Lecture Notes
Part 1 of 2

Mathematical Foundations of Computational Linguistics

Manfred Klenner, Jannis Vamvas
Spring Semester 2024
University of Zurich

This PDF is formatted to fit on a phone screen.

Contents

The Greek Alphabet

A	α	Alpha
B	β	Beta
Γ	γ	Gamma
Δ	δ	Delta
E	ϵ	Epsilon
Z	ζ	Zeta
H	η	Eta
Θ	θ / ϑ	Theta
I	ι	Iota
K	κ	Kappa
Λ	λ	Lambda
M	μ	Mu
N	ν	Nu
Ξ	ξ	Xi
O	o	Omicron
Π	π	Pi
P	ρ	Rho
Σ	σ	Sigma
T	τ	Tau
Υ	υ	Upsilon
Φ	ϕ	Phi
X	χ	Chi
Ψ	ψ	Psi
Ω	ω	Omega

Chapter 1

Sets

Definition of Sets

A set is a collection of objects: $M = \{a, b, c\}$.

If an object x is in a set M, we write $x \in M$.
If it is not in the set, we write $x \notin M$.

The number of elements in a set M is called its cardinality and is denoted by $|M|$.

There are several ways to define a set:

1. Enumeration:
   $M = \{\mathrm{a}, \mathrm{e}, \mathrm{i}, \mathrm{o}, \mathrm{u}\}$
2. Comprehension:
   $M = \{x \mid x \text{ is a vowel}\}$

Examples of Sets

The following sets of numbers are especially relevant for us:

$\mathbb{R}$ is the set of real numbers.

$\mathbb{Q}$ is the set of rational numbers (fractions like $\frac{1}{2}$).

$\mathbb{Z} = \{\ldots, -2, -1, 0, 1, 2, \ldots\}$ is the set of integers.

$\mathbb{N} = \{1, 2, 3, \ldots\}$ is the set of natural numbers.

$\mathbb{N}_0$ is the set of natural numbers including zero.

$\mathbb{R} \setminus \mathbb{Q}$ is the set of real numbers that cannot be expressed as a fraction of two integers (*irrational numbers* such as $\sqrt{2}$ or π).

Subsets

A set M is called a subset of a set N if every element of M is also an element of N.

Conversely, N is a superset of M.

A subset is called a *proper* subset if it contains fewer elements than the superset (is not equal to the superset); we write $M \subset N$.

The notation $M \subseteq N$ means that M is either a subset of N or equal to N.

To say that M is not a subset of N, we write $M \not\subseteq N$.

Properties of Sets

Sets do not have a defined order:
$\{a,b\} = \{b,a\}$.

Duplicate elements are ignored in sets:
$\{a,a,b\} = \{a,b\}$.

Sets can have an infinite number of elements.

Sets can contain other sets as elements:
$\{\{a,b\},\{c,d\}\}$.

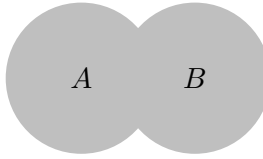
The empty set $\emptyset$ is the set that contains no elements.

Every set has the empty set as a subset:
$\emptyset \subseteq M$.[1]

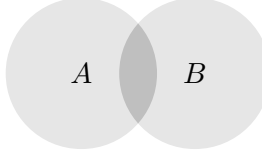
[1] This follows from the definition of subsets on the previous page.

Set Operations

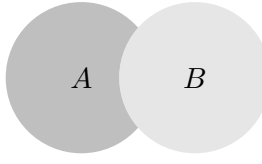
Union: $A \cup B = \{x \mid x \in A \text{ or } x \in B\}$



Intersection: $A \cap B = \{x \mid x \in A \text{ and } x \in B\}$



Difference: $A \setminus B = \{x \mid x \in A \text{ and } x \notin B\}$



More Definitions

Two sets are called disjoint if their intersection is empty, i.e., $A \cap B = \emptyset$.

The complement of a set A is the set of all elements that are not in A; we write $\overline{A}$.

The power set $\mathcal{P}(A)$ is the set of all subsets of A. The power set has the cardinality $2^{|A|}$.[2]

[2] For this reason, 2^A is sometimes used as an alternative notation for the power set.

Cartesian Product and Tuples

The Cartesian product of two sets A and B is the set of all ordered pairs between elements of the two sets: $A \times B = \{\langle x, y\rangle \mid x \in A \text{ and } y \in B\}$.

A pair $\langle x, y\rangle$ is called a tuple. Contrary to sets, tuples are ordered and can contain duplicate elements.

A Cartesian product can also be created from more than two sets: $A_1 \times A_2 \times \cdots \times A_n$. In this case, the resulting tuples contain n elements and are called *n-tuples*.

n-tuples of length 3 are also called *triples*, n-tuples of length 4 are called *quadruples*, and so on.

