Count Modeling with Sum-Product Networks for Syndromic Surveillance

Modellierung von Fallzahlen mit Sum-Product Netzwerken für die syndromische Überwachung von Krankheitsausbrüchen

Bachelor thesis by Bennet Wittelsbach

Date of submission: February 16, 2021

1. Review: Dr. Eneldo Loza Mencía
2. Review: Moritz Kulessa
Darmstadt

TECHNISCHE UNIVERSITÄT DARMSTADT

Computer Science Department
Knowledge Engineering Group

Erklärung zur Abschlussarbeit gemäß §22 Abs. 7 und §23 Abs. 7 APB der TU Darmstadt

Hiermit versichere ich, Bennet Wittelsbach, die vorliegende Bachelorarbeit ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die Quellen entnommen wurden, sind als solche kenntlich gemacht worden. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Mir ist bekannt, dass im Fall eines Plagiats (§38 Abs. 2 APB) ein Täuschungsversuch vorliegt, der dazu führt, dass die Arbeit mit 5,0 bewertet und damit ein Prüfungsversuch verbraucht wird. Abschlussarbeiten dürfen nur einmal wiederholt werden.

Bei der abgegebenen Thesis stimmen die schriftliche und die zur Archivierung eingereichte elektronische Fassung gemäß §23 Abs. 7 APB überein.

Bei einer Thesis des Fachbereichs Architektur entspricht die eingereichte elektronische Fassung dem vorgestellten Modell und den vorgelegten Plänen.

Darmstadt, 16. Februar 2021

B. Wittelsbach

Contents

List of Figures

List of Tables

1. Introduction

Disease outbreaks and possible consequent epidemics present a serious risk to humanity and moreover may affect nearly all humans directly or indirectly in a globalized world, as the 2019 coronavirus crisis demonstrates. Wide-spread diseases, caused by organisms called pathogens, expose several threats, individually and collectively, from higher morbidity and mortality to changes in individual social behavior and mental well-being, and eventually long-term decline in economic growth [39, 41]. Therefore, the question of how to detect novel disease outbreaks in the public sector is becoming increasingly important.

The earliest known epidemic is dating back to around 430 BC, when an estimate of up to a quarter of the population of Athens died of probably a typhus infection [30]. The second large-scale occurrence of the Bubonic plague, known as Black Death, led to the death of an estimate of one fifth of Europe's total population in the time between 1347 to 1351 and a long-term economic downturn [24, 18]. Pathogens may even be used as biological weapons, as it has happened in the United States in 2001, when several politicians and journalists were infected with anthrax after receiving letters containing respective spores [4].

The spreading and mutation rates of pathogens correlate with the number of possible exposures between contagious and healthy persons. Due to growing populations, urbanization and globalization, which lead to more densely populated areas that are highly connected, disease-causing agents may spread faster in and between populations. This can be best exemplified at the current pandemic caused by the new coronavirus strain: At 31.12.2019, several people were hospitalized with pneumonia of unknown causes in the city of Wuhan, China [33]. The pathogen was isolated in China at 07.01.2020, and identified as a new strain of the coronavirus family, known as SARS-CoV-2 and its associated disease COVID-19. One month later, the WHO announced 9826 confirmed cases of COVID-19 in different regions in China and 19 other countries, including neighbors like Vietnam and Australia, but also faraway ones like Germany and the United States of America, which are leading trade partners of China [32, 47]. After 1 year of its identification, the new virus infected over 83 million and killed around 1.8 million humans in all countries of the world [1] [31].

Without any containment measures, the spread rate of such pathogens may be of exponential order. Therefore, quick detection of new infection-clusters in real-time (e.g. a daily basis) is crucial to be able to act and deploy such measures. In contrast to the many earlier epidemics, today, industrialized countries track medical records of patients, which allows us to develop mechanisms that can detect novel outbreaks of diseases by monitoring health data. Previous work has shown that early outbreak detection systems indeed are able to identify infectious clusters in public health data [16]. To implement such systems, a data baseline of historic medical records has to be constructed, that models the "status quo" of a medical institution or a region. But measuring diseases or pathogens directly is not practicable - a diagnose is often not available when a patient is admitted in a hospital, and for novel ones there cannot be a sufficient one - and they have to rely on reported symptoms and disease-related attributes in the patient data like gender or home district.
This task can be defined as an anomaly detection problem and, regarding health issues, is known as syndromic surveillance. In this context, syndromes are general, possible disease patterns in health data, e.g. the reported

[1] Except Turkmenistan and North Korea, who have not reported any cases ever.

symptom and the patient's age form a syndrome, or a purchased drug and the district of the person buying it. With aggregating daily counts of a set of syndromes over each patient, the aforementioned data baseline can be established and used to monitor the condition of public health by comparing counts of a new instance against the learned baseline. If the observed count of a syndrome appears to be suspicious with respect to the model, the user, e.g. medical staff, will be alarmed and can check if a novel disease outbreak may be present or other circumstances can explain the anomaly sufficiently.

When dealing with public health data, it has to be considered that the underlying distributions depend on a set of environmental variables, for example the day of week, weather or seasons, with the latter being correlated with e.g. the spreading rate of the influenza virus that leads to higher reported counts of respiratory symptoms in and around February. So, health datasets are usually heterogeneous and expose dependencies among their attributes, in particular the reported patient records depend on environmental conditions. Hence, it is essential to determine the context of new daily counts and to choose the appropriate clusters as baseline data to compare new observations against historic data that was collected under the same circumstances.

To address this issue, we use Sum-Product Networks (SPNs), which are hierarchical probabilistic models [37, 35, 34], that model the joint probability distribution of data by learning factorizing mixture models. They allow to learn a computationally efficient, fine-grained tree-like structure of coherent clusters in tabular data like medical records and to evaluate new data entries with respect to the environmental attributes in an interpretable manner.

In the next chapter, we deal with the foundations of the epidemiological domain and the machine learning domain, with a focus on statistical methods, on which the approach of this thesis is based. In chapter 3, we explore the emergence of this research area and what other types of syndromic surveillance systems were developed to tackle the problem of early disease outbreak detection. This leads us to the methodology of this thesis in chapter 4, where we outline the pre-processing of the public health data we use for syndromic surveillance, how this setting differs from typical machine learning problems and why Sum-Product Networks are well-suited to handle this task. The experimental results and their interpretations are discussed in Chapter 5. We take a look at possible shortcomings of the presented approach and also discuss how well different evaluation methods approximate underlying empirical distributions. In the following conclusion, we provide a summary of the presented work, reflect the contributions of this thesis and what actions can be taken to further improve outbreak detection in public health domains.

2. Foundations

2.1. Epidemiology

According to the "Dictionary of Epidemiology" by John Last[22], epidemiology is "the study of the distribution and determinants of health-related states or events in specified populations, and the application of this study to the control of health problems". In other words, epidemiology is a data-driven science, which aims to identify underlying correlations and causes of diseases and other health-related issues. To reliably assess all kinds of questions concerning human well-being, epidemiology needs to quantify large collections of health data and builds upon sound statistical methods. A wide-spread inferential approach is the traditional framework of hypothesis testing. This allows epidemiological researchers to evaluate the compatibility of observations with testable assumptions and reason about the influence of diseases, harmful particles, and biologic weapons as well as overall health statuses and medical evaluations in general, usually by comparing stratified populations[1].

Pathogens and Diseases

Pathogens are any kind of infectious agents that cause diseases in other lifeforms. This include germs like bacteria and fungi, but also toxic entities like viruses and bigger organism like maggots. To get infected, a host needs to be exposed to a pathogen, whereupon pathogens can then use the host's resources for reproduction. Bacteria and fungi nourish from the host and their number can grow exponentially by cellular division. They usually release exotoxins, which are responsible for diseases and respective symptoms by damaging the host's cells or metabolism. Whereas viruses force some of the host's cells to produce replica based upon the injected blue-print of the virus, until the cell dies and releases new viruses that infect neighbours.
Host-to-host infection usually happens through transmission by contact with skin, body fluids, or aerial particles. To reliably identify pathogens or diseases, clinical tests have to be conducted, which analyze samples from the host organism and search for abnormalities. As this requires relatively many resources and takes time up to multiple days, diseases are often diagnosed preliminary by means of observed symptoms that are known to be likely associated with a particular disease.

Public Health Data

Health data generally describes data concerning health-related issues like medical records of patients, the over-all status of health aspects within a region, food inspections or screening results of commercial products.

[1] Stratification means that the populations we want to compare exhibit the same attributes and ratios regarding personal and environmental characteristics. If we can ascertain an effect between two groups due to some observations, we want to be sure that it does not stem from any influence from unaccounted confounders.

More specific, this may be the initial assessments of patients that are hospitalized in an emergency room and contain information about the date of admission, age, home-district, the reported symptom and others. It can also be the nation-wide monitoring of cases of a dangerous disease, which would describe the change of cases per state over time. This kind of health data is collected within a temporal and spatial context and therefore most often only supports inference in that context.

Public health data is usually raised and administrated by medical institutions and state authorities. In Germany, the Robert Koch-Institut (RKI) is a federal agency for public health care and responsible for disease control. It is entrusted with national and regional health reports; identification, prevention and combating of infectious and non-infectious diseases; epidemiological investigations of such diseases including assessments of risks and their documentation; and risk assessment of genetically modified organisms and products.

2.1.1. Syndromic Surveillance

The central concern of syndromic surveillance is the early detection and investigation of disease outbreaks [16]. This is a fairly new research area which relies on large collections of electronic records of public health data. In 1998, the Centers for Disease Control and Prevention (CDC), a federal agency under the U.S. Department for Health and Human Services, released a plan to "combat today's infectious diseases and prevent those of tomorrow" by developing "new mechanisms for detecting, evaluating, and reporting suspicious events" [5, 3]. *Early* detection means that we want to find clusters of infections as soonish as they emerge and use the fastest available health-related assessments, before official diagnoses and large studies are conducted. Subsequently, a range of surveillance systems in different scopes were designed: for capturing trends in non-medical, health-related data like absence in workplaces and schools, for assuring that no disease outbreak has occurred at large-scale events like Olympia, or for risk estimation of pathogens that are found in ill or dead animals [1]. Another creative approach was to predict the onset of the yearly flu-season by monitoring the number of flu-related search queries [11]. However syndromic surveillance systems are not meant to replace traditional health monitoring, but aim to complement it.

Syndromes, in a general sense, are possible disease patterns. The chance of getting infected with a disease strongly depends on different variables. First of all, to be infected by the disease, a person needs to be exhibited to another, ill organism or the pathogen itself in a specific time and place. In this work, we additionally refer to indicators, such as the number of reported cases in a particular area, as syndromes, since a high observed count of these also allow to detect infectious disease outbreaks within that region. Syndromes include medical or health-related reports, like the symptom a physician diagnoses, the medicines purchased in local drug stores, or the absence frequency in schools and workplaces. Additionally, we can model attributes like age or home-district as disease patterns, because it could be that an outbreak occurs in some neighborhood or only affects people within a certain age range.
We should also think of the cases where the observations of simple indicators like the gender or the age itself is not suspicious, but the combination of some values of both can be. Let's assume that a new disease emerged and infects primarily men in their 30's, but only infected a few in a city. The count of infected men and infected people in their 30's may be unsuspicious, but if we monitor the combination of both, we could detect anomalies in the respective cities and act accordingly. If we observe every possible combination of disease patterns, we call this approach unspecific syndromic surveillance. In practice, including all possible combinations of many syndromes is a combinatorial problem and not feasible. Therefore we usually constraint our monitoring model to syndromes of a fixed size, like pairs of observations.

ESEG

The ESEG project (ger: Erkennung und Sicherung Epidemischer Gefahrenlagen, eng: recognition and safeguarding of epidemic risk situations) aims to utilize public health data from emergency rooms and other sources to identify the occurrence of outbreaks and infectious events at an early stage and to be able to quickly initiate infection control measures.
Within this project, this thesis examines if modeling medical records with probabilistic deep learning approaches like SPNs, that allow fine-grained clustering of such heterogeneous data, can improve upon existing surveillance systems that are already in use. For this project, the RKI partnered with the Knowledge Engineering Group at TU Darmstadt, to developed methods to link different sources of health data into a coherent one[13, 15, 40], which is essential for a comprehensive view of the current public health condition, and how to further improve the detection capability of syndromic surveillance systems[20, 21].

