CS540 Introduction to Artificial Intelligence

Lecture 7

Young Wu

Based on lecture slides by Jerry Zhu, Yingyu Liang, and Charles Dyer

June 20, 2022

K Nearest Neighbor

Description

- Given a new instance, find the K instances in the training set that are the closest.
- Predict the label of the new instance by the majority of the labels of the K instances.

Distance Function

Definition

- Many distance functions can be used in place of the Euclidean distance.

$$\rho\left(x, x'\right) = \left\|x - x'\right\|_2 = \sqrt{\sum_{j=1}^{m} \left(x_j - x'_j\right)^2}$$

- An example is Manhattan distance.

$$\rho\left(x, x'\right) = \sum_{j=1}^{m} \left|x_j - x'_j\right|$$

P Norms

Definition

- Another group of examples is the p norms.

$$\rho\left(x, x'\right) = \left(\sum_{j=1}^{m} \left|x_j - x'_j\right|^p\right)^{\frac{1}{p}}$$

- $p = 1$ is the Manhattan distance.
- $p = 2$ is the Euclidean distance.
- $p = \infty$ is the sup distance, $\rho\left(x, x'\right) = \max_{i=1,2,\ldots,m} \left\{\left|x_j - x'_j\right|\right\}$.
- p cannot be less than 1.

K Nearest Neighbor

Algorithm

- Input: instances: $\{x_i\}_{i=1}^n$ and $\{y_i\}_{i=1}^n$, and a new instance $\hat{x}$.
- Output: new label $\hat{y}$.
- Order the training instances according to the distance to $\hat{x}$.

$$\rho\left(\hat{x}, x_{(i)}\right) \leqslant \rho\left(\hat{x}, x_{(i+1)}\right), i = 1, 2, ..., n-1$$

- Assign the majority label of the closest k instances.

$$\hat{y} = \text{ mode } \left\{y_{(1)}, y_{(2)}, ..., y_{(k)}\right\}$$

Natural Language

Motivation

- Generative model: next lecture Bayesian network.
- This lecture: a review of probability, application in natural language.
- The goal is to estimate the probabilities of observing a sentence and use it to generate new sentences.

Tokenization

Motivation

- When processing language, documents (called corpus) need to be turned into a sequence of tokens.

1. Split the string by space and punctuations.
2. Remove stopwords such as "the", "of", "a", "with" ...
3. Lower case all characters.
4. Stemming or lemmatization words: make "looks", "looked", "looking" to "look".

Vocabulary

Motivation

- Word token is an occurrence of a word.
- Word type is a unique token as a dictionary entry.
- Vocabulary is the set of word types.
- Characters can be used in place of words as tokens. In this case, the types are "a", "b", ..., "z", " ", and vocabulary is the alphabet.

Zipf's Law

Motivation

- If the word count if f and the word rank is r, then
$$f \cdot r \approx \text{ constant}$$
- This relation is called Zipf's Law

Bag of Words Features

Definition

- Given a document i and vocabulary with size m, let c_{ij} be the count of the word j in the document i for $j = 1, 2, ..., m$.
- Bag of words representation of a document has features that are the count of each word divided by the total number of words in the document.

$$x_{ij} = \dfrac{c_{ij}}{\displaystyle\sum_{j'=1}^{m} c_{ij'}}$$

TF IDF Features

Definition

- Another feature representation is called tf-idf, which stands for normalized term frequency, inverse document frequency.

$$\text{tf}_{ij} = \frac{c_{ij}}{\max_{j'} c_{ij'}}, \ \text{idf}_j = \log \frac{n}{\sum_{i=1}^{n} \mathbb{1}_{\{c_{ij}>0\}}}$$

$$x_{ij} = \text{tf}_{ij} \, \text{idf}_j$$

- n is the total number of documents and $\sum_{i=1}^{n} \mathbb{1}_{\{c_{ij}>0\}}$ is the number of documents containing word j.

Cosine Similarity

Definition

- The similarity of two documents i and i' is often measured by the cosine of the angle between the feature vectors.

$$\text{sim}\ (x_i, x_{i'}) = \frac{x_i^T x_{i'}}{\sqrt{x_i^T x_i}\sqrt{(x_{i'})^T x_{i'}}}$$

N-Gram Model

Description

- Count all n gram occurrences.
- Apply Laplace smoothing to the counts.
- Compute the conditional transition probabilities.

Token Notations

Definition

- A word (or character) at position t of a sentence (or string) is denoted as z_t.
- A sentence (or string) with length d is $(z_1, z_2, ..., z_d)$.
- $\mathbb{P}\{Z_t = z_t\}$ is the probability of observing $z_t \in \{1, 2, ..., j\}$ at position t of the sentence, usually shortened to $\mathbb{P}\{z_t\}$.