The Cartesian product of a set A with itself n times is written as A^n:

$$A^n = \underbrace{A \times A \times \cdots \times A}_{n \text{ times}}$$

Sums and Products

The sum of the elements of a set A can be written as:

$$\sum_{x \in A} x$$

Similarly, the product of the elements can be written as:

$$\prod_{x \in A} x$$

When dealing with sequences of variables (e.g., a list of numbers), we can use the same notation, but we need to index the variables using a subscript:

$$\sum_{i=1}^{n} x_i = x_1 + x_2 + \cdots + x_n$$

$$\prod_{i=1}^{n} x_i = x_1 \cdot x_2 \cdot \cdots \cdot x_n$$

In the above notation, i is an *index variable* that runs from 1 to n. The initial value of the index is written below the summation or product symbol, and the final value is written above.

Chapter 2

Relations and Functions

Relations

A relation R between two sets A and B is a set of tuples $\langle x, y \rangle$ where $x \in A$ and $y \in B$.

Typical examples of relations include:

- The set of all pairs of people who are married to each other.
- The set of all pairs of people $\langle x, y \rangle$ where x is the parent of y.
- The set of all numbers $\langle x, y \rangle$ where x is greater than y.

Two notation conventions are commonly used to express that a tuple $\langle x, y \rangle$ is in a relation R:

$$xRy \qquad \text{or} \qquad \langle x, y \rangle \in R$$

A relation R between two sets M and N is a subset of the Cartesian product of the two sets: $R \subseteq M \times N$.

The domain of a relation R is the set of all elements x for which there exists a y such that $\langle x, y \rangle \in R$.

The range of a relation R is the set of all elements y for which there exists an x such that $\langle x, y \rangle \in R$.

The *inverse* of a relation R is denoted with R^{-1}. It contains all the reversed tuples $\langle y, x \rangle$ for which $\langle x, y \rangle \in R$.

Properties of Relations

A relation R is reflexive if $\langle x,x\rangle \in R$ for all x in a set M.
Example: the relation *"a is equal to b"* is reflexive.

A relation R is symmetric if $\langle x,y\rangle \in R$ implies $\langle y,x\rangle \in R$.
Example: the relation *"a is married to b"* is symmetric.

A relation R is asymmetric if $\langle x,y\rangle \in R$ implies $\langle y,x\rangle \notin R$.
Example: the relation *"a is the parent of b"* is asymmetric.

A relation R is transitive if $\langle x,y\rangle \in R$ and $\langle y,z\rangle \in R$ implies $\langle x,z\rangle \in R$.
Example: the relation *"a is older than b"* is transitive.

A relation R is intransitive if $\langle x,y\rangle \in R$ and $\langle y,z\rangle \in R$ implies $\langle x,z\rangle \notin R$.
Example: the relation *"a is the parent of b"* is intransitive.

A relation R is connex if $\langle x,y\rangle \in R$ or $\langle y,x\rangle \in R$ for all x and y in a set M with $x \neq y$. Example: *"a is greater than b"* is connex for natural numbers.

Equivalence Relations

A relation is called an equivalence relation if it is reflexive, symmetric, and transitive.

An equivalence relation splits a base set A into a set of disjoint equivalence classes. All elements in an equivalence class are related to each other and are not related to any elements in other equivalence classes.

Given a base set A and an equivalence relation R, the equivalence class of an element a is denoted as $[a] = \{x \in A \mid xRa\}$.

Transitive Closure

The transitive closure of a relation R is denoted as R^+. It is the superset of R that explicitly contains all transitively given pairs.

The transitive closure of a relation R can be computed by adding all pairs $\langle x, z \rangle$ for which there exists a y such that $\langle x, y \rangle \in R$ and $\langle y, z \rangle \in R$.

Functions

A function is a special case of a relation, where each element in the domain is related to no more than one element in the range.

A *total* function is a function where each element in the domain is related to an element in the range. A *partial* function is a function where some elements in the domain are not related to any element in the range.

The most common notation for a function f is $f : M \to N$. It means that f is a function that maps from the domain M to the range N.

The notation $f(x) = y$ indicates that $\langle x, y \rangle \in f$.

Functions can also be defined piecewise, for example:

$$f(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{otherwise} \end{cases}$$

The above is an example of an indicator function. It returns 1 if a condition is true and 0 otherwise.

Chapter 3

Probability

Probability Functions

A probability function P is a function that maps an event A to its probability. A probability is a value between 0 and 1, where 0 means that the event is impossible and 1 means that it is certain.

The set of all elementary events is called the sample space and is denoted as Ω.

Events can be combinations of elementary events. Thus, events are subsets of the sample space: $A \subseteq \Omega$, and a probability function is a function $P : 2^{\Omega} \to [0, 1]$.