2.2. Stochastic

Stochastic generally describes events or systems that can be modeled with probability distributions. It is also a broader term that unifies probability theory and statistic. Statistic in general is the application of mathematical methods to describe and analyse collections of data, usually with the goal of inferring knowledge from the data or predicting future events. Probability theory is the branch of mathematics that defines and applies probabilities in a rigorous mathematical manner. Computer scientists often may lack of statistical training and deep understanding of probabilistic concepts, so the very foundations of probability theory will be revisited, on which concepts of this thesis like the definition, structure-learning and inference procedures of Sum-Product Networks rely on. Afterwards, statistical methods for dealing with count data and finding anomalies in data generally will be introduced.

2.2.1. Probability Theory

While different interpretations of probability and hence consequences in the respective inferential approaches exist, they share common approaches to define the basic concepts. One possible axiom set for dealing with probabilities is the set of the Kolmogorov axioms, which commonly are used to define probability in the sense of set theory and events. The following definitions are required to construct a well-defined probability space and are based upon lecture notes from "Mathematik für Informatiker III" from professor Pfetsch at the TU Darmstadt[36] and Hartmann's "Mathematik für Informatiker" [14]. They are not required to actually implement the proposed syndromic surveillance system - most statistical computation methods are implemented by common frameworks in a sophisticated manner anyway -, but may help to reason about the following concepts.

Random Variables and Probability Distributions

Definition 2.2.1 (Random variable)**.** Let (Ω, p) be a probability space. A random variable (RV) is a function $X : \Omega \to \mathbb{R}$, such that the preimage Ω is part of the σ-algebra $\mathcal{A}$ for every interval $I \subseteq \mathbb{R}$: $\{\omega \in \Omega | X(\omega) \in$

$I\} \in \mathcal{A}$.[2] The values that the image of X can take on are denoted by $values(X)$. The probability $p(X(\omega) \in I)$ will be denoted as $p(x_- \leq X \leq x_+)$, $p(X \leq x)$ or $p(X = x)$.

A revision of the definitions required to define a probability space is given in the appendix. The σ-algebra $\mathcal{A}$ is the set of all possible combinations of outcomes that are "permitted" in the probability space. As the probability space is defined in such a way that $\mathcal{A}$ is exactly the power-set of Ω, it is just defined that each event that is mapped into the space of real numbers by the RV actually exists in our set of combinations of outcomes. This would also be expected intuitively, but the strict definition gives rise to well-defined operations.

While random variables here are defined as a mapping onto the real numbers, they can be defined as a mapping onto an arbitrary field. For example, a categorical random variable can be constructed, so X can take on values from a set of categorical events.
Equipped with random variables, we can now define distribution functions. Probability distribution functions are monotonously increasing functions and probability measures with $\lim_{x \to -\infty} f(x) = 0$, $\lim_{x \to \infty} f(x) = 1$. We distinguish between the probability mass and probability density functions, which assign each value or each range of values a probability for their occurrence, opposing to the cumulative distribution function, which returns the aggregated probability that the observed value or less is observed.

Definition 2.2.2 (Cumulative distribution function)**.** Let $X : \Omega \to \mathbb{R}$ be a random variable on (Ω, p). The function

$$f_X : X(\Omega) \to [0, 1], f_X(x) = p_X(X \leq x), x \in values(X) \tag{2.1}$$

is called cumulative distribution function (CDF).

Definition 2.2.3 (Probability mass function)**.** Let X be a discrete random variable on (Ω, p). The probability measure

$$p_X : X(\Omega) \to [0, 1], p_X(x) = p_X(X = x), x \in values(X) \tag{2.2}$$

is called probability mass function (PMF).

The PMF returns the probability that the associated RV takes on a specific, discrete value x. $p_X(x)$ is usually denoted as $p(x)$, $f_X(x)$ as $f(x)$. The CDF of a discrete RV is a staircase function, which has jumps of the height $p(X = x_i)$ at the points x_i. It is given by

$$f_X(x) = \sum_{x' \in \{values(X) \leq x\}} p_X(x') \tag{2.3}$$

Definition 2.2.4 (Probability density function)**.** Let X be a continuous random variable on (Ω, p) and $f_X(x)$ the CDF of X. If there exists an non-negative integrable function $p_X(x)$, such that

$$f_X(x) = \int_{-\infty}^{x} p_X(x)dx = p_X(X \leq x), x \in values(X) \tag{2.4}$$

then $p_X(x)$ is called the probability density function (PDF) of X.

[2] More formally, a random variable maps from one measurable space into another. But this requires to introduce some measure theory with additional definitions, and the given formalism shall be sufficient for this thesis.

Definition 2.2.5 (Joint distributions)**.** Let $\mathbf{X} = \{X_1, ..., X_n\}$ be a set of random variables on the same (Ω, p). The joint probability distribution of $\mathbf{X}$ is a distribution that encodes the probability that each X_i is realized by $x_i \in values(X_i)$. It is denoted as

$$p_{\mathbf{X}}(\mathbf{x}) = p_{X_1,...,X_n}(x_1, ..., x_n) = p(x_1, ..., x_n). \tag{2.5}$$

The joint CDF is then given by

$$f_{\mathbf{X}}(\mathbf{x}) = p_{\mathbf{X}}(X_1 \leq x_1, ..., X_n \leq x_n) \tag{2.6}$$

and the joint PMF by

$$p_{\mathbf{X}}(\mathbf{x}) = p_{\mathbf{X}}(X_1 = x_1, ..., X_n = x_n). \tag{2.7}$$

The joint PDF for continuous $\mathbf{X}$, if $p_{\mathbf{X}}$ exists, is defined in an analogous way and the specific realizations x_i are replaced with suitable ranges.

Definition 2.2.6 (Marginal distribution)**.** Let $\mathbf{X} = \{X_1, ..., X_n\}$ be a set of continuous random variables on the same (Ω, p), $\mathbf{Y} \subset \mathbf{X}$, $\mathbf{Z} = \mathbf{X} \setminus \mathbf{Y}$ with $\mathbf{z} = \{z_1, ..., z_k\}$ and $p_{\mathbf{X}}$ the respective joint probability distribution. Then, the marginal distribution $p_{\mathbf{Y}}$ is defined as

$$p_{\mathbf{Y}}(\mathbf{y}) = \int_{\mathbf{z}} p_{\mathbf{X}}(\mathbf{y}, \mathbf{z}) d\mathbf{z} \tag{2.8}$$

The same holds for discrete random variables, where the integrals are replaced with respective sums.

So, marginalizing variables out of a joint distribution means that we integrate (or sum) over the probabilities of all occurrences of the marginalized variables in the joint distribution and obtain the probability distribution that just describes the marginalized random variable(s) $\mathbf{Y}$ and does not consider any of the dropped $\mathbf{Z}$.

Definition 2.2.7 (Conditional distribution)**.** Let $\mathbf{X}$, $\mathbf{Y}$, $\mathbf{Z}$, p be defined as in 2.2.6, but the random variables X_i can be discrete or continuous. The conditional distribution is defined as

$$p_{\mathbf{Y}|\mathbf{Z}}(\mathbf{y}|\mathbf{z}) = \frac{p_{\mathbf{X}}(\mathbf{y}, \mathbf{z})}{p_{\mathbf{Z}}(\mathbf{z})}, p_{\mathbf{Z}}(\mathbf{z}) \neq 0. \tag{2.9}$$

The conditional distribution describes the distribution of RV $\mathbf{Y}$ given that they occur together with the RV $\mathbf{Z}$. This can be interpreted as first observing $\mathbf{z}$ with the probability $p_{\mathbf{Z}}(\mathbf{z})$ and then asking how likely it is to see $\mathbf{y}$. In terms of tabular data, the conditional distribution of $\mathbf{Y}$ is the estimated distribution over the instances that contain the values $\mathbf{z} \in \mathbf{Z}$.

Definition 2.2.8 (Statistical independence)**.** Let X and Y be discrete random variables with respective $x \in values(X)$ and $y \in values(Y)$. X and Y are said to be statistically independent, if and only if

$$p_{XY}(X = x, Y = y) = p_X(X = x)p_Y(Y = y), \forall x, y \tag{2.10}$$

In the continuous case, the following has to be true:

$$p_{XY}(X \leq x, Y \leq y) = p_X(X \leq x)p_Y(Y \leq y), \forall x, y \tag{2.11}$$

This can also be generalized from the bi-variate to the multi-variate case. Then, the probability of the joint distribution needs to equal the multiplication of all involved RV for all respective values. Statistical independence also implies that the conditional distribution of independent RVs is the same as the associated marginal distribution, because they do not influence each other:

$$p_{XY}(X = x, Y = y) = p_X(X = x)p_Y(Y = y) \forall x, y \rightarrow p_{Y|X}(y|x) = p_Y(y), p_{X|Y}(x|y) = p_X(x) \tag{2.12}$$

Definition 2.2.9 (Expectation)**.** Let X be a continuous random variable with probability distribution p_X. The expectation of X is given by

$$\mathbb{E}[X] = \int_{x \in values(X)} x p_X(x) dx = \mu. \tag{2.13}$$

The integral simplifies to $\sum_{x \in values(X)} x p_X(x)$ in the case of discrete RV. The expectation of a set of observations is called (arithmetic) mean and denoted as $\bar{x}$. The expectation of a distribution is also called mean or statistical moment of first order and usually denoted as μ.

The expectation of RVs is linear in addition and also in multiplying if the RV are statistically independent:

$$\mathbb{E}[X + Y] = \mathbb{E}[X] + \mathbb{E}[Y] \tag{2.14}$$

$$\mathbb{E}[XY] = \mathbb{E}[X]\mathbb{E}[Y], \text{ if X and Y are statistically independent} \tag{2.15}$$

$$\mathbb{E}[XY] = \int_{x \in values(X)} \int_{y \in values(Y)} p_{XY} xy(x, y) dx dy, \text{ else} \tag{2.16}$$

Definition 2.2.10 (Variance)**.** Let X be a RV and $\mathbb{E}[X]$ the expectation of X. Then, the variance of X is defined as

$$Var[X] = \mathbb{E}[(x - \mathbb{E}[X])^2] = \mathbb{E}[X^2] - \mathbb{E}[X]^2 = \sigma_X^2 \tag{2.17}$$

The variance is the squared expectation of the difference of the RV to its expectation. It is a common measure of expected spread of a variable in relation to its mean. The square root of the variance is called the standard deviation and denoted as σ_X. It also describes the spread of a probability distribution and additionally shares the same "units" as its RV. If two RV X, Y are independent and measured by the same probability distribution, addition of variance is also linear:

$$Var[X + Y] = Var[X] + Var[Y] \textit{ if X and Y are statistically independent} \tag{2.18}$$

Parametrization

If a probability distribution can be described by some summary statistics - in the case of the Normal distribution, that will be introduced soonish, these are the expectation and the variance -, then the set of summarizing statistics will be called parameters and denoted as θ. In contrast, non-parametrized probability distributions are distributions that do not depend on parameters, but on a whole data-set[3]. The categorical distribution,

[3]The distinction between parametrized and non-parametrized models can be somewhat unintuitive at first sight. While parametrized distributions rely on summarizing statistics of the data and bound the complexity of the model by the number of parameters, non-parametrized distributions rely on all observed data-points and can be - roughly speaking - be seen as parametrized by every single data point. So, non-parametrized models do not rely on summarizing statistics, but use the available data as it is and accordingly grow in complexity.

which assigns each event its probability, is an example for non-parametrized distributions. Generally, they represent the data in a complete manner. To actually conduct inferential statistics, we may need to summarize our data. If we have two datasets that can be described by two realizations θ_1, θ_2 of the same probability distribution, we can use the parameters of the respective distributions to compare them and carry out further statistical studies[4].

With parametrized probability distributions, the question of how to estimate these parameters arrives naturally. First, we need a metric to define what a good fit of parameters for estimating a probability distribution from a sample of data actually means. This metric is called likelihood.

Definition 2.2.11 (Likelihood)**.** Let $x_1, ..., x_n$ be observed data-points and let θ be a set of parameters. Let p_X^θ be a probability distribution of the random variable X parametrized by θ. The likelihood function of this probabilistic model with respect to the observations is defined as

$$\mathcal{L}_X(\theta; x_1, ..., x_n) = p_X^\theta(x_1)p_X^\theta(x_2)...p_X^\theta(x_n) \tag{2.19}$$

The likelihood function is often denoted as $L(\theta) = p(X|\theta) = p(x_1, ..., x_n|\theta)$.
Now, let Θ be the set of all permitted parameters and $\hat{\theta}$ be a parameter. If

$$\mathcal{L}_X(\hat{\theta}; x_1, ..., x_n) \geq \mathcal{L}_X(\theta; x_1, ..., x_n), \forall\theta \in \Theta, \tag{2.20}$$

then $\hat{\theta}$ is called the maximum-likelihood estimator of X.