Unigram Model

Definition

- Unigram models assume independence.

$$\mathbb{P}\left\{z_1, z_2, ..., z_d\right\} = \prod_{t=1}^{d} \mathbb{P}\left\{z_t\right\}$$

- In general, two events A and B are independent if:

$$\mathbb{P}\left\{A | B\right\} = \mathbb{P}\left\{A\right\} \text{ or } \mathbb{P}\left\{A, B\right\} = \mathbb{P}\left\{A\right\} \mathbb{P}\left\{B\right\}$$

- For a sequence of words, independence means:

$$\mathbb{P}\left\{z_t | z_{t-1}, z_{t-2}, ..., z_1\right\} = \mathbb{P}\left\{z_t\right\}$$

Maximum Likelihood Estimation

Definition

- $\mathbb{P}\left\{z_t\right\}$ can be estimated by the count of the word z_t.

$$\hat{\mathbb{P}}\left\{z_t\right\} = \frac{c_{z_t}}{\sum_{z=1}^{m} c_z}$$

- This is called the maximum likelihood estimator because it maximizes the probability of observing the sentences in the training set.

MLE Example

Definition

- Let $p = \hat{\mathbb{P}}\{0\}$ in a string with $c_0$0's and $c_1$1's.
- The probability of observing the string is:

$$\binom{c_0 + c_1}{c_0} p^{c_0} (1 - p)^{c_1}$$

- The above expression is maximized by:

$$p^{\star} = \frac{c_0}{c_0 + c_1}$$

Bigram Model

Definition

- Bigram models assume Markov property.

$$\mathbb{P}\left\{z_1, z_2, ..., z_d\right\} = \mathbb{P}\left\{z_1\right\} \prod_{t=2}^{d} \mathbb{P}\left\{z_t | z_{t-1}\right\}$$

- Markov property means the distribution of an element in the sequence only depends on the previous element.

$$\mathbb{P}\left\{z_t | z_{t-1}, z_{t-2}, ..., z_1\right\} = \mathbb{P}\left\{z_t | z_{t-1}\right\}$$

Conditional Probability

Definition

- In general, the conditional probability of an event A given another event B is the probability of A and B occurring at the same time divided by the probability of event B.

$$\mathbb{P}\left\{A|B\right\} = \frac{\mathbb{P}\left\{AB\right\}}{\mathbb{P}\left\{B\right\}}$$

- For a sequence of words, the conditional probability of observing z_t given z_{t-1} is observed is the probability of observing both divided by the probability of observing z_{t-1} first.

$$\mathbb{P}\left\{z_t|z_{t-1}\right\} = \frac{\mathbb{P}\left\{z_{t-1}, z_t\right\}}{\mathbb{P}\left\{z_{t-1}\right\}}$$

Bigram Model Estimation

Definition

- Using the conditional probability formula, $\mathbb{P}\left\{z_t \middle| z_{t-1}\right\}$, called transition probabilities, can be estimated by counting all bigrams and unigrams.

$$\hat{\mathbb{P}}\left\{z_t \middle| z_{t-1}\right\} = \frac{c_{z_{t-1},z_t}}{c_{z_{t-1}}}$$

Transition Matrix

Definition

- These probabilities can be stored in a matrix called transition matrix of a Markov Chain. The number on row j column j' is the estimated probability $\hat{\mathbb{P}}\left\{j' | j\right\}$. If there are 3 tokens $\left\{1, 2, 3\right\}$, the transition matrix is the following.

$$\begin{bmatrix} \hat{\mathbb{P}}\left\{1|1\right\} & \hat{\mathbb{P}}\left\{2|1\right\} & \hat{\mathbb{P}}\left\{3|1\right\} \\ \hat{\mathbb{P}}\left\{1|2\right\} & \hat{\mathbb{P}}\left\{2|2\right\} & \hat{\mathbb{P}}\left\{3|2\right\} \\ \hat{\mathbb{P}}\left\{1|3\right\} & \hat{\mathbb{P}}\left\{2|3\right\} & \hat{\mathbb{P}}\left\{3|3\right\} \end{bmatrix}$$

- Given the initial distribution of tokens, the distribution of the next token can be found by multiplying it by the transition probabilities.