A probability distribution is a function that maps an elementary event to its probability. The sum of all probabilities in a probability distribution must be 1.

Special Types of Probabilities

$P(A \mid B)$ is the *conditional probability* of event A given that event B has occurred.

In some contexts, $P(A \mid B)$ is also called the *posterior probability* of A given B. $P(A)$ is called the *prior probability* of A, i.e., the probability of A before any additional evidence is taken into account.

$P(A \cap B)$ is the *joint probability* of events A and B, i.e., the probability that both A and B occur.

Joint probability can be expressed in terms of conditional probability:

$$P(A \cap B) = P(A \mid B) \cdot P(B) = P(B \mid A) \cdot P(A)$$

Two events A and B are called independent if $P(A \cap B) = P(A) \cdot P(B)$.

Above, we used $\cdot$ to denote multiplication, but we will omit it in the following and simply write $P(A)P(B)$.

Bayes' Theorem

Bayes' theorem describes the relationship between conditional probabilities:

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}$$

Logarithms

The logarithm of a number x is the *exponent* to which another fixed number, the *base* b, must be raised to produce that number x. It is denoted as $\log_b x$.

The *binary logarithm* is the logarithm with base 2. It is denoted as $\log_2 x$.

The *common logarithm* is the logarithm with base 10, denoted as $\log_{10} x$.

Finally, the *natural logarithm* is the logarithm with base e, where e is Euler's number, approximately equal to 2.718. It is denoted as $\ln x$.

Some useful properties of logarithms are:

$$\log(uv) = \log u + \log v$$
$$\log\left(\frac{u}{v}\right) = \log u - \log v$$
$$\log(u^v) = v \log u$$

The exponential function e^x is the inverse of the natural logarithm: $e^{\ln x} = x$.

An alternative notation for the exponential function is $\exp(x)$.

Logarithms of Probabilities

In practice, we often work with the logarithms of probabilities instead of the probabilities themselves. This is called working in *log space* as opposed to *probability space*.

Recall that probabilities are values between 0 and 1. Conversely, log probabilities range between $-\infty$ and 0.

The relation between probabilities and log probabilities is *monotonic*, meaning that if $P(A) > P(B)$, then $\log P(A) > \log P(B)$.

Chapter 4

Naive Bayes Classifiers

Supervised Learning

In a supervised classification task, we are given a set of training examples, each of which is a pair consisting of an input x and a label y. The goal is to learn a function f that can accurately label new examples. For example, in a spam detection task, the input x is an email, and the label y is either 'spam' or 'not spam'.

A set of examples X is usually split into a *training set* and a *test set*. The training set is used to learn the classifier function, and the test set is used to evaluate its accuracy on new examples.

Examples are represented as collections of *fea-*

tures. For example, in a spam detection task, the features could be information about the words in the email (e.g., whether a word is present or not, or how many times it appears).

Because counting the words in a document does not take their order into account, this representation is called a *bag of words*. A bag-of-words classifier is easy to describe mathematically, but it makes simplifying (*naive*) assumptions about language.

Maximum Likelihood Estimation

Maximum likelihood estimation (MLE) means to observe a data sample and then estimate the probabilities that are most likely to produce the observed data.

MLE builds on the principle of counting the occurrences of events in a sample and then using these counts to estimate the probabilities of the events. If N observations are made and event A has occured $f(A)$ times, then MLE estimates the probability of A as:

$$P(A) = \frac{f(A)}{N}$$

Naive Bayes for Text Classification

In the remainder of this chapter, we will mathematically describe a bag-of-words classifier for text documents. The specific approach that we use is called a naive Bayes classifier.

It is *naive* because it makes the simplifying assumption that the features (words) of a document are independent of each other. It is called *Bayes* because it makes use of Bayes' theorem.

The classifier, given a document d, returns the class $\hat{c}$ with the maximum posterior probability of all classes C.

We use the following notation:

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} P(c \mid d)$$

Using Bayes' theorem, we can rewrite:

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} \frac{P(c)P(d \mid c)}{P(d)}$$

Finally, we can omit $P(d)$ because it does not make a difference to the argmax:

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} P(c)P(d \mid c)$$

Estimating the Prior

The prior probability $P(c)$ is the overall probability of a class c. Using maximum likelihood estimation, we can estimate it as the frequency of the class in the training set:

$$\hat{P}(c) = \frac{|\{d \in D \mid d \text{ is of class } c\}|}{|D|}$$

where D is the set of documents that we use for training.

