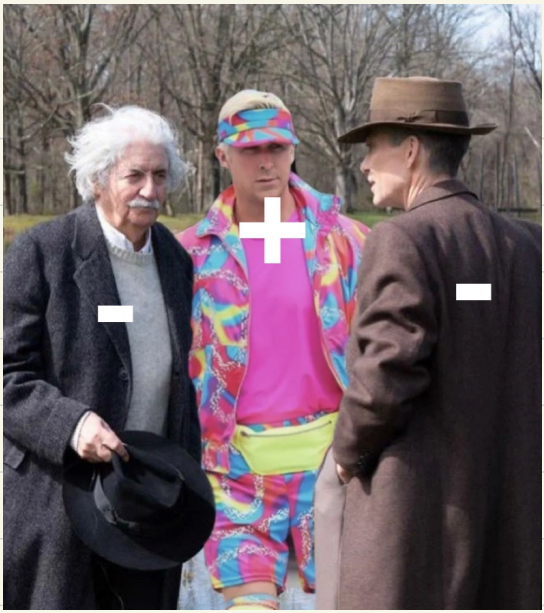
CS371N Lecture 2
Classification 1: Features, Perceptron



Announcements

- AI
- Reading notation != lecture notation

Today
- Classification (linear, binary)
- Feature extraction
- ML basics + Perceptron

Classification

Points $\bar{x}$ (for us: strings)

$f(\bar{x}) \in \mathbb{R}^n$ f: feature extractor

Label $y \in \{-1, +1\}$

Classifier maps $\bar{x} \rightarrow y$

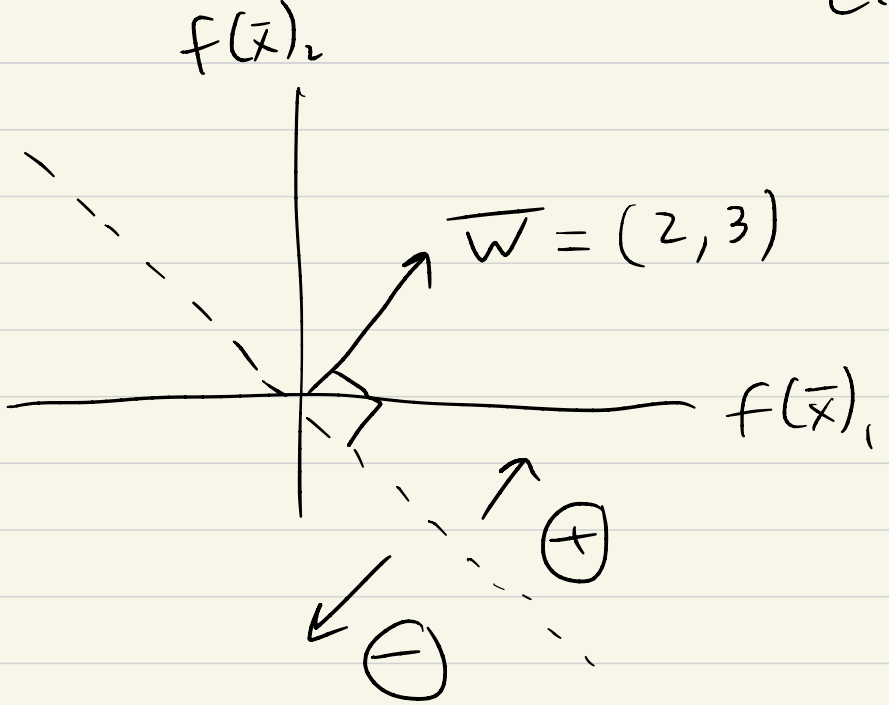
Linear classifier: represented by a weight vector $\bar{w} \in \mathbb{R}^n$

Decision rule: $\bar{w}^T f(\bar{x}) \overset{?}{>} 0$

$\bar{w} \cdot f(\bar{x})$

if >0: +1
else: -1

$n=2$



Sentiment Analysis

$\bar{x}$ = the movie was great!

(1) Feature extraction

$\bar{x} \Rightarrow f(\bar{x}) \in \mathbb{R}^n$

string

(2) Learning algorithm

D exs

Training set $\{(f(\bar{x}^{(i)}), y^{(i)}\}_{i=1}^{D}$

$\Rightarrow \bar{w}$ learned weight vector

Today: Cover #1

see Perceptron for #2

Feature Extraction

$\bar{x}$ = the movie was great

What do we want? — reflect word order

- word frequency ✓
- word meaning / proximity ✗
- negation (not good) ✗
- parts of speech ✗

Bag-of-words featurization

[1 0 0 1 1 1]

the a of ... movie ... great ... was ...

vocabulary ~10,000 words

1 if present

or count of the word

sparse feat. repr.