Aside: Stationary Probability

Discussion

- Given the bigram model, the fraction of times a token occurs for a document with infinite length can be computed. The resulting distribution is called the stationary distribution.

$$p_{\infty} = p_0 M^{\infty}$$

Aside: Spectral Decomposition

Discussion

- It is easier to find powers of diagonal matrices.
- Let D be the diagonal matrix with eigenvalues of M on the diagonal and P be the matrix with columns being corresponding eigenvectors.

$$\begin{aligned} MP &= \lambda_i P, i = 1, 2, ..., K \\ MP &= PD \\ M &= PDP^{-1} \\ M^n &= \underbrace{PDP^{-1}PDP^{-1}...PDP^{-1}}_{n \text{ times}} = PD^n P^{-1} \\ M^\infty &= PD^\infty P^{-1} \end{aligned}$$

Aside: Stationarity

Discussion

- A simpler way to compute the stationary distribution is to solve the equation:

$$p_\infty = p_\infty M$$

Trigram Model

Definition

- The same formula can be applied to trigram: sequences of three tokens.

$$\hat{\mathbb{P}}\left\{z_t \middle| z_{t-1}, z_{t-2}\right\} = \frac{c_{z_{t-2}, z_{t-1}, z_t}}{c_{z_{t-2}, z_{t-1}}}$$

- In a document, likely, these longer sequences of tokens never appear. In those cases, the probabilities are $\dfrac{0}{0}$. Because of this, Laplace smoothing adds 1 to all counts.

$$\hat{\mathbb{P}}\left\{z_t \middle| z_{t-1}, z_{t-2}\right\} = \frac{c_{z_{t-2}, z_{t-1}, z_t} + 1}{c_{z_{t-2}, z_{t-1}} + m}$$

Laplace Smoothing

Definition

- Laplace smoothing should be used for bigram and unigram models too.

$$\hat{\mathbb{P}}\left\{z_t | z_{t-1}\right\} = \frac{c_{z_{t-1}, z_t} + 1}{c_{z_{t-1}} + m}$$

$$\hat{\mathbb{P}}\left\{z_t\right\} = \frac{c_{z_t} + 1}{\displaystyle\sum_{z=1}^{m} c_z + m}$$

- Aside: Laplace smoothing can also be used in decision tree training to compute entropy.

N Gram Model

Algorithm

- Input: series $\left\{z_1, z_2, ..., z_{d_i}\right\}_{i=1}^{n}$.
- Output: transition probabilities $\hat{\mathbb{P}}\left\{z_t \middle| z_{t-1}, z_{t-2}, ..., z_{t-N+1}\right\}$ for all $z_t = 1, 2, ..., m$.
- Compute the transition probabilities using counts and Laplace smoothing.

$$\hat{\mathbb{P}}\left\{z_t \middle| z_{t-1}, z_{t-2}, ..., z_{t-N+1}\right\} = \frac{c_{z_{t-N+1}, z_{t-N+2}, ..., z_t} + 1}{c_{z_{t-N+1}, z_{t-N+2}, ..., z_{t-1}} + m}$$

Sampling from Discrete Distribution

Discussion

- To generate new sentences given an N gram model, random realizations need to be generated given the conditional probability distribution.
- Given the first $N-1$ words, $z_1, z_2, ..., z_{N-1}$, the distribution of next word is approximated by $p_x = \hat{\mathbb{P}}\left\{z_N = x \middle| z_{N-1}, z_{N-2}, ..., z_1\right\}$. This process then can be repeated for on $z_2, z_3, ..., z_{N-1}, z_N$ and so on.

Inverse Transform Sampling, Part I

Discussion

- Most programming languages have a function to generate a random number $u \sim$ Unif $[0, 1]$.
- If there are $m = 2$ tokens in total and the conditional probabilities are p and $1 - p$. Then the following distributions are the same.

$$z_N = \begin{cases} 0 & \text{with probability } p \\ 1 & \text{with probability } 1 - p \end{cases} \Leftrightarrow z_N = \begin{cases} 0 & \text{if } 0 \leqslant u \leqslant p \\ 1 & \text{if } p < u \leqslant 1 \end{cases}$$

Inverse Transform Sampling, Part II

Discussion

- In the general case with m tokens with conditional probabilities $p_1, p_2, ..., p_m$ with $\sum_{j=1}^{m} p_j = 1$. Then the following distributions are the same.

$$z_N = j \text{ with probability } p_j \Leftrightarrow z_N = j \text{ if } \sum_{j'=1}^{j-1} p_{j'} < u \leqslant \sum_{j'=1}^{j} p_{j'}$$

- This can be used to generate a random token from the conditional distribution.

Sparse Matrix

Discussion

- The transition matrix is too large with mostly zeros.
- Usually, clustering is done so each type (or feature) represent a group of words.
- For the homework, treat each character (letter or space) as a token, then there are $26 + 1$ types. All punctuations are removed or converted to spaces.