Estimating the Document Likelihood

For estimating the likelihood of the document d given the class c, we decompose it into the likelihood of each feature f given the class c:

$$P(d \mid c) = P(f_1, f_2, \ldots, f_n \mid c)$$

Thanks to our naive assumption that the features are independent, we can rewrite this as:

$$\begin{aligned} P(d \mid c) &= P(f_1 \mid c) P(f_2 \mid c) \cdots P(f_n \mid c) \\ &= \prod_{i=1}^{n} P(f_i \mid c) \end{aligned}$$

Log Space

A numerical challenge with the naive Bayes classifier is that multiplying many probability values can lead to such a small result that the computer cannot distinguish it from zero, because it represents numbers with limited precision.

To avoid this problem, we can use logarithms of probabilities instead of probabilities themselves:

$$\begin{aligned}\hat{c} &= \operatorname*{argmax}_{c \in C} P(c) \prod_{i=1}^{n} P(f_i \mid c) \\ &= \operatorname*{argmax}_{c \in C} \log \left(P(c) \prod_{i=1}^{n} P(f_i \mid c) \right) \\ &= \operatorname*{argmax}_{c \in C} \left(\log P(c) + \sum_{i=1}^{n} \log P(f_i \mid c) \right)\end{aligned}$$

Estimating the Feature Likelihoods

Recall that as features, we use the individual words in the document. First, we collect all words in all training documents into a vocabulary V. Every feature f_i corresponds to a word $w_i \in V$.

To train the classifier, we need to estimate the likelihood $P(f_i \mid c) = P(w_i \mid c)$ of each feature.

We again use maximum likelihood estimation. Let $\text{count}(w_i, c)$ be the number of times that word w_i occurs across all documents of class c.[1] Then, we estimate the likelihood of the words as:

$$\hat{P}(w_i \mid c) = \frac{\text{count}(w_i, c)}{\sum_{w \in V} \text{count}(w, c)}$$

[1] A naive Bayes classifier that relies on the counts of words in documents is called a *multinomial* naive Bayes classifier. There are other types of naive Bayes classifiers, e.g., those that just rely on the presence or absence of words in documents, irrespective of their counts.

Smoothing

A problem with the above estimation is that if a word w_i does not occur in any document of class c, then $\text{count}(w_i, c) = 0$ and $\hat{P}(w_i \mid c) = 0$. The classifier will assign zero probability to a document that contains an unexpected word.

Smoothing is a technique to avoid this problem by reserving some probability mass for unseen words.

A simple smoothing technique is *add-one smoothing*, where we assume that every word occurs once more than it actually does in the training data:

$$\hat{P}(w_i \mid c) = \frac{\text{count}(w_i, c) + 1}{\sum_{w \in V} \text{count}(w, c) + |V|}$$

Chapter 5

Language Modeling with n-grams

Chain Rule of Probability

Recall that the joint probability of two events can be written as: $P(A \cap B) = P(A)P(B \mid A)$.

The *chain rule* generalizes this to the joint probability of multiple events:

For 3 events:

$$P(A_1 \cap A_2 \cap A_3) = P(A_1)P(A_2 \mid A_1)P(A_3 \mid A_1 \cap A_2)$$

For n events:

$$P(\cap_{i=1}^{n} A_i) = P(A_1)P(A_2 \mid A_1) \cdots P(A_n \mid \cap_{i=1}^{n-1} A_i)$$

n-grams

An n-gram is a sequence of n tokens. Here, we use words as tokens, but n-grams can also be defined for other types of tokens, such as individual characters.

A *unigram* is a single word, a *bigram* is a sequence of two words, a *trigram* is a sequence of three words, and so on.

For example, the sentence *"I am Sam and"* contains the following n-grams:

- Unigrams: *I*, *am*, *Sam*, *and*
- Bigrams: *I am*, *am Sam*, *Sam and*
- Trigrams: *I am Sam*, *am Sam and*
- 4-grams: *I am Sam and*

Mathematically, we can denote an n-gram as an n-tuple of words:

$$\langle w_1, \ldots, w_n \rangle$$

or using a shorthand notation:

$$w_{1:n}$$

n-gram Language Models

Generally, a language model estimates the probability of a word w_i at a specific position in a sentence, given the history of preceding words $w_{1:i-1}$. A training corpus is used to estimate these probabilities.

An *n-gram language model* is a simplified language model that considers a limited history of $n-1$ preceding words to estimate the probability of the word.

$$P(w_i \mid w_{1:i-1}) \approx P(w_i \mid w_{i-n+1:i-1}),$$

where $\approx$ indicates that this is an approximative model (since only a limited history is considered).

The simplifying assumption that the probability of a word depends on a fixed number of preceding words is called the Markov assumption.