$\hat{\theta}$ is the most likely parameter from which our observations could accordingly be drawn and provides the best "description" or "summarization" of our data regarding the chosen probability measure. The maximum-likelihood estimator can be obtained by solving

$$\hat{\theta} = \arg\max_\theta \mathcal{L}(\theta) = \arg\max_\theta p_X^\theta(X) = \arg\max_\theta \prod_{i=1}^{n} p_X^\theta(x_i) \tag{2.21}$$

This equation can be drastically simplified (especially in the computational sense) by taking the logarithm of the likelihood-function, which transforms the logarithm of a product to the sum of the respective logarithms, and thus maximizing the log-likelihood also yields the maximum-likelihood estimator:

$$\hat{\theta} = \arg\max_\theta \log \mathcal{L}(\theta) = \arg\max_\theta \log p_X^\theta(X) = \arg\max_\theta \sum_{i=1}^{n} p_X^\theta(x_i) \tag{2.22}$$

This is permitted as the logarithm is a monotonously increasing function and the maximum of the logarithm is given by the same parameter as the maximum of the original function.

Definition 2.2.12 (Mixture distribution)**.** Let X be a random variable and $p_{X_1}, ..., p_{X_n}$ probability distributions on X. Let $w_{X_1}, ..., w_{X_n}$ be weights, such that each $w_{X_i} \geq 0$ and $\sum_{i=1}^{n} = 1$. A finite mixture distribution of X is defined as

$$f_X(x) = \sum_{i=1}^{n} w_{X_i} p_{X_i}(x). \tag{2.23}$$

[4]Of course we can also compare the distributions of empirical datasets. But as the samples are only a partial observation and also may be "porous", it might be benefiting to summarize the structure of the data by using parametrized distributions. (In frequentist statistics, we still only use the sample data to construct this distributions, but in the bayesian way, one can also "correct" observed data by incorporating prior knowledge.)

If all p_{X_i} represent the same probability measure and are parametrized by respective sets of parameters θ_i, we can also write the definition above as

$$f_X(x; \theta_1, ..., \theta_n) = \sum_{i=1}^{n} w_i p(x; \theta_i) \tag{2.24}$$

Estimating the parameters of a mixture of distributions θ_i is more difficult, as we also need to estimate the membership of observed data-points to one of the segmentated distributions. A simple approach to this problem is to define clusters of the data-points beforehand and simply estimate each distribution in the mixture from the respective data-points[5]. But, by defining the clusters before the parameter estimation, we actually might get results that do not coincide with our perception of the cluster's centres and the data-point's memberships. More sophisticated methods like the Expectation-Maximization algorithm [29] unite the clustering of data-points and the estimation of the maximum-likelihood parameters, thus allow robust estimation of whole mixture distributions with identification of latent clusters that are present in the data. The Expectation-Maximization algorithm will not be further investigated in this thesis and its application remains for future work.

Frequentist interpretation of probability

There exist different interpretations of the concept of probability with their associated schools. The two most common ones are the frequentist and the bayesian schools. While bayesians define probability as a *degree of belief* into an event, which allows them to incorporate prior expert knowledge into their probabilistic models, the frequentist approach interprets probability as expected frequency. Let us say we observe an industrial production line which produces some tools and we take a sample of 1000 tools, of which 50 are flawed. Then, the sample probability that a tool is flawed is 5%. If we now assume that we can repeat the production process under the exact same conditions infinitely many times, then we would expect every 50 out of 1000 tools to be flawed, so we have an expected frequency and a probability of 5%. The definitions and theorem provided above also follow the frequentist interpretation of probability, which will also be pursued in this thesis. Of course this view introduces some disputable concepts and consequences, but the old discussion between the bayesian and frequentist schools is beyond the scope of this work and further details are provided by many statistical books.
Nonetheless, in an intuitive way it often helps to think of expected frequencies, especially when we want to capture a *normal* frequency of observations and check if it gets violated. Also, we may not know the concrete data we are operating on (in the case of syndromic surveillance of emergency rooms, we actually might not know how diseases are locally distributed), so using the frequentist approach for syndromic surveillance is well justified.

2.2.2. Count Modeling

A statistical model attempts to describe dependencies and correlations between random variables by using stochastic methods. They consist of explanatory variables, whose values can be observed or chosen, and response variables, the objects of interest whose behavior we want to infer and gain knowledge about. We can

[5] One possible, simple cluster algorithm is the k-means algorithm, where we want to find k clusters (k is pre-defined) and decide the membership of data-points by the distance of the data-point to the estimated cluster centres. This approach will be presented in the section Machine Learning

also observe variables and reason about their relationship to earlier observations, like time-series, by using statistical tests. In the case of count modeling, either the response or all variables are simply counts.

Probability Distributions for Count Modeling

As we want to model counts of observed syndromes, we need probability distributions that are capable of representing count data. The Poisson distribution is very appealing for this task. It describes the probability of events that *can* occur often, but occur rather seldom, in a fixed time interval. The Normal distribution or Gaussian distribution is an universal distribution that emerges naturally when we observe large sets of (roughly) independent events. The negative Binomial distribution models the probability that a number of desired events happen before a number of undesired events occur. While the Poisson and the negative Binomial distributions directly model some form of count data, a *mixture* of Normal distributions can approximate any other distribution arbitrary well (by choosing an appropriate number of mixtures)[12].

Poisson Distribution

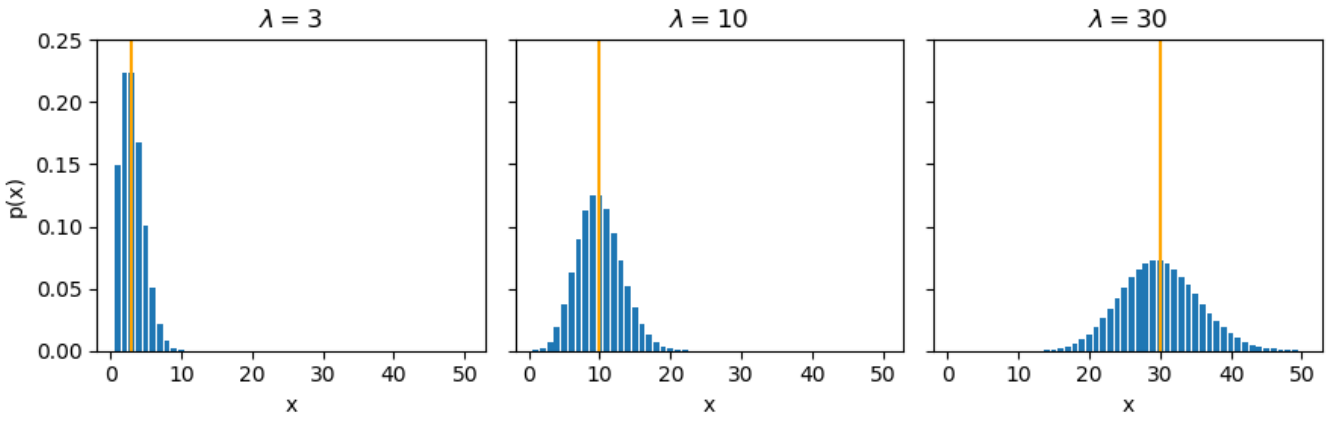

Figure 2.1.: Exemplary probability mass functions of the Poisson distribution
Exemplary Poisson PMFs for $\lambda \in \{3, 10, 30\}$. The x-axis denotes x, the y-axis p(x). The orange vertical lines show the means of the distributions, which are the respective λ. It can easily be seen that for larger λ, the form of the Poisson distribution converges to the form of the standard Normal distribution.

The Poisson distribution is a discrete distribution and models the independent occurrence of events in a fixed time-slot, given that there are many possible occasions for an event happening, n, with a low probability of that event, p. Traditional examples for the usage of Poisson distributions are telephone calls that arrive at a service center per hour, the Prussian officers that are kicked to death by their horses each year, or the number of decays of a radioactive element in a given time interval.

The Poisson distribution is described by only one parameter $\lambda \in \mathbb{R}$, which is the expected count of its random variables in a fixed time interval. The PMF of a Poisson distribution and its associated CDF are defined as

([17])

$$p_X(x;\lambda) = \frac{\lambda^x e^{-\lambda}}{x!} \tag{2.25}$$

$$f_X(x;\lambda) = \sum_{x'=0}^{x} p_X(x';\lambda) = \sum_{x'=0}^{x} \frac{\lambda^{x'} e^{-\lambda}}{x'!} = e^{-\lambda} \sum_{x'=0}^{x} \frac{\lambda^{x'}}{x'!}. \tag{2.26}$$

While the distribution per se is a discrete one, its parameter λ can be continuous. So we can construct a Poisson distribution over a random variable that has a mean of, say, 2.5, but naturally can only infer the probability of 2 or 3 occurring. Also, λ summarizes the mean and the variance of a Poisson distribution: $\lambda = \mu = \sigma^2$.
The maximum-likelihood estimation of λ is the mean of the sample population:

$$\hat{\lambda} = \arg\max_{\lambda} \log \mathcal{L}(\lambda) = \frac{1}{n} \sum_{i=1}^{n} x_i = \bar{x} \tag{2.27}$$

Normal Distribution

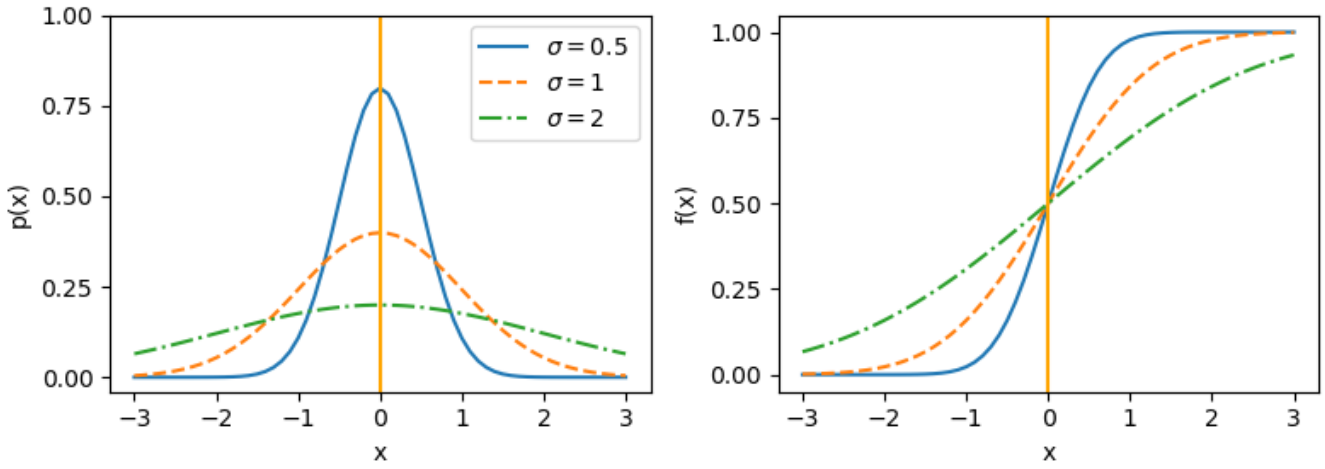

Figure 2.2.: Exemplary PDFs and CDFs of the Normal Distribution
Three Normal distributions with mean $\mu = 0$ and standard deviation $\sigma \in \{0.5, 1, 2\}$. Normal distributions are always symmetric and half of the values lie on each respective side of the vertical line at 0. If σ approaches 0, the normal distribution can degenerate and is just a "tight spike" around its mean, because of the inadequate standard deviation from samples with small sizes or no variation. The orange, dashed line with $\mu = 0$ and $\sigma = 1$ is also called the standard Normal distribution and its CDF is f_N (2.29).