weight vector: $\bar{w} \in \mathbb{R}^n$

+7 awesome

$[-0.1 \quad +0.2 \quad \ldots \quad +0.3 \quad \ldots \quad +10 \quad \ldots]$

the a movie great ^

$\bar{w}^T f(\bar{x}) = w_{the} + w_{movie} + w_{was} + w_{great}$

"weighted voting"

awesome and great have independent weights

Preprocessing ① Vocab selection: vector space is a fixed set of words

replace unseen words w/ UNK

have a weight for UNK

.split(" ")

That movie... really, it wasn't great!

was not great!

punc
w = [great ... great!] +3 +7 18,000

split punc
w = [great ... ! ...] +3 +0 15,000

Typical tokenization

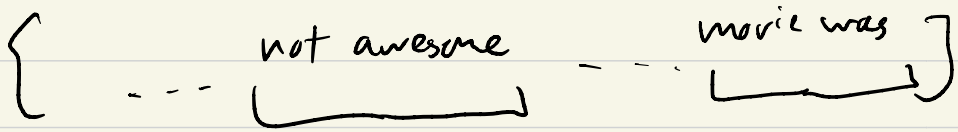
- break out punc.
- break out contractions (was n't)
2. Remove stopwords (the, of, be, etc.)
3. Lowercasing / stemming (arrived ⇒ arrive)

optional

Is lowercasing always good? No!

- text messages
- capture names

Revisit "not awesome"



bigram bag of words

sentence ⇒ adjacent pairs

vocab is now huge ($\sim$ old vocab^2)

We can mix unigrams + bigrams in our feature space

"Custom" feature space

movie	1
was	1
awesome	0
great	1
⋮	⋮
not awesome	0
movie was	1
⋮	

Machine Learning

Optimize parameters $\bar{w}$ to fit some training data (labeled)

We want $\bar{w}$ to make good predictions

$$\text{loss} = \sum_{i=1}^{D} \text{loss}\left(\bar{x}^{(i)}, y^{(i)}, \bar{w}\right)$$

"if we use $\bar{w}$ as our weights, how badly do we mess up?"

Stochastic Gradient Descent

for t in range(0, epochs)

  for i in range(0, D)

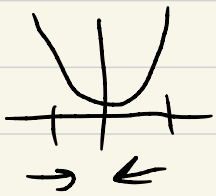
$$\bar{w} \leftarrow \bar{w} - \alpha \frac{\partial}{\partial \bar{w}} \text{loss}(\bar{x}^{(i)}, y^{(i)}, \bar{w})$$

step size

≈ 1

Subtracting gradient of the loss $\Rightarrow$ finding a $\bar{w}$ with lower loss

$\text{loss}(w) = w^2$



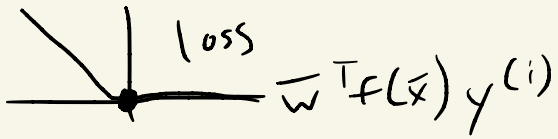
$w = 1$

$\text{gradient} = 2w$

$= 2$

$w = -1$

$\text{gradient} = -2$



Perceptron (instance of SGD)

Initialize $\bar{w} = \bar{0}$

for t in range (0, epochs)

  for i in range (0, D) (shuffle exs each epoch)

$$y_{pred} \leftarrow \begin{cases} 1 & \bar{w}^T f(\bar{x}^{(i)}) > 0 \\ -1 & \text{else} \end{cases}$$

$$\bar{w} \leftarrow \begin{cases} \bar{w} & \text{if } y_{pred} = y^{(i)} \\ \bar{w} + \alpha f(\bar{x}^{(i)}) & \text{if } y^{(i)} = +1 \\ \bar{w} - \alpha f(\bar{x}^{(i)}) & \text{if } y^{(i)} = -1 \end{cases}$$

Let $\alpha = 1$ for now

before

$$\bar{w}^T f(\bar{x}^{(i)})$$

after

$$(\bar{w} + f(\bar{x}^{(i)}))^T f(\bar{x}^{(i)})$$

$$= \bar{w}^T f(\bar{x}^{(i)}) + \underbrace{f(\bar{x}^{(i)})^T f(\bar{x}^{(i)})}_{\|f(\bar{x}^{(i)})\|^2 > 0}$$

Our update rule is sparse

$|V|$ vocab = 10k words

$f(\bar{x}^{(i)})$ has 4 words

we only update 4 weights

Step size

$loss = w^2$

$\frac{\partial l}{\partial w} = 2w$

$w = 1$

$\Rightarrow$ grad $= 2$

$\Rightarrow$ update $= -2\alpha$ α step size

$\alpha = 1$

$\Rightarrow 1 - 2 = -1$

$w = -1$

$\Rightarrow \ldots 1$

$\alpha = 0.5$ ✓

For AI:

Schedule

Start with

$\alpha = 1$

then reduce

$\alpha = \frac{1}{t}$ t epoch