Here, we consider the special case of *bigram language models*, where it is assumed that the probability of a word only depends on the immediately preceding word:

$$P(w_i \mid w_{1:i-1}) \approx P(w_i \mid w_{i-1})$$

Estimating n-gram Probabilities

The probabilities of a bigram language model can be estimated by counting frequencies of n-grams in a training corpus:

$$P(w_i \mid w_{i-1}) = \frac{\text{count}(\langle w_{i-1}, w_i \rangle)}{\sum_{w \in V} \text{count}(\langle w_{i-1}, w \rangle)}$$

$$= \frac{\text{count}(\langle w_{i-1}, w_i \rangle)}{\text{count}(w_{i-1})}$$

Recall that we already used the relative frequency of words for estimating probabilities in the context of naive Bayes classifiers. We called this approach *maximum likelihood estimation* (MLE).

Sentence Probabilities

A language model can also be used to estimate the probability of a sequence of words, e.g., a sentence.

To estimate the probability of a word sequence $w_{1:m}$, we can use the chain rule of probability to decompose it into the probabilities of the individual words:

$$\begin{aligned} P(w_{1:m}) &= P(w_1)P(w_2 \mid w_1) \cdots P(w_m \mid w_{1:m-1}) \\ &= \prod_{i=1}^{m} P(w_i \mid w_{1:i-1}) \end{aligned}$$

In the case of a bigram language model, the Markov assumption gives us:

$$P(w_{1:m}) \approx \prod_{i=1}^{m} P(w_i \mid w_{i-1})$$

Practical Tricks for n-gram Language Models

In practice, n-gram language modeling can benefit from several tricks:

- *Start-of-sentence* and *end-of-sentence* tokens: We add a special symbol <s> to the beginning of each sentence. Similarly, we add a special symbol </s> to the end of each sentence. This allows us to model the probability of the actual first word as $P(w_1 \mid \text{<s>})$, and the probability of a sentence ending after word w_n as $P(\text{</s>} \mid w_n)$.

- *Smoothing*: We can use smoothing techniques such as add-one smoothing to avoid zero probabilities for unseen n-grams.

- *Log space*: We use the sum of logarithms instead of the product of probabilities to avoid numerical underflow:

$$P(w_{1:m}) = \exp\left(\sum_{i=1}^{m} \log P(w_i \mid w_{i-1})\right)$$

Chapter 6

Statistics

Random Variables

A random variable is a variable that can take on different values, each with a certain probability. For example, the possible values of a random variable X that represents the outcome of a die roll are 1, 2, 3, 4, 5, and 6.

The *probability distribution* $P(X)$ of a random variable is a function that maps each possible value to its probability. In the example of a die, the probability distribution of X is:

$$P(X=1)=\frac{1}{6},\quad \ldots,\quad P(X=6)=\frac{1}{6}.$$

Expected Value

The expected value of a random variable is the average value that we expect to see if we repeat the experiment many times.

Assuming that a random variable X can take a finite number of values $x_1, x_2, \ldots, x_n$, the expected value of X is:

$$E(X) = \sum_{x_i} x_i P(X = x_i).$$

In other words, the expected value is the sum of the values weighted by their probability (*weighted average*).

The expected value of a die roll is:

$$E(X) = \frac{1}{6} \cdot 1 + \frac{1}{6} \cdot 2 + \ldots + \frac{1}{6} \cdot 6 = 3.5.$$

The mode of a random variable is the value with the highest probability. If the probability distribution is drawn as a histogram, the mode is the highest bar.

Variance and Standard Deviation

The expected value $E(X)$ of a random variable is a measure of its *central tendency*, of the center of its distribution. Because $E(X)$ is a mean, it is often denoted by the symbol μ.

The variance $V(X)$ of a random variable is a measure of its *spread*, of how far its values are from the center. An alternative notation for the variance is σ^2, where σ is called the standard deviation of X.

There are several equivalent definitions of the variance of a random variable X.

$$\begin{aligned} V(X) &= E\left((X-\mu)^2\right) \\ &= E(X^2) - \mu^2 \\ &= \sum_{x_i} (x_i - \mu)^2 P(X = x_i) \end{aligned}$$

Analyzing Data Samples

Mean and variance are especially useful when analyzing a *sample* drawn from an overall *population*. We use different notation to distinguish between a description of the population and a description of a sample from that population:

- The expected value and variance of the overall population are denoted with μ and σ^2.

- The mean and variance of a sample are denoted with $\bar{x}$ and s^2.

Whenever we estimate the mean and variance of a population from a sample, we use the 'hat' operator to indicate that we are dealing with estimated values: $\bar{x} = \hat{\mu} \approx \mu$ and $s^2 = \hat{\sigma}^2 \approx \sigma^2$.

The mean of a sample is simply the arithmetic mean of the values in the sample:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$$

The variance of sample tends to be smaller than the variance of the population. For this reason, it is recommended to use a corrected formula for sample variance.

Urn Model

The urn model allows us to reason about random experiments, such as drawing balls from an urn. We distinguish between different types of urn problems:

- How many trials: Do we draw a ball once or k times?