The Normal or Gaussian distribution is a continuous distribution that naturally arises due to the central limit theorem (CLT). The CLT holds for a set of many RVs $X_1, ..., X_N$ that are all defined on the same probability space (Ω, p) and are all distributed identically by the same p_X, therefore the expectation and variance of all X_i are equal. Additionally, these RVs need to be statistically independent. Then, the distribution of the sum of all outcomes of all X_i approximates a normal distribution with the parameters $\mu_X = \sum_{i=1}^{n} \mu_{X_i} = n\mu_{X_i}$ and $\sigma = \sqrt{\sum_{i=1}^{n} \sigma_{X_i}^2} = \sqrt{n\sigma_{X_i}^2}$. This follows directly from the linearity of expectation (2.15) and variance (2.18)

of *independent and identically distributed* random variables (i.i.d. assumption). The following 3 equations describe the PDF, the CDF and the standardized CDF of the Normal distribution:

$$p_X(x;\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x-\mu}{\sigma})^2} \tag{2.28}$$

$$f_N(x;0,1) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{x^2}{2}} \tag{2.29}$$

$$f_X(x;\mu,\sigma) = f_N(\frac{x-\mu}{\sigma};0,1) \tag{2.30}$$

Let us look at an example: We throw 5 fair dices. Each dice is represented by a RV of which all have the same uniform distribution (each value $1-6$ is equally like). The dices do not influence the outcomes of other (except when they collide, what we will not model here), so their throws are independent. Each fair dice has an expectation of $\mu_i = 3.5$ and a variance of $\sigma_i^2 = 2.91667$. Due to the CLT, the sum of all dices should be approximately distributed by a Normal distribution with $\mu = 17.5$ and $\sigma^2 = 14.58333$, thus $\sigma = 3.81881$. This conclusion holds for any probability distribution that possesses finite expectation and variance, so if we sum up many independent observations like repeated experiments, we often can justify the usage of the Normal distribution.

The Normal distribution exhibits a few handy traits, but also a few we have to consider: First, it has infinite support. This means that *any* observation $r \in \mathbb{R}$ can *theoretically* occur, although the probability of events outside the range of 3σ around the mean quickly approaches 0. A very useful characteristic of Normal distributions is that they can be somewhat arbitrarily combined and the same rules still apply. The marginals of a joint Normal distribution are Normal distributed; the joint distribution of two Normal distributions is also Normal distributed. They are very flexible and can either be narrow, modeling clusters for which we are certain that we observe values around the mean, or be flat and therefore express uncertainty about the observation's locations. Upon these characteristics, we can build expressive Gaussian Mixture Models (GMM or MoG) that can easily be extended and diminished and, with the right amount of Gaussian mixture components, can approximate *any other probability distribution arbitrarily well* [12].

When we only want to estimate uni-variate Gaussians, the MLE is simply given by the sample mean ((2.31)) and the sample variance (2.32).

$$\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} x_i = \bar{x} \tag{2.31}$$

$$\hat{\sigma}^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2 \tag{2.32}$$

We can also use MLE with a Mixture of Gaussians, if we know the memberships of the observations to the mixture components. If not, the Expectation-Maximization algorithm allows to combine latent membership classification and parameter estimation, but it will not be discussed any further here.

Negative Binomial Distribution

The Negative Binomial distribution is a discrete probability distribution that models the repeated execution of independent and identically distributed Bernoulli experiments until a pre-defined number of successful

events occur. If the variance of a Negative Binomial distributed random variable is equal to its expectation, the Negative Binomial Distribution is equal to the Poisson distribution. It typically finds applications in count modeling scenarios where the variance is larger than the mean. It is also used in the field of epidemiology to model possible disease transmissions.

In terms of underlying Bernoulli distributions, the PMF is defined as:

$$f(x) = \binom{x+n-1}{n-1} p^n (1-p)^x, \tag{2.33}$$

where n is the number of expected successes and p is the respective Bernoulli distributed probability for each success.

For maximum-likelihood estimation, the number of successes n cannot be determined in analytical form, but needs to be approximated with numerical methods, which requires additional computational resources. However, the parameters of a Negative Binomial distribution can also be approximately computed from the parameters of a Gaussian distribution:

$$n_{NB} = \frac{\mu_G^2}{\sigma_G^2 - \mu_G}, \text{ if } \sigma_G^2 > mu_G \tag{2.34}$$

$$p_{NB} = \frac{n_{NB}}{n_{NB} + \mu_G} \tag{2.35}$$

If the variance of the Gaussian distribution is equal to or smaller than its mean, the term is replaced by a small value like 10^{-7}. This leads to to very large n and p near 1.

2.2.3. Hypotheses Tests

Hypotheses tests are a statistical method of inference and framework for assessing testable assumptions, the so called hypotheses, about a statistical model, built from a set of random variables. A falsifiable null hypothesis H_0 is constructed that shall describe what we believe to be true about our model until proven otherwise. We never can really tell if our provisional null hypothesis is actually true, but if we observe events that seem to be very improbable, we can reject the null hypothesis and state that our observation is not compatible with our previously built model. In the sense of Ronald Fisher, the inventor of *traditional* hypotheses testing, a significance level α should be determined before carrying out experiments, which determines a probability threshold for rejecting the null hypothesis and therefore our beliefs. After we have built our model X, we can assess the statistical significance of observations x by computing the probability that an outcome at least as extreme as the observed one can occur; this is called the p-value. If this value is less than the pre-specified α, we reject H_0 and call the observation statistical significant. Otherwise, if the value is not less than the significance threshold, we cannot simply accept H_0, because we do not have sufficient evidence. So, with hypotheses tests we are able to check if observations violate our statistical model.

A wide range of hypotheses test procedures for testing different assumptions of parameters and distributions are available. However, in this thesis we want to focus on null hypothesis significance testing, where we have a statistical model at hand and want to check if new observations come from the same distribution as the one we have estimated with our model. For this purpose, the p-value is computed. The p-value is the probability that an event at least as the observed one occurs. There exist two possibilities for significance testing: the

one-sided approach, where we want the likelihood that the observed value is extremely high *or* low with respect to our estimated distribution, and we reject the null hypothesis if the p-value is below the significance level α. In the one-sided case, the observation is only tested if it lies on the specified side of the distribution; values on the other side are not considered. With two-sided hypothesis testing, the observations may lie on both sides of the distribution and we want to check if it is extreme with respect to the estimation in any sense. When conducting this method, α needs to be divided by two for each side, to assure that the rejection zone in total is in the size of α.

To summarize, reject H_0, if the following is true, where (2.36) describes one-sided and (2.37) describes two-sided tests:

$$\text{left: } p(X \leq x) < \alpha \text{ , right: } p(X \geq x) < \alpha \tag{2.36}$$

$$p(X \leq x) < \frac{\alpha}{2} \text{ or } p(X \geq x) < \frac{\alpha}{2} \tag{2.37}$$

Other common tests for the Normal distribution allow for null hypotheses based on the relation of parameters and thus testing the distribution's mean or variance against observed statistics, like the Gauss-test or the "normal" chi-squared test. The latter is not to be confused with contingency tests based on the χ^2 distribution, which allows to test for statistical independence, or homogeneity tests, which assess if two samples come from the same generating distribution.

Combining p-values

When we use p-values and conduct combined hypotheses tests on the same data, the total rejection power of the combined tests has to be considered. If we want to combine two tests of two statistically independent random variables, the multiplication of the respective p-values would drastically understate the true p-value of the combined test. Of course we could just try to directly test the two RV w.r.t to their joint distribution in one test together, but this requires the data to sufficiently model both RV together. If we instead have distributions over factorized (= statistically independent) at hand and want to know how likely their coincident occurrence is, we can use sophisticated methods to estimate their joint p-value.

In the following, two approaches for combining p-values of independent tests are presented. A detailed examination of their advantages and shortcomings with a comparison to standard aggregation techniques is given later-on, in the section 4.3.1.

One possible method was introduced by Tippett[42] in 1931, who proposed to just take the minimum p-value of a set of independent tests. Let us assume we have a set of RV $\mathbf{X} = \{X_1, ..., X_n\}$ and observations $\mathbf{x} = \{x_1, ..., x_n\}$. Then

$$p(\mathbf{X} \geq \mathbf{x}) \approx min(\{p(X_i \geq x_i), \forall i \in [1, n]\}) \tag{2.38}$$

This is a very simple approach and a rather bad approximation. While the minimum of the set of p-values surely does not "make things worse" as multiplying p-values of statistically independent RV would do, it still returns a statistical significant value, if *any* of the aggregated p-values is less than α, although the other tested events might be very probable.

A more sophisticated method was proposed by Fisher[8], who suggested to construct a chi-squared distributed test metric out of a set of p-values. Fisher's method builds up onto the insight that the probability of rejecting

a null hypothesis falsely should always be the equal to the significance threshold α, whether we conduct a single test or combining multiple. If we assume that p-values of a RV itself are uniformly distributed, such that every p-value is equally like, then the negative double of the logarithm of the multiplication of all p-values is approximately χ^2-distributed with 2n degrees of freedom (2.39). So, for independent tests, this equation yields the combined probability that we falsely reject our original null hypothesis. If we assess it against a selected α, we can use this statistic as we would use p-values.

$$p(\mathbf{X} \geq \mathbf{x}) \approx \chi^2_{2n} \sim -2 \sum_{i=1}^{n} log(p(X_i \geq x_i)) \tag{2.39}$$

2.3. Machine Learning

Machine Learning (ML) is a sub-discipline of the study of Artificial Intelligence (AI) and describes the application of algorithms to instruct computers how to learn from data and subsequently act in a rational or in a human-like manner. The probably most wide-spread, more formal definition of ML was given by Tom Mitchell in 1997: "A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E"[25]. In practice, we want to instruct machines to *learn* a mapping from an input to an output, which can later be used to predict the output of unknown observations. The input can be any form of digital, structured and unstructured data like tabular data, text, images, sensor measurements, and many more. The form of the mapping is provided by the underlying machine learning algorithm and its concrete parameters are then learned from the given data.

The domain of machine learning algorithms can be grouped into three approaches: supervised approaches for classification and regression assign each input an associated, pre-specified output and the training data needs to contain labeled outputs for learning the model. Unsupervised algorithms attempts to structure unlabeled data by grouping (clustering) the data-points according to some distance measure, estimating their distribution, or reducing their features (random variables used for learning the model) by exploiting internal dependencies. If there is no data at all, reinforcement learning enables intelligent agents like robots to learn from actions taken in an interactable environment.

To instruct machines how to learn from data, (mainly) computer scientists have borrowed techniques from a range of different scientific domains, with the most successful coming from the statistical, logic and cognitive sciences. In the recent two decades, astonishing results of ML systems were demonstrated and successfully applied to a whole range of tasks like object detection in videos, speech recognition for voice assistants, deceptively real seeming natural language and image generation and a whole range of applications in bioinformatics, utilizing aforementioned methods to actually improve human health. Moreover, the size of today's data collections may be orders of magnitude larger than decades before, and structuring millions or billions of documents, each containing thousands of words, such that relevant documents are ordered according to their priority (for humans), presents a full-grown problem, where ML plays a fundamental role. But this also shows a shift in the interpretation of such algorithms: while computer scientists and others started applying and scaling methods from statistics, logic and so on to use them for their own purposes, one may wonder if the term *AI* just describes solutions that are not yet developed; the computer programs of tomorrow, so to speak.

2.3.1. Clustering

Clustering is an unsupervised ML technique that attempts to structure data by grouping similar datapoints. In the case of probabilistic clustering, learning is actually estimating this structure and its parameters of (usually) complex, multivariate probability distributions. This approach is originally called density estimation and many according ML approaches actually rely on research of the statistical sciences. The main difference that machine learning makes lies in the automatization and the scale and respectively computational challenges of the methods presented in section 2.2.1. To learn better estimations for complex data, we require flexible, complex models, that are able to identify small coherent clusters and do not need much human interaction. Such methods allow for splitting parts of the data in small, hierarchical units, that together resemble the whole distribution, but give a more fine-grained view on the structure of the data (in a top-down manner), or to interpret the inputs as small building blocks, whose connections and thus higher-order representations shall be learned within a certain depth of a model (in a bottom-up manner). Such algorithms for flexible, hierarchical model or parameter learning are researched in the field of "deep learning".

2.3.2. Anomaly Detection

Anomaly detection is the process of identifying data-points that are rare or unusual with respect to a statistical model. An anomaly is an observation disturbed by a lot of noise or other unobserved influences, so that it is suspicious if it comes from the same generating distribution or belongs to the same class of data points that are closer to the distribution's mean. Anomalies are outcomes of random variables (or a set of random variables) that are unlikely to be generated by the probability distribution of that random variable (or set).

The method of how to find outliers depends on the context and the corresponding definition of a suspect data point. One might either want to find particular outcomes that are unlikely to happen at all, like detecting discriminated numbers of an unfair dice, or find outcomes whose probability of occurring as values at least as extreme as the observed ones is low. But applying traditional anomaly detection techniques that aim to find rare datapoints can lead to problems in medical settings: First, some indicators may be rare at all, like a very high age or a patient from a sparsely populated area, because the population itself is distributed accordingly and such records do not have to be linked to disease outbreaks. Second, a low count of syndromes does not pose a problem at all; we only have to find suspicious high counts.

