probability for one class is $\sigma(w \cdot x + b)$, so the other class must have prob $1 - \sigma(w \cdot x + b)$
5. Given our model, we can estimate the probability of a train set under the model $P(\mathscr{D})$
    1. We will set w, b so that $P(\mathscr{D})$ = $\Pi_i P(c_i \mid d_i)$ is maximal (MLE principle)
    2. For stability and convenience we can take the $\log$ to minimize $-\sum_i \log P(c_i \mid d_i)$ this is CE loss

Logistic Regression: Summary

1. Given a document $d = w_1, \ldots, w_K$ and a set of classes $\mathscr{C} = \{c_1, \ldots, c_m\}$, we want to find the class c that maximizes $P(c \mid d)$. Let's say we estimating $P(d \mid c)$ reliably is hard, we will need to estimate $P(c \mid d)$ directly.
2. Want to turn d into a vector x because then we can operate on it more conveniently.
   1. We can use a BOW, where each dim in $x \in \mathbb{R}^{|V|}$ is the # of times a word in V appears
   2. We can also be creative and add additional features we think are important (e.g. # of emojis in text)
3. Somehow we need to turn x into a single number, because $P(c \mid d)$ is a single number.
   1. Let's be as lazy as possible and just take a linear combination of the features: $w \cdot x + b$
4. Oh no! The linear combination might not be in $[0,1]$, so we normalize using sigmoid: $\sigma(x) = (1 + e^{-x})^{-1}$
   1. The probability for one class is $\sigma(w \cdot x + b)$, so the other class must have prob $1 - \sigma(w \cdot x + b)$
5. Given our model, we can estimate the probability of a train set under the model $P(\mathscr{D})$
   1. We will set w, b so that $P(\mathscr{D}) = \Pi_i P(c_i \mid d_i)$ is maximal (MLE principle)
   2. For stability and convenience we can take the $\log$ to minimize $-\sum_i \log P(c_i \mid d_i)$ this is CE loss
6. We can then use GD to minimize the CE loss! Since the function is convex, we will converge to the optimum.

Logistic Regression: what's good and what's not

- More freedom in designing features
  - No strong independence assumptions like Naive Bayes
  - Can even have the same feature twice! (*why?*)
- May not work well on small datasets (compared to Naive Bayes)
- Interpreting learned weights can be challenging

Multiclass Classification

- Supervised learning task (e.g. input-output pairs: $\vec{\mathbf{x}}, y$)
- Predict one of k categories (i.e. **classes**)
- Typically, $y \in \{0,1,2,\ldots,k-1\}$
- Examples:
  - Blood typing: Medical information → {A, B, AB, O}
  - Digit recognition: image → {0, 1, …, 9}
  - Object recognition: image → {“golden retriever”, “laptop”, …}
  - Weather prediction: weather metrics → {“sunny”, “cloudy”, “rainy”, “snowy”}

MNIST dataset

From Binary to Multiclass Classification

Extension of logistic regression to multiclass setting

Given $\vec{\mathbf{x}} \in \mathbb{R}^d$, learn k vectors $\theta_1, \theta_2, \ldots, \theta_k \in \mathbb{R}^d$:

$$\Pr[y = i \text{ on } \vec{\mathbf{x}}] = \text{softmax}(\vec{\mathbf{z}}) = \frac{\exp(\theta_i \cdot \vec{\mathbf{x}})}{\sum_{j=1}^{k} \exp(\theta_j \cdot \vec{\mathbf{x}})}, \text{ where } \vec{\mathbf{z}} = \theta_i \cdot \vec{\mathbf{x}}$$

Note about softmax function: $\sum_{i=1}^{k} \text{softmax}(\theta_i \cdot \vec{\mathbf{x}}) = 1$

(e.g. sum of softmax probabilities for all k classes is 1).

θ: `\theta` in Latex

$\theta_1, \ldots, \theta_k$ are generalizations of $\overrightarrow{\mathbf{w}}$ in logistic regression for binary classification.

Multinomial Logistic Regression

Logistic regression

Given $\vec{\mathbf{x}} \in \mathbb{R}^d$ and $y \in \{1, -1\}$,
learn $\overrightarrow{\mathbf{w}} \in \mathbb{R}^d$.

$$\Pr[y \text{ given } \vec{\mathbf{x}}] = \sigma(z)$$

$$= \frac{1}{1 + \exp(-y(\overrightarrow{\mathbf{w}} \cdot \vec{\mathbf{x}}))}$$

where $z = y(\overrightarrow{\mathbf{w}} \cdot \vec{\mathbf{x}})$.

$$\sigma(z) = \frac{1}{1 + e^{-z}} \in [0,1]$$

Multinomial Logistic Regression

Logistic regression

Given $\vec{\mathbf{x}} \in \mathbb{R}^d$ and $y \in \{1, -1\}$,
learn $\vec{\mathbf{w}} \in \mathbb{R}^d$.

$$\Pr[y \text{ given } \vec{\mathbf{x}}] = \sigma(z)$$

$$= \frac{1}{1 + \exp(-y(\vec{\mathbf{w}} \cdot \vec{\mathbf{x}}))}$$

where $z = y(\vec{\mathbf{w}} \cdot \vec{\mathbf{x}})$.

$$\sigma(z) = \frac{1}{1 + e^{-z}} \in [0,1]$$

Multinomial logistic regression

Given $\vec{\mathbf{x}} \in \mathbb{R}^d$ and $y \in \{0, \ldots, k-1\}$,
learn k vectors $\vec{\mathbf{w}}^{(0)}, \vec{\mathbf{w}}^{(1)}, \ldots, \vec{\mathbf{w}}^{(k-1)} \in \mathbb{R}^d$.

$$\Pr[y = i \text{ given } \vec{\mathbf{x}}] = \text{softmax}(\vec{\mathbf{z}})$$

$$= \frac{\exp(\vec{\mathbf{w}}^{(i)} \cdot \vec{\mathbf{x}})}{\sum_{j=0}^{k-1} \exp(\vec{\mathbf{w}}^{(j)} \cdot \vec{\mathbf{x}})}$$

where $\vec{\mathbf{z}} = \vec{\mathbf{w}}^{(i)} \cdot \vec{\mathbf{x}}$.

- $\text{softmax}(\vec{\mathbf{z}}) \in [0,1]$
- $\Sigma_{i=0}^{k-1} \text{softmax}(\theta_i \cdot \vec{\mathbf{x}}) = 1$
- (In CN: $\vec{\mathbf{w}}^{(i)} = \vec{\theta}^i$, θ: `\theta` in Latex)