- With or without *repetition*: If we draw a ball from the urn, do we put it back before drawing the next one?

- With or without *order*: Do we care about the order in which we draw the balls?

Combinatorics of Urn Problems

The factorial of a number n is the product of all numbers from 1 to n:

$$n! = n \cdot (n-1) \cdot (n-2) \cdots 1$$

A permutation is an ordered sequence of k elements. The number of possible permutations of k elements is equal to the factorial of k. For example, three objects can be arranged in $3! = 6$ different ways.

A combination is an unordered selection of k elements.

We use the notation $\binom{n}{k}$ to denote the number of combinations of k elements from a set of n elements (read: "n choose k").

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

For example, there are $\binom{3}{2} = 3$ combinations of two elements from a set of three elements.

The four formulas below are useful for solving urn problems. Let n be the number of elements in the urn and k the number of draws.

Number of permutations with repetition:

$$n^k$$

Number of permutations without repetition:

$$\frac{n!}{(n-k)!}$$

Number of combinations with repetition:

$$\binom{n+k-1}{k}$$

Number of combinations without repetition:

$$\binom{n}{k}$$

Chapter 7

Probability Distributions

Recall that a probability distribution is a function that assigns a probability to each possible outcome of a random experiment.

Probability distributions can be *discrete* or *continuous*.

Discrete probability distributions are defined for random variables that can only take on a finite number of value (e.g., the outcome of a die roll).

Continuous probability distributions are defined for random variables that can take on any value in a continuous range (e.g., the height of a person). In this case, it is only useful to describe the probability of a range (*interval*) of values (e.g.,

the probability that a person is between 170 and 180 cm tall), and not the probability of a specific value.

This distinction is reflected in two types of functions that describe probability distributions:

- *Probability mass functions* (PMFs) are used to describe discrete probability distributions. They assign a probability to each possible outcome of a random experiment. A PMF looks like a *histogram*, i.e., a bar chart with a bar for each possible value of the random variable.

- *Probability density functions* (PDFs) are used to describe continuous probability distributions. They assign a probability *density* to each possible value of a random variable.

Binomial Distribution

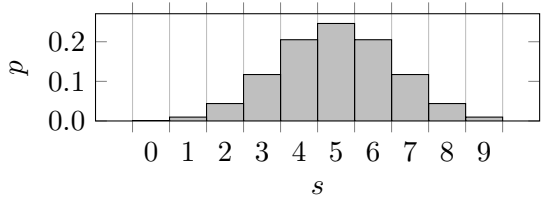
An important discrete probability distribution is the binomial distribution. It describes the probability of observing s successes in n independent trials, where the probability of success is p.

Specifically, the binomial distribution models a series of *Bernoulli trials*, i.e., experiments with two possible outcomes, one of which is considered a success and the other a failure. For example, the binomial distribution allows us to calculate the probability that we will obtain 5 heads when flipping a coin 10 times.

The binomial distribution has two parameters: n and p:

$$P(X = s) = \binom{n}{s} p^s (1-p)^{n-s}$$

Below is a plot of the PMF for $n = 10$ and $p = 0.5$:



Normal Distribution

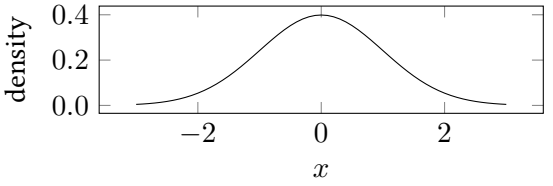
The normal distribution is a continuous probability distribution that is often used to model the distribution of real-valued random variables. For example, physical measurements such as height and weight tend to be normally distributed.

The normal distribution has two parameters: μ and σ^2. Recall that μ describes the mean of the distribution and σ^2 its variance.

The PDF of the normal distribution is given by:

$$\mathcal{N}(x|\mu,\sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

Below is a plot of the PDF for $\mu = 0$ and $\sigma^2 = 1$:



The *standard normal distribution* is a normal distribution with $\mu = 0$ and $\sigma^2 = 1$.

Every normal distribution can be transformed into a standard normal distribution (*standardized*) by applying the following transformation:

$$z = \frac{x - \mu}{\sigma},$$

where z is the transformed variable, x is the original variable, μ is the mean of the original distribution, and σ is its standard deviation.

Chapter 8

Evaluation

Confusion Matrix

A confusion matrix is a table that shows the number of true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN) predicted by a classifier.

	Gold Labels	
	Positive	**Negative**
Predicted Positive	TP	FP
Predicted Negative	FN	TN

Accuracy

The accuracy of a classifier is the proportion of correctly classified instances:

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

Precision and Recall

Precision and recall are two alternative measures that are often used to evaluate the performance of a classifier.