Within syndromic surveillance, we want to establish a data baseline of conditions over attribute combinations in a *normal* time period and assess if the generating distribution of new observations has changed, in specific if a disease outbreak has altered the typical distribution of a cluster in the data. This is also referred to as *activity monitoring*, *change detection*, or *event monitoring*. One possible technique for comparing observations against an estimated population is the use of aforementioned null hypothesis significance tests, for which a threshold α is defined before carrying out statistical tests and a predefined null-hypothesis will be rejected, if the p-value of an observation of a random variable of interest is less than α. However, if we want to just find suspicious clusters regarding the data baseline and do not attempt to draw causal conclusions from the data or hypothesis, we can also search for an optimal threshold after evaluating our test-set and optimize it with respect to the false alarm rate of such a model. This is especially important for models designed to detect disease outbreaks as early as possible and allows for calibration with respect to the local circumstances.

2.4. Sum-Product Networks

A Sum-Product Network (SPN) is a deep probabilistic model that represents the joint probability distribution over a set of random variables [37]. They are rooted directed acyclic graphs, so they have exactly one node as root and all other nodes are descendants of some order. In graph SPNs, each descendant can have multiple parents, which allows for reusing sub-SPNs and reducing redundancy. We will focus on tree SPNs, in which each descendant has exactly one parent, because their structure is easier to learn.

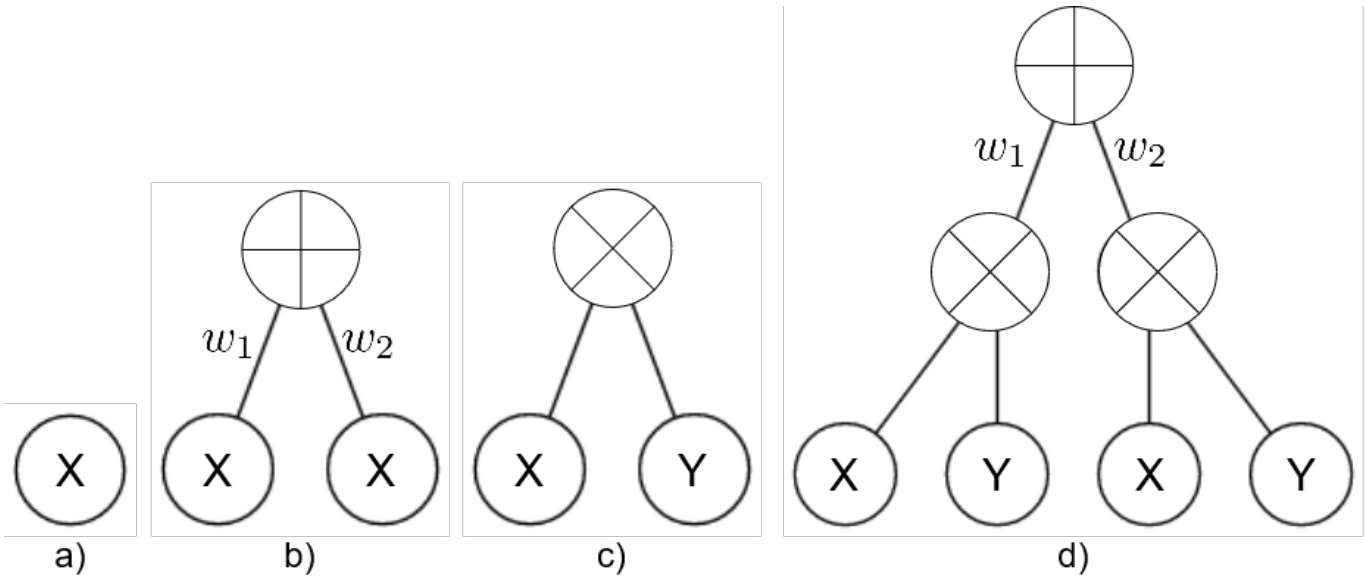

Figure 2.3.: Examples of valid Sum-Product Networks
a) a node representing a single probability distribution of a random variable X; $p(X = x)$
b) a smooth mixture of differently parametrized distributions over the same X with weights of the mixture components; $p(X = x) = w_1 p(X_1 = x) + w_2 p(X_2 = x)$
c) a decomposable factorization of two disjunct RV X and Y; $p(X = x, Y = y) = p(X = x)p(Y = y)$
d) a tractable deep probabilistic model; $p(X = x, Y = y) = w_1(p(X_1 = x)p(Y_1 = y)) + w_2(p(X_2 = x)p(Y_2 = y))$

The set of all modeled RVs of an SPN is called the *scope* and each node within an SPN represents a sub-SPN over a subset of the original scope. All nodes take on one of three different types: leaves, that are nodes that do not have descendants, sum nodes, and product nodes. Leaves usually encode a univariate probability distribution over a random variable (a scope of size 1). This probability distribution can be of any kind, be it categorical, Gaussian, Poisson or Negative Binomial. Sum nodes and product nodes need to have some additional properties so that the SPN is valid and can compute probabilities correctly: All children nodes of sum nodes need to have the same scope; and respective sum nodes represent mixture distributions over their scope. This property is called smoothness (or completeness in some literature). Additionally, the weights of the mixture components of a sum node are attached to the edges between the node and its children. These weights are normalized and can be interpreted as the probability of the respective mixture component, which is just the relative expected frequency of this cluster. All children nodes of product nodes need to have distinct scopes; and respective product nodes represent factorizations of statistically independent RV in their scope. This property is called decomposability (or consistency). A SPN that is smooth and decomposable is valid and computes the probabilities of evidence, marginal, and conditional queries exact and in linear time in

the size of the network (the number of nodes). This property is also called tractability, which means that the expression for computing the probability for one of those queries is in closed form, like a finite sum. Hence, we can compute the exact probabilities and do not need to rely on numerical methods for approximate values. Tractability also ensures the linear time inference of SPNs, which is a major advantage over the class of probabilistic graphical models (PGM) like Bayesian Networks, whose inference procedures are either exponential in the size of the PGM or only approximate. But to be computationally efficient at all, it still needs to be ensured that the size of the learnt SPN is not exponential in nodes with respect to the number of modeled random variables.

Some exemplary SPNs are shown in figure 2.3. Any probability distribution over one RV is a valid Sum-Product Network, as well as mixtures distributions over a set of RV and factorizations of statistically independent RV. Considering the structural limitations, sums and products can be combined in an alternating manner to build deep probabilistic models, either to design networks by hand or with structural learning algorithms, which will be introduced in a moment.

2.4.1. Structure Learning

The structure of SPNs can be learned from data. For this purpose, Gens and Domingos proposed LearnSPN[9], an algorithm scheme that recursively tries to cluster and split a data-set and follows the idea that we need to split the random variables in approximately independent sets and find context-sensitive coherent clusters in the set of respective instances. Context-sensitive means that an instance can be split according to the sets of independent RV it is part of and the slices of the instance do not have to be assigned to similarly structured clusters.

Algorithmus 1 : LearnSPN(T, V)

Input : set of instances T and set of random variables V
Output : an SPN representing a distribution over V learned from T
if $|V| = 1$ **then**
  return univariate distribution estimated from the variable's values in T
end
else
  try to split V into approximately independent subsets V_j
  if $j > 1$ **then**
    return $\prod_j$ LearnSPN(T, V_j)
  end
  else
    cluster T into subsets of similar instances T_i
    return $\sum_i \frac{|T_i|}{|T|} \cdot$ LearnSPN(T_i, V)
  end
end

LearnSPN is displayed in algorithm 1 and works as follows: First, if the set of random variables V in the current step only consists of one element, a univariate distribution over that RV with its associated, current instances T is constructed directly. If the set V contains more than one element, the algorithm attempts to split it by carrying out tests of statistical independence. If the test was successful and able to group the set

into approximately independent sets V_j, the algorithm will be executed recursively on each new subset. If the algorithm cannot assert statistical independence among the current set of RV, then it clusters the current instances T_i and LearnSPN is called with each cluster.

Algorithmus 2 : LearnPSPN(T, V, m, r)

Input : set of instances T, set of RV V, minimum number of instances for splitting m, and threshold for strength of dependency r

Output : a tree PoissonSPN S

if $|V| = 1$ **then**
  return PoissonNode$(\lambda \leftarrow \sum_i \frac{T_i}{|T|})$
end
if $|T| \leq m$ **then**
  for $V_j \in V$ **do**
    $T_j \leftarrow T$ of V_j
    $\lambda_j \leftarrow \sum_i \frac{T_{ji}}{|T_j|}$
  end
  return ProductNode(PoissonNode$(\lambda_1), ..., $PoissonNode$(\lambda_j))$
end
try to split V into approximately independent subsets V_j with threshold r (Poisson Instability Test)
if $j > 1$ **then**
  for $k \in \{1, ..., j\}$ **do**
    $S_k \leftarrow$ LearnPSPN(T, V_k, m, r)
  end
  return ProductNode(S_1, ..., S_j)
end
else
  cluster T into subsets of similar instances T_i
  for $l \in \{1, ..., i\}$ **do**
    $S_l \leftarrow$ LearnPSPN(T_l, V, m, r)
  end
  return SumNode$(\frac{|T_1|}{|T|} S_1, ..., \frac{|T_i|}{|T|} S_i)$
end

The statistical independence test and the clustering algorithm can be easily exchanged, which allows the learning procedure of SPNs to be adapted to different contexts. Molina et al. developed two approaches, the first for learning a set of Poisson distributed random variables, called PoissonSPN [26], and the second for learning from data consisting of any kind of distribution, be it categorical, discrete or continuous, called MixedSPN [27]. In LearnSPN, we can assume that splitting V induces a product node and clustering T a sum node, but unfortunately, the algorithm has the shortcoming that it does not try to find mixture components for a single RV. The algorithms of Molina et al. complement these steps and LearnPSPN can be seen in algorithm 2. It extends the original LearnSPN by two parameters m and r: m is the number of minimum instances needed for constructing a leaf node. Usually we set m by multiplying the total number of instances with a ratio, say 0.1. This allows to control for the size of the learned SPN and helps to prevent overfitting. r is a parameter for the statistical independence test and controls how strong the independence value has to be. The strict definition of statistical independence rarely holds in large datasets, as the values may differ alone from noise and because we only have a subset of observations instead of the true distribution. So r allows for context-sensitive control of the behavior of the splitting test.

LearnMSPN, the algorithm for MixedSPNs, operates in a similar way, but instead of learning PoissonNodes as leafs, the appropriate type of node with respect to the probability measure of the RV is selected. As MSPNs allow to combine different types of random variables, the splitting test is replaced with the randomized dependency coefficient (RDC) test [23], which can be used to assess approximate independence between differently distributed RV. For the clustering step, various standard cluster algorithms can be used. One option is to cluster the current instances with the k-means algorithm. When constructing a leaf node, the data points of those clusters can be used to estimate the respective parameters of each leaf and its associated distribution with MLE. More sophisticated splitting and clustering methods can be found in the publications of the presented learning algorithms.

2.4.2. Inference

Smooth and decomposable Sum-Product Networks guarantee exact and fast inference for computing the probabilities of evidence, marginal and conditional queries. For those three operations, the time needed grows linearly with the number of edges in the network. In tree-SPNs, this is equal to the number of nodes of the SPN minus 1 (because the root has no edge to a parent). To ensure that the computational complexity of those queries is in P, so that they are efficiently computable for very large models, the number of edges may not be exponential in the size of modeled random variables.

Probability of (Full) Evidence

Evidence or full evidence computes the probability for a complete instantiation (we observed a value for each RV). If our SPN models the distribution of two RV X, Y, evidence means computing $p(X = x, Y = y)$. This enables us to check how likely the observed state is. However, when the SPN represents many RVs, the curse of dimensionality has to be considered, which leads to low probabilities to each fully observed state, because the system as a whole exhibits lots of possible combinations between the RVs and therefore lots of states in total. To evaluate the full evidence probability with an SPN, the observed values have to be set in the leaf nodes to compute the respective probabilities $p(X_1 = x), ..., p(X_n = x)$ and respectively for Y. These probabilities are propagated upwards to compute the probability of combined RVs. At product nodes, the product of the probabilities of all children is computed. If X, Y are children of a product node, then $p(X = x, Y = y) = p(X = x)p(Y = y)$. This is equal to the joint probability of statistical independent RV, defined in equation 2.2.8. At sum nodes, the probability of each child is multiplied with the probability for the respective cluster, which is the corresponding weight w_i. Then, all weighted probabilities are summed up: if X is split into two clusters, then $p(X = x) = w_1 p(X_1 = x) + w_2(X_2 = x)$. This is equal to the probability of the mixture model represented by the sum node. The root node then computes the correct probability of the evidence query given the joint probability distribution modeled by the SPN.