Precision is the proportion of instances that the classifier labeled as positive that are actually positive:

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Recall is the proportion of actual positive instances that the classifier labeled as positive:

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

F1-score

The F1-score is the harmonic mean of precision and recall:

$$F_1 = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Compared to accuracy, the F1-score is a better evaluation measure for imbalanced test sets, i.e., test sets where one class is much more frequent than the other.

Micro-Averaged and Macro-Averaged F1-score

If the test set contains multiple classes, one can compute the F1-score for each class separately and then average the results. This is the *macro-averaged F1-score*, which gives equal weight to each class, regardless of its frequency.

Alternatively, one can compute precision, recall, and then the F1-score globally by counting the total true positives, false negatives, and false positives across all classes. This is called the *micro-averaged F1-score*. The micro-averaged F1-score focuses on the overall performance and weights the impact of each class by its size.

T-Test

Student's t-test is a statistical test that can be used to determine whether the difference between two means is statistically significant. In computational linguistics, t-tests are sometimes used to compare the accuracy of two classifiers, and to determine whether the difference is significant.

The test is called *Student's t-test* because it was developed by statistician who wrote under the pseudonym "Student".

As a statistical hypothesis test, the t-test requires a null hypothesis and an alternative hypothesis. In the case of comparing two classifiers, the null hypothesis is that the two classifiers have the same accuracy, and the alternative hypothesis is that one classifier is more accurate than the other. We can only reject the null hypothesis if we find it to be very unlikely given the observed data.

By choosing a significance level α, we determine how unlikely the null hypothesis must be in order to reject it. The significance level is the error risk, i.e., the probability of rejecting the null hypothesis when it is actually true.

The t-test returns a p-value, which is the probability of observing the data when assuming that the null hypothesis is true. If the p-value is smaller than the significance level we have chosen, we reject the null hypothesis. If the p-value is larger, we cannot reject the null hypothesis.

Bootstrapping

Bootstrapping is a technique for estimating the variance of an evaluation metric (e.g., accuracy or F1-score) by repeatedly sampling with repetition from a dataset and computing the metric on the sample.

Like the t-test, bootstrapping can be used to determine whether the difference between two classifiers is statistically significant. A p-value can be computed and used to determine whether the difference is significant.[1]

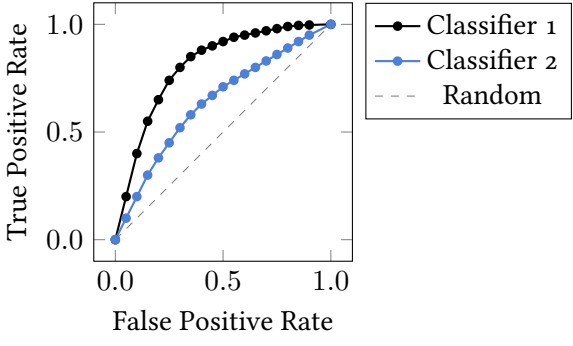
[1] Compared to the t-test, bootstrapping does not require the assumption that the data is normally distributed. This is called a *non-parametric* test.

ROC Curve

In binary classification, a single accuracy score is sometimes not meaningful, because it depends on the specific threshold that is used to convert the classifier's output into a binary decision.

If we choose a rather low threshold, the classifier will classify many instances as positive, resulting in a high true positive rate, but also a high false positive rate. Conversely, a high threshold will result in a low true positive rate and a low false positive rate.

In such a case, we plot a curve for different thresholds:



The two lines in the plot are made up of points that correspond to different thresholds.

According to the plot, classifier 1 is better than classifier 2: Its line is closer to the top-left corner, indicating that for any threshold, classifier 1 has a higher true positive rate and a lower false positive rate than classifier 2.

A classifier that outputs random decisions will result in a diagonal line from the bottom-left to the top-right corner.

For historical reasons, this curve is called the ROC curve (*receiver operating characteristic*).

Area Under the ROC Curve

Above, we visually compared the two classifiers on the ROC curve. There is also a numerical way to compare classifiers using the ROC curve: the AUC (*area under the curve*) of the ROC.

The AUC is a number between 0 and 1, and a random classifier has an AUC of 0.5. The higher the AUC, the better the classifier.

Chapter 9

Linear Functions

Definition of a Vector

A vector is a tuple of numbers:

Vectors can be written as rows or columns:

$$\mathbf{v} = \begin{pmatrix} 1 & 2 & 3 \end{pmatrix} \quad \text{and} \quad \mathbf{w} = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$$

Transposition is the operation that turns a row vector into a column vector and vice versa:

$$\mathbf{v}^T = \mathbf{w} \quad \text{and} \quad \mathbf{w}^T = \mathbf{v}$$

The individual numbers in a vector are called *scalars*.

The *dimensionality* n of a vector is the number of scalars it contains. A vector can be interpreted as a point in n-dimensional space.