Marginal Probability

Marginal queries compute the probability of the occurrence of a value of a subset of RVs modeled by a joint distribution. When an SPN encodes the distribution of X and Y, we might be interested in the probability that only a specific value of x occurs, $p_{XY}(X = x)$. In this case, we need to sum all occurrences of x with all other possible occurrences of the other RVs, in the case of X, Y: $p(X = x) = \int_{y \in Y} p(X = x, Y = y)$ (for continuous distributions). To compute the marginal probability in SPNs, the probabilities of the distributions of leaf nodes

that do not belong to the set of marginalized variables need to be adjusted. These leafs are set to return a value of 1. Then, the probability of the marginal variables is computed in the same manner as for evidence by just propagating the probabilities from the leafs upward and combining them at sum and product nodes.

Conditional Probability

Conditional queries can be executed in two ways. First, the conditional probability is defined as $p(Y = y|X = x) = \frac{p(Y=y,X=x)}{p(X=x)}$. To obtain this conditional probability, we simply have to compute the probabilities of the joint distribution and the respective marginal distribution, and divide the first by the second. This requires two passes in the SPN and is still bounded linearly with respect to the network size. The second option is to condition the SPN and then compute the evidence or marginal in this conditioned network. To do this, a set of values of RVs that are to be conditioned on is passed to the SPN. The corresponding leafs are eliminated from the SPN and the weights at sum nodes that describe the distribution of the clusters are adjusted accordingly. Then, the same procedure as for computing evidence and marginals is applied.

3. Related Work

The research field of syndromic surveillance was founded in the late 1990's and according research activities increased in the 2000's. There exists a large body of research regarding specific health surveillance systems, which attempt to monitor particular diseases or indicators like drug purchases or trends in search engines [16, 11]. Such systems perform well for their given task, but have the disadvantage that the attributes of interest need to be specified beforehand, hence can only find according outbreaks and cannot generalize to other diseases or outbreaks with unknown characteristics. Another common approach is to monitor univariate time-series of every attribute present in health-data with fully factorized models. This allows for a more comprehensive view than the previous systems and is suitable for the surveillance of general public health, but might miss subtle disease outbreaks that only appear to be rare with respect to a combination of monitored attributes, but not for each single component [2, 38].

The first unspecific syndromic surveillance algorithm that considers all kinds of combinations of health-related attributes is known as WSARE ("What's strange about recent events") and was proposed by Wong et al. in 2002 [46]. Since then, they developed multiple versions, which differ in the method for selecting the data-baseline that is used to evaluate new observations against it, but operate identically on this data. For each syndrome of size 1, WSARE obtains the count of instances that contain the syndrome and the count of those that do not. These values are computed for the recent data and the historic data and compared by means of a χ^2-test of independence or Fisher's exact test for small values. The syndrome with the lowest significant p-value is selected and used to search for the best-scoring syndrome of size 2 in the same manner. To compensate for multiple hypotheses testing, randomization tests are used to compute a valid p-value for the day of evaluation. If the compensated p-value is significant, an alarm is raised.
For the data-baseline, WSARE 2.0 ignores environmental information and uses the syndrome counts of the admissions from a pre-specified set of days prior to the current date. WSARE 2.5 uses all available data that matches the environmental conditions of the current day. WSARE 3.0 learns a joint probability distribution of the emergency room data with a Bayesian Network, conditions on the environmental attributes and uses 10000 sampled records as data-baseline [45]. WSARE was deployed successfully in several real-world scenarios, but is prone to a high false alarm rate [10, 19, 2].

EigenEvent, proposed by Fanaee-T et al. in 2014 [6], tries to detect statistically significant differences in the eigenvalues and eigenvectors of spatiotemporal syndrome counts. The data-baseline is interpreted as a tensor of the dimensions *Counts x Region x Time*, the recent data as a matrix *Counts x Region*. Singular value decomposition (SVD) and higher-order SVD are used to compute the respective eigenvalues and -vectors. If the ratio of eigenvalues or the length of eigenvectors between the recent and historic data appear to be unprobable, an alarm is raised. Compared to WSARE, EigenEvent reduces the false alarm rate, but cannot report the suspicious syndrome causing the alarm and is inferior with respect to the outbreak detection delay.

4. Count Modeling with Sum-Product Networks

This chapter presents the detailed approach of how to prepare health data from emergency rooms and build according Sum-Product Networks. Different evaluation techniques for disease outbreak detection are presented, and their advantages and shortcomings are discussed briefly. Using hypotheses tests and substituting the original procedures of sum and product nodes with viable methods for combining p-values of multiple tests in Sum-Product Networks is a novel technique that was developed first within this thesis. By conducting hypotheses tests on all syndromes up to a defined size, we can assess if any observed syndrome count is anomalous with respect to the historic patient records and if the suspicious cluster should lead to giving an alarm, on which further investigation of a possible disease outbreak is handed off to medical staff. The presented system can either aid health institutions to find anomalies in combinations of syndromes that would not have been evaluated otherwise, or to assure that no or most of the evaluated syndrome counts do not appear to be suspect to a disease outbreak.

4.1. Data

Obtaining data for the evaluation of syndromic surveillance systems is a difficult task due to two circumstances: First, we need training data that approximately describes a history of "usual" medical records, in the number of cases per time-slot and in the distribution of the reported attributes. This data should be free from any major outbreaks that significantly alter the structure of what can be assumed to be a "status quo". Secondly, to reliable evaluate the performance of such models, the test data needs to contain an identifiable disease outbreak and a label that provides its onset. Theoretically, such data could be collected and prepared with the help of experts like epidemiologists and physicians, but this was not pursued within this thesis.

Other requirements for general detection systems lie in the temporal and spatial coverage of the data. The datasets should at least contain one year for the training phase and, to assess the model's quality, optimally another year for the test phase. In the spatial context, the data should contain entries from the regions that shall be monitored[1]. Health data can strongly depend on environmental attributes like the season or weather and follows seasonal (or temporal) trends, in this context also termed cyclic drift[44]. To decrease bias towards a specific set of environmental attributes, ideally the test data does not only persists of one set, but multiple with differently timed disease outbreaks. While Sum-Product Networks are able to deal with missing data entries, these should be kept to a minimum for reliable performance.
While it is reasonable that the set of patient attributes should contain categories like the reported symptom, the age, the gender and the patient's home district, the concrete realization can vary and nonspecific syndromic surveillance does not necessarily need the complete view of patients, as we monitor for irregularities in any personal feature.

[1] Transferring learned models to other regions might be possible, but would require intense validation that the assumptions hold and similar distributions of cases are present in the original and target regions.

Finding datasets that fulfill above-mentioned criteria is challenging and for the proof of concept we rely on a collection of synthetic datasets from a previous syndromic surveillance approach developed by Wong et al. [45].

4.1.1. Synthetic Data

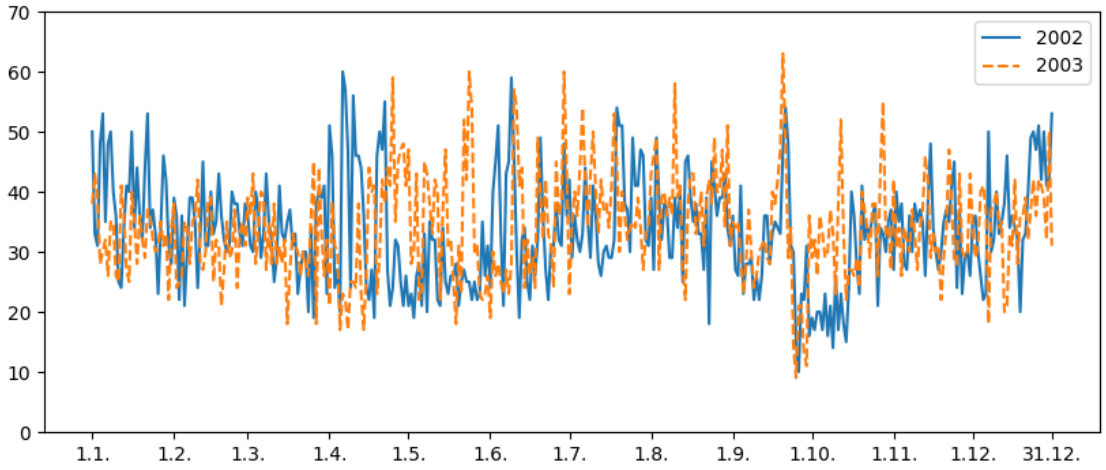

Figure 4.1.: Total admissions in a simulated emergency room dataset
The count of all admissions to the emergency room fluctuates throughout the year. The high peaks in the summer time are due to more reported accidents, while there are more cases that stem from viral and bacterial diseases like the flu in the winter months. To make informed decisions about the commonness of reported symptoms with respect to the environmental setting of the day of the evaluation, the according environmental attributes need be accounted for.

Wong and his colleagues created datasets that simulate the admissions in an emergency room of a hospital in a small-scale city. The purpose of the data-set originally was to detect anthrax outbreaks, following the anthrax attacks in 2001, when 22 people were infected by spore-containing letters, of which 5 died [4]. Although we do not want to detect specific anthrax outbreaks, but disease outbreaks in general, the datasets are still viable, as the anthrax outbreaks are designed in a subtle way and anthrax in general leads to similar symptoms as the flu. In specific, reported symptoms often include respiratory problems, fever and coughing. With the current epidemic crisis in mind, these are also the symptoms that we expect from a range of corona-vira, like Covid-19, SARS, MERS and others. Moreover, the datasets fulfill each of the criteria mentioned above and are, to our best knowledge, the only collection that sufficiently allows to assess the performance of syndromic surveillance algorithms.

To create the 100 datasets, Wong et al. used Bayesian Networks (BN). Bayesian Networks are probabilistic graphical models (PGM) that provide a clear factorization of a joint distribution and model (conditional) statistical independence between the involved random variables. They allow to describe causal relationships between entities and therefore are an appealing choice to create data that we believe to exhibit such dependencies. First, they built a BN that simulates the epidemiological status of a city. This contains the fixed attributes date, day of week, and season and the sampled attributes weather, flu-level, food condition and anthrax concentration. Using these environmental attributes, they used a second BN to simulate the patient admissions

and respective attributes like age, gender, and home district, that influence a set of latent variables[2] like the diseases a person is infected with and the actual resulting symptom, which leads to the reported symptom, action and drug purchase covered by the data. Finally, the patient records with the attributes described in the next subsection are fed into the final datasets, if the patients have chosen any other action than "none".

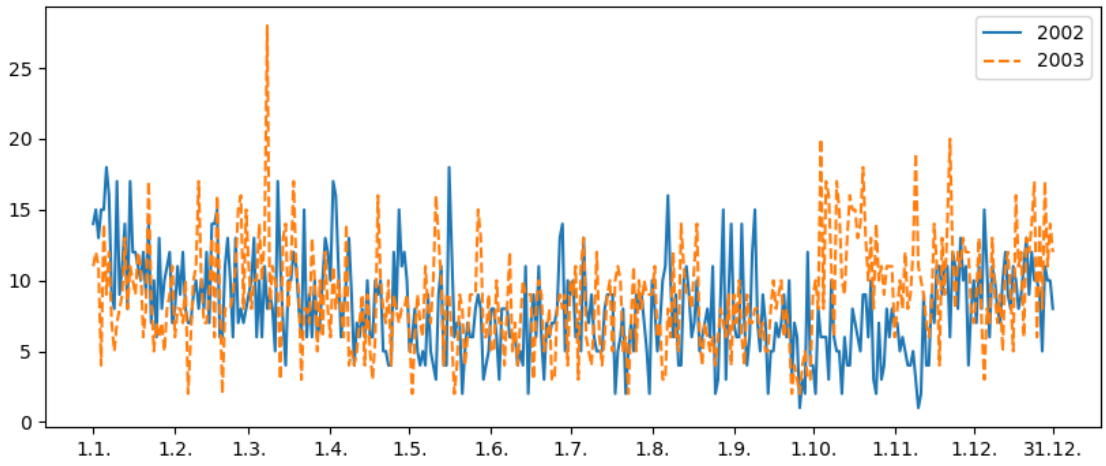

Figure 4.2.: Counts of respiratory symptoms for training and test year
Comparison of the distribution of counts of reported respiratory symptoms in the first data-set. The anthrax onset happens at 03.10.2003 and can be seen at the undulating orange cluster between October and November. The anomaly at 08.03.2003 is likely due to chance, which the system should also detect and hand off further investigation to medical staff.

For the test set, the same procedure was followed, but an anthrax outbreak of 14 days at a randomly chosen day of the second year was added to the simulated data of one of the 9 simulated districts, such that count of total admissions increases and respective cases are more likely. An illustration of how such an outbreak might look like can be seen in the visualization of the reported counts of respiratory symptoms for the training year (2002) and the test year (2003) in figure 4.2. The anthrax disease outbreak occurs at the 03. October 2003, which can be clearly identified as the group of high values of the orange, dotted line shortly after the label *1.10.*. With regard to the counts of the previous year, the difference can be easily spotted and a proper syndromic surveillance system should have no trouble detecting this cluster. Contrary, the suspicious high count of 28 reported respiratory symptoms at 03. March 2003 is an artifact from the generation process. The system should also throw an alarm when it encounters such phenomena and the alarmed hospital staff can then investigate and eventually confirm or reject the assumed infectious cluster.

Structure

Each of the 100 datasets contains 12 attributes and 23,250 to 25,889 instances representing admissions from an emergency room over a time of 2 years. The second year contains a simulated anthrax outbreak. The

[2]The variables in the Bayesian Networks they used are explicitly stated. But the datasets we have at hand only contain a few attributes as output, and thus are latent from out point of view. Theoretically, these latent variables could be estimated, but it would be tremendously challenging to reverse engineer the parameters of those fairly complex networks.

day of the onset is randomly chosen over the whole year and the outbreak always lasts 14 days (except if it occurs after the 17. December). Before we examine how the patient's records during outbreaks differ from the data-baseline, we will take a look at the structure of the unprocessed datasets.

The 12 attributes can be grouped as 6 personal attributes and 6 environmental attributes:

- Personal attributes:
  - **age**: 3 categories: {"young", "working", "senior"}
  - **sex**: 2 categories: {F:"female", M:"male"}
  - **XY**: the district of the patient's residence. 9 categories: {C:"central", N:"north", S:"south", E:"east", W:"west", NE:"north-east", NW:"north-west", SE:"south-east", SW:"south-west"}
  - **symptom**: the reported, possibly provisional symptom of the patient, also called prodrome. 4 categories: {"none", "nausea", "rash", "respiratory"}
  - **action**: the action the patient performed: being absent from work or school, visiting a doctor or purchasing medicine. 3 categories: {"absent", "evisit", "purchase"}
  - **drug**: the drug/medicine the patient received, if any. 4 categories: {"none", "aspirin", "nyquil", "vomit-b-gone"}
- Environmental attributes:
  - **season**: 4 categories: {"spring", "summer", "fall", "winter"}
  - **weather**: 2 categories: {"cold", "hot"}
  - **flu**: the reported level of influenza activity in the city. Usually increases in mid-winter and results in more admissions with respiratory symptoms. 4 categories: {"none", "low", "high", "decline"}
  - **day_of_week**: 3 categories: {"weekday", "sat", "sun"}
  - **date**: date in the format "MON-dd-yyyy". 365 entries for each year
  - **daynum**: an integer counting the number of days since 01.01.1800

personal attributes						environmental attributes				
age	sex	XY	symptom	action	drug	season	weather	flu	dow	date
child	F	NW	nausea	absent	none	winter	cold	high	week	01JAN2002
senior	M	SE	none	purchase	niquil	spring	cold	none	week	20MAR2002
working	M	SW	none	purchase	aspirin	spring	cold	none	sun	02JUN2002
senior	F	NE	rash	evisit	none	summer	hot	none	week	03SEP2002
child	F	C	respiratory	absent	aspirin	fall	cold	high	sat	14DEC2002

Table 4.1.: Exemplary entries of patient records in the simulated emergency room data
The personal and environmental attributed present in the synthetic datasets from Wong et al.. In this unprocessed form, each column only contains categorical values. The column "daynum" was omitted, as it does not contain relevant information for the data-baseline.

Relation to Real Emergency Room Data

While the synthetic datasets from Wong et al. fulfill the requirements for syndromic surveillance data above, they are not real emergency room data. Real emergency room data from a german hospital "Sana-Klinikum" in Offenbach was obtained from the Robert-Koch-Institut. The "Sana-data" comprise admissions to the emergency room of the same-named hospital from 01. October 2016 to 13. December 2019, and new data is collected regularly. It also contains the gender, discretized age, partial zip codes and reported symptoms, but is far more comprehensive than the simulated datasets and contains lots of additional professional information ascertained by medical staff.

4.1.2. Preprocessing

As syndromic surveillance aims to find anomalies in the patterns of possible diseases, the data introduced above needs to be processed accordingly. Before the concrete procedure is outlined, we'll show how to construct syndromes from the medical information.
First, recall that a syndrome is a possible disease pattern and, in this context, the same as probabilistic events, so they are subsets of the power-set of all possible outcomes of distinct random variables - with a chosen maximum cardinality of $c \in \mathbb{N}_+$. When we want to model syndromes with the size of 1, each category of the personal attributes represents one syndrome. With syndromes of size 2, we would combine all instances of every pair of RVs, with size 3 of triples of RVs, and so on. But choosing higher maximum sizes also leads to massively (combinatorial) more features and thus increases the complexity of the data.
Let's look at the Wong datasets: each contains 6 personal attributes with a total of 25 categories. With only syndromes of size 1, we have exactly 25 as syndromes. When we add the syndromes of size 2, which accumulate to 245 unique ones, we already have 270 in total. But as we only have data available for one year, the training set would have a size of 365 rows times 270 columns. In other words, we only have 365 observations for 270 input dimensions. Typically, this may lead to weaker performing models, as the space of full observations is very sparse (in the ML community also known as *curse of dimensionality*). This can lead to issues when evaluating new counts against this data-baseline, simply because for a few syndromes there may be no observations with respect to the environment. The problem can only get worse when we add syndromes of higher size, as long as the number of instances does not significantly increase. Instead of building bloated, sparse models on insufficient data sources, we rather model the count of syndromes of size 1 and then try to evaluate higher-order syndromes with sophisticated testing methods in the *simple* model.

Within such simple models, we do not constraint the input to selected syndromes and, except for transforming patient records to syndrome counts, there is no real feature-engineering involved. Instead, we want to learn a distribution over and evaluate as many combinations of possible disease patterns as practicable; we want to build an unspecific syndromic surveillance model.

In the following, the scheme of how to concretely preprocess emergency room datasets that follow similar structures as the presented ones is provided. The categories of the personal attributes need to be aggregated to establish a data-baseline over the outbreak-free distribution of the syndromes. While we only consider daily counts in this thesis, the time-slot can be chosen arbitrarily, as long as each slot covers a fixed time-interval. The steps of transforming raw patient records to syndrome counts are:

1. Split each data-set into personal and environmental attributes, if any environmental attributes are present. If not, the data remains unaltered and some environmental information can be added to the data later.

2. Group the set of personal attributes by day.
3. For each syndrome that should be modelled, count the cases with this syndrome for the given day. For each day, we now have an entry that contains the syndrome counts for that day; so for one year of patient records, we have 365 (or 366) instances for the actual model.
4. Enrich the new syndromic instances with the (previously removed) environmental attributes.
    - The environmental variables usually need to be discretized, mapping their alphanumeric values to strict, unique numeric values, which is required by many statistical frameworks to construct probability distributions over those RVs.
    - Environmental features that represent unique events like the complete dates of admissions should be simplified to make sure that we can choose an appropriate data-baseline in the learning (1 value per instance) and inference steps (e.g. 29. February could make trouble). In particular, the dates in the synthetic data were truncated, such that only the month of the admission is used.
5. Split the datasets into respective training and test sets, if any test-data is available (in case of presented synthetic data into first year and second year).

If we want to test new observations, the same procedure is applied to the set of recent daily admissions. With the newly constructed syndrome data, we can finally learn a Sum-Product Network, which will be discussed in the next section, before we eventually get to the evaluation techniques and show how we actually can compare new counts with the learned distribution of our just constructed syndromes.

	syndromes									environment	
day	#F	#M	#senior	#working	#N	...	#evisit	#respiratory	#aspirin	dow	season
1	13	18	10	14	5	...	16	11	5	0	3
2	19	20	8	20	6	...	12	12	9	0	3
3	16	12	8	12	6	...	10	11	5	0	3
4	11	12	9	11	6	...	2	4	5	1	3
5	24	14	17	14	7	...	14	14	4	2	3

Table 4.2.: Exemplary syndrome counts of a simulated emergency room
After executing the steps described in the pre-processing section, we obtain a table that looks similar to this. The cases of each value of the personal RV have been aggregated. Two environmental variables, numerically discretized, are also shown, where "dow" is short for day of week. As the season and other, not displayed, environmental values are the same for each entry, the counts of reported cases vary alone due to chance, as we would expect it from real world data.

4.2. Modeling

Before we can learn a probabilistic model, we have to think about some modeling aspects: What distributions do we want to use for each random variable and do our assumptions hold, so can we justify our decisions? Is our chosen model able to do what we expect from it? After answering these questions, we can finally build SPNs.

4.2.1. Interpreting the Data

As seen in section 4.1.1, the preprocessed data-set contains two types of attributes: personal attributes, these are the syndrome counts, and environmental attributes, like the season and the flu-level on each day. Within both classes, all attributes are of the same kind and can be modeled with the same type of probability distributions. The first are all count data, produced by summing up the occurrences of all possible events per day, and the second are discrete, unordered values.

Let us model the environmental attributes first, as there is little room for interpretation: Each of these consists of disjunct events. They all are perfect depictions of categorical random variables, which assume just discrete and unordered values. We cannot summarize statistics from the environmental attributes, as metrics like mean or variance cannot be computed (expect the mode, which is simply the most frequent event). The PMF maps each event to its probability and only supports values that are explicitly defined within the distribution. As there is no ordering, there also does not exist an according CDF.
However, for the personal attributes, some options for the underlying probability distributions are available, and the choice will have an impact on the interpolation and extrapolation quality of the respective SPNs. Therefore, we want to assess if the assumptions made by the Poisson, Gauss and Negative Binomial distributions also hold for the syndrome counts and examine the approximation quality empirically at the example of respiratory syndrome counts of one synthetic data-set.

Poisson Distribution

The first option is the Poisson distribution. Roughly spoken, the Poisson distribution models counts, and therefore might be applicable for our data. According to "Modeling Count Data" from Joseph Hilbe [17], the Poisson distribution assumes the following conditions, which the modeled data also has to fulfill:

1. The mean is understood as a rate parameter and is the expected number of times that an event occurs in a fixed time interval.
2. The observed values are nonnegative integers and the distribution allows for values greater than 0.
3. The observations are independent of each other.
4. No observed count is substantially more or less than the expectation of the observations. The higher μ, the lower the chance of seeing a count of 0.
5. The mean and variance, which are both described by λ, are approximately the same. Higher counts lead to greater variability.
6. The observed and predicted variances are approximately the same, so we do not encounter over- or underdispersion.

We now assess these assumptions:

1. The syndrome counts are count data and do represent the expected value per day.
2. We count the occurrences of events of each attribute, so this value cannot be negative. A count of 0 is possible.

3. The observations may not be independent, as diseases spread by contact with infected persons, which requires spatial and temporal dependence. However, syndromes are not equal to diseases and usually more general. They most often do not concentrate in one place, but are distributed within a region.
4. It is very unlikely that syndromes that occur often suddenly do not occur. Even if, this would not pose a problem for the task of syndromic surveillance. By previous criteria, the data for the historic baseline should represent outbreak-free records and not contain suspiciously high counts. From an engineering point of view, we can decide that this criterion is less important.
5. The average mean of total admissions over the first year of all 100 simulated datasets is 33.40, the variance 79.51, so the data exhibits overdispersion. Therefore, this criterion for the Poisson distribution is not fulfilled.
6. The average mean of total admissions in the second year of all 100 simulated datasets is 33.78, the variance 89.16. The mean is very close to the value of the training year and the variance somewhat higher, so there is some difference between the statistics of the training and test years.

Moreover, if we later observe that one of our assumptions truly does not hold, e.g. a very dense, intra-dependent cluster of reported syndromes is observed, we can rightfully reject that the suspicious data was generated by our learned Poisson distribution. Although independence among observations might not hold, a mixture of Poisson distributions might still be a good approximation for our syndrome counts.

Normal Distribution

Another viable option for modelling this count data is the Normal distribution. A good indication for this is that we can model our counts with Poisson distributions, that slowly takes on the form of a Normal distribution, if the mean lies around 30 or higher, which is the case for our observations of syndromes of size 1. But let's rather assess the theoretical foundation:
We can interpret each syndrome, like female and male, as a Bernoulli distributed random variable. This means that a patient has a chance p_f of being female or not, and also has a chance p_m of being male or not. The evaluation of syndromes belonging to the same attribute does not make trouble, as long as we do not evaluate the probability of a patient being male and female together, which would not make sense anyway. We now only consider one of those Bernoulli distributed events and assign a value of 1 if it is present and 0 if it is not. The chance of being female is the same for every patient, and the gender of one person does not influence the gender of another, so being female is an independent and identically distributed random variable. When we now sum up all observations of these i.i.d. distributed observations, we fulfill the criteria of the Central Limit Theorem and our sums can be described by a Normal distribution. This distribution computes the probability that we observe a particular count, given that the probability of the underlying Bernoulli distribution and the number of distributions (one for each person in the observed population) remain unaltered.