Vector Operations

Vectors can be added and subtracted from each other:

$$\mathbf{v}+\mathbf{w}=\begin{pmatrix}v_1\\ \vdots\\ v_n\end{pmatrix}+\begin{pmatrix}w_1\\ \vdots\\ w_n\end{pmatrix}=\begin{pmatrix}v_1+w_1\\ \vdots\\ v_n+w_n\end{pmatrix}$$

If we multiply a vector with a scalar, we multiply each of its components with that scalar:

$$\lambda\mathbf{v}=\lambda\begin{pmatrix}v_1\\ \vdots\\ v_n\end{pmatrix}=\begin{pmatrix}\lambda v_1\\ \vdots\\ \lambda v_n\end{pmatrix}$$

Addition, subtraction and scalar multiplication can be described as *element-wise* operations, since they are performed on each element of the vector separately.

Components of a Linear Function

A linear function is a function of the form:

$$f(x_1,\ldots,x_n) = a_1x_1 + \ldots + a_nx_n + b,$$

where $a_1,\ldots,a_n$ are called the *weights*, *coefficients* or *parameters* of the function and b is called its *bias* or *intercept*.

Such a function is called *linear* because the inputs are not multiplied with each other or with themselves, but only with constants. This is called a *linear combination* of the inputs.

Both the inputs and the weights of a linear functions can be represented as vectors:

$$\mathbf{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} \quad \text{and} \quad \mathbf{a} = \begin{pmatrix} a_1 \\ \vdots \\ a_n \end{pmatrix}$$

Dot Product

The dot product of two vectors $\mathbf{v}$ and $\mathbf{w}$ is defined as follows:

$$\mathbf{v} \cdot \mathbf{w} = \sum_{i=1}^{n} v_i w_i = v_1 w_1 + \ldots + v_n w_n$$

An alternative notation is:[1]

$$\mathbf{v} \cdot \mathbf{w} = \mathbf{v}^T \mathbf{w}$$

We can use the dot product to write the linear function in a more compact form:

$$f(\mathbf{x}) = \mathbf{a} \cdot \mathbf{x} + b$$

The expression can be further simplified by "absorbing" the bias into the weights and appending a 1 to the input vector:

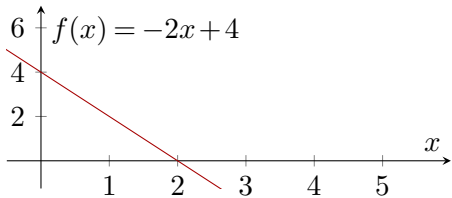
$$\mathbf{a'} := \begin{pmatrix} a_1 \\ \vdots \\ a_n \\ b \end{pmatrix} \quad \text{and} \quad \mathbf{x'} := \begin{pmatrix} x_1 \\ \vdots \\ x_n \\ 1 \end{pmatrix}$$

$$\mathbf{a} \cdot \mathbf{x} + b = \mathbf{a'} \cdot \mathbf{x'}$$

[1] This alternative notation comes from matrix algebra, which is not covered in this course.

Linear Equations

If we plot a linear function, we get a straight line. Here's an example for a linear function in two-dimensional space:



To describe the line directly, we convert the linear function into a linear equation. We assign x to x_1 and $f(x)$ to x_2 (one variable for each dimension), and then rearrange:

$$x_2 = -2x_1 + 4$$
$$2x_1 + x_2 - 4 = 0$$

The red line in the plot above is the set of points (x_1, x_2) that satisfy this equation.

If the linear equation has three variables, it describes a plane in three-dimensional space.

In higher-dimensional space (an equation with $x_1, \ldots, x_n$), the set of points that satisfy the equation is called a *hyperplane*.

Linear Classifiers

The line described by a linear equation can be used for classifying points: If a point is above the line, it is classified as positive; if it is below, it is classified as negative.

The classifier function $f(x_1, x_2) = 2x_1 + x_2 - 4$ returns a positive value for points above the line and a negative value for points below.

We can look at the *sign* of the value to convert the classifier function into a binary classifier:

$$\text{sign}(x) = \begin{cases} 1 & \text{if } x > 0 \\ -1 & \text{if } x \leq 0 \end{cases}$$

$$\begin{aligned} \text{class}(x_1, x_2) &= \text{sign}(f(x_1, x_2)) \\ &= \text{sign}(2x_1 + x_2 - 4) \end{aligned}$$

In vector notation:

$$\text{class}(\mathbf{x}) = \text{sign}(\mathbf{a} \cdot \mathbf{x}),$$

where

$$\mathbf{a} = \begin{pmatrix} 2 \\ 1 \\ -4 \end{pmatrix} \text{ and } \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ 1 \end{pmatrix}$$

Index

Template by François Fleuret.

Version 2024.04.06