However, the chance that a syndrome is admitted in an emergency room depends on environmental variables, and the resulting multi-modal distribution might not be well described by a single Normal distribution. Combining differently parametrized distributions within a mixture model allows us to approximate all clusters well. An illustration of the advantage of environmental-based clustering can be seen in figure 4.3. Hence, we can justify the usage of Normal distributions.

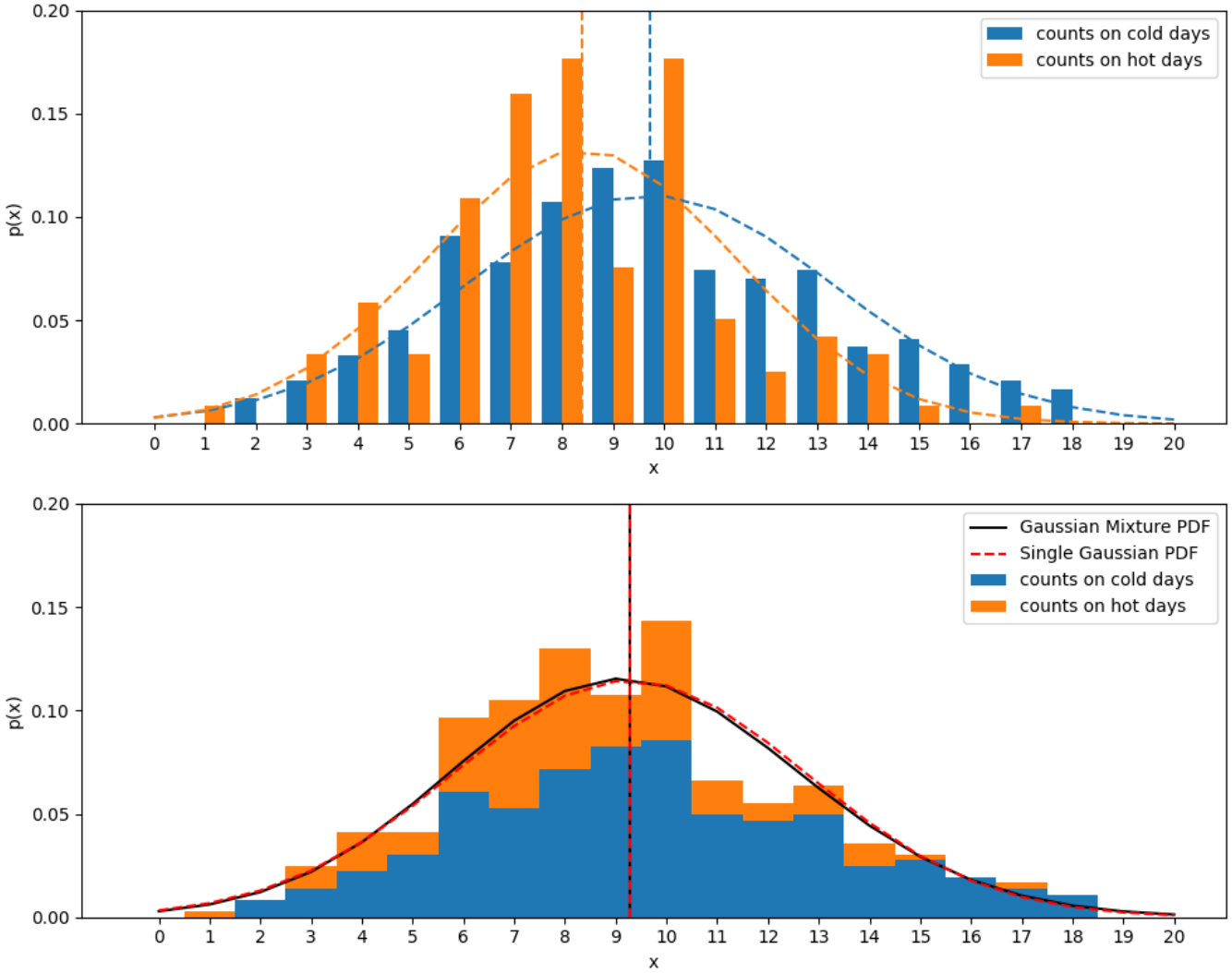

Figure 4.3.: Histograms and Gaussian PDFs of count data clustered by the weather
Top: Counts of respiratory symptoms are split into two clusters of the respective weather for each day. The frequency histograms and the estimated Gaussian PDFs for both clusters are plotted. The vertical dashed lines are the respective means. We can see that the two probability distributions are differently parametrized and lead to better approximation of both clusters than a single distribution over all datapoints. E.g. the probability for observing counts greater than 10 differs between the two clusters and should be considered when evaluating new instances against this model.
Bottom: A frequency histogram of the stacked clusters and its according Gaussian PDF estimated from all data points (red dashed line). The black solid line represents the mixture model PDF of the weather-based clusters, for which the cluster PDFs are multiplied with the relative frequency of data points belonging to that cluster, and those weighted distributions are summed up afterwards. The single PDF and the mixture PDF are very similar and their means almost equal, but the mixture model allows to consider only the appropriate cluster with respect to the weather on a new day of observations (as in the top figure).

Negative Binomial Distribution

As the Negative Binomial and the Poisson distribution both model the occurrence of (originally Bernoulli distributed) counts, they have very similar requirements. However, the Negative Binomial distribution can account for overdispersion in the data, meaning that in contrast to the Poisson distribution, the variance may be larger than the mean. If the variance is equal to the mean, the both distributions are the same.

As the assumptions for the Poisson distribution were already assessed in this section, we'll only check if samples from the data do not exhibit underdispersion. The empirical values for the mean and variances of the training and test data are the same as in section 4.2.1: $\mu_{\text{train}} = 33.40, \sigma^2_{\text{train}} = 79.51$ and $\mu_{\text{test}} = 33.78, \sigma^2_{\text{test}} = 89.16$. Therefore, the requirements for applying the Negative Binomial distribution to model our count data are fulfilled.
Another criterion for using the Negative Binomial distribution is that it should not be used to model data with relatively few counts. The goodness of fit regarding modelling syndrome counts with the Negative Binomial distribution can be seen by the comparison of the evaluation metric with the other two probability distributions in chapter 5.

4.2.2. Learning Sum-Product Networks

#symptoms	month
28	Jan
29	Feb
25	Mar
21	Apr
18	May
16	Jun
15	Jul
15	Aug
18	Sep
21	Oct
24	Nov
25	Dec

#symptoms	season
28	Winter
29	Winter
25	Spring
21	Spring
18	Spring
16	Summer
15	Summer
15	Summer
18	Fall
21	Fall
24	Fall
25	Winter

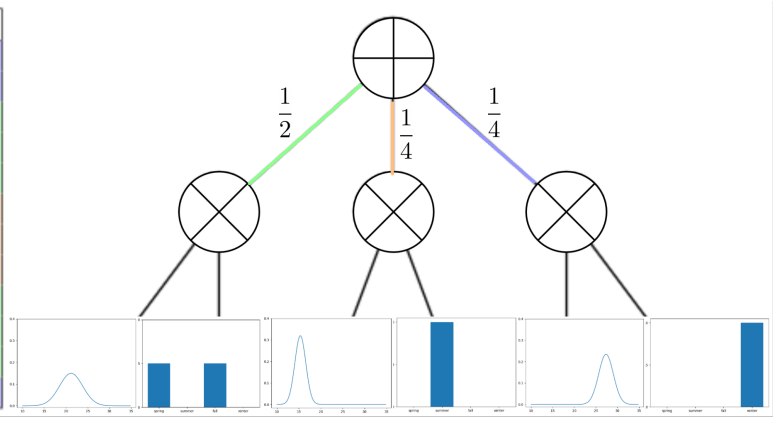

Figure 4.4.: Designing the structure of an SPN by hand on the basis of tabular data
Left: The original health data containing each month of a year and associated counts of a symptom. Mid: Preprocessed data. The single months were replaced with the meteorological seasons. Right: An SPN built by hand on the basis of the health data. The sum node splits the data into three clusters and the product node splits the two RVs in a context-sensitive manner. The left-side leafs are Gaussian distributions describing the counts and the right-side leafs are categorical distributions describing the seasons.

We have now assessed the assumptions for using those three probability distributions for our syndrome counts and finally can learn respective Sum-Product Networks from the data.
First, let's see how to design an SPN by hand using exemplary tabular data: We have some exemplary health data in the form of a table containing each month of the year and the respective counts of reported cases with respiratory symptoms in an emergency room for each month. The unknown distribution of the reported counts is multi-modal, meaning that it has multiple peaks, and a single Gauss distribution cannot model all counts sufficiently. We associate the modes of this distribution with the respective months and see that there are more reported counts in the winter months than in the summer months, and the fall and spring months have

similar numbers in between. As we only have one observation per month, which would lead to degenerate probability distributions, the months are replaced with their meteorological seasons. We now want to cluster and split this data to expressively represent the context-sensitive dependencies among it. We interpret the respiratory counts as a Gaussian distributed RV (because they are sums of i.i.d. observations) and the seasons as a categorical distributed RV.
Now, there are three clusters present in the data: one for winter with high counts, one for summer with low counts and one for spring and fall. We construct the SPN with a sum node as root and three links for each cluster and assign the frequencies of the clusters as weights: $\frac{1}{4}$ for summer and for winter and $\frac{1}{2}$ for the spring-fall cluster. As each cluster still contains multiple RV, we select product nodes as the three children of the sum nodes, which split the Gaussian and the categorical RVs. The children of the product nodes are these distributions with according parameters: the mean and variance of the Gaussian nodes can be estimated from the reported counts and the categorical nodes represent the probability of the season within the respective clusters. This SPN correctly encodes the joint probability distribution of the given data in a compact graph, which allows us to either model our knowledge of the data directly, like in this example, or interpret the structure and dependencies of automatically learned deep models.

Count SPNs

The method for the hand-designed example above operates roughly the same way as the presented structure learning algorithms in section 2.4.1. However, before learning SPNs over all data and incorporating the set of environmental attributes, I focused on strict count SPNs, meaning that the SPNs are learned on data only containing the syndrome counts based on the personal attributes, but no environmental information whatsoever. While I mentioned that the reported emergency room admissions are dependent of the environmental conditions of the day of admission, using only count data can have two benefits. First, having less attributes, the data is less complex and can lead to more robust models. Moreover, if we construct the data-baseline for a day of evaluation only from historic records that match the same values of the environmental variables, there is a chance that the present environmental conditions have been observed only rarely or not at all before, which could lead to problems in inference. Secondly, real health data usually does not include any environmental information, and enriching the data with such aspects may be straight-forward for e.g. the season or month, but not for more complex variables like the weather or the flu-level, which can vary between segments of the observed region and also may need pre-processing in the form of discretization.

Let us look at how the process of learning the structure of syndrome counts works in detail: The data is in tabular form and consists of instances that contain the daily aggregated syndrome counts similar to those in table 4.2. We have split the data into a training and a test set, and only use the training set for learning the SPNs. Each syndrome is interpreted as a random variable with the same probability measure, either Gaussian, Poisson or Negative Binomial distribution. Different learning algorithms are now available: With Gaussian nodes, we can use the traditional LearnSPN or LearnMSPN, and with Poisson nodes, LearnMSPN or LearnPSPN is applicable. The Negative Binomial distribution cannot be estimated via MLE in closed form, so we instead learn SPNs with Gaussian Nodes and derive the Negative Binomial parameters from the Gaussian parameters in the evaluation step. To make the different node types comparable, we want to use the same algorithm for each type, and therefore choose LearnMSPN. It may be assumed that larger SPNs approximate the underlying distribution of the training data better, therefore we learn SPNs of different sizes by learning unique SPNs for a set of pre-defined minimal instances ratios $m \in (0, 1]$. If this ratio is set to 1, the algorithms learn one leaf per random variable which parametrizes all available data of that RV. So it only consists of one product node having as many children as there are columns in the tabular dataset. If $0 < m < 1$, the

algorithms can learn hierarchical networks that may approximate the joint probability distribution better than the fully factorized model represented by the flat SPN.

Count SPNs with Environmental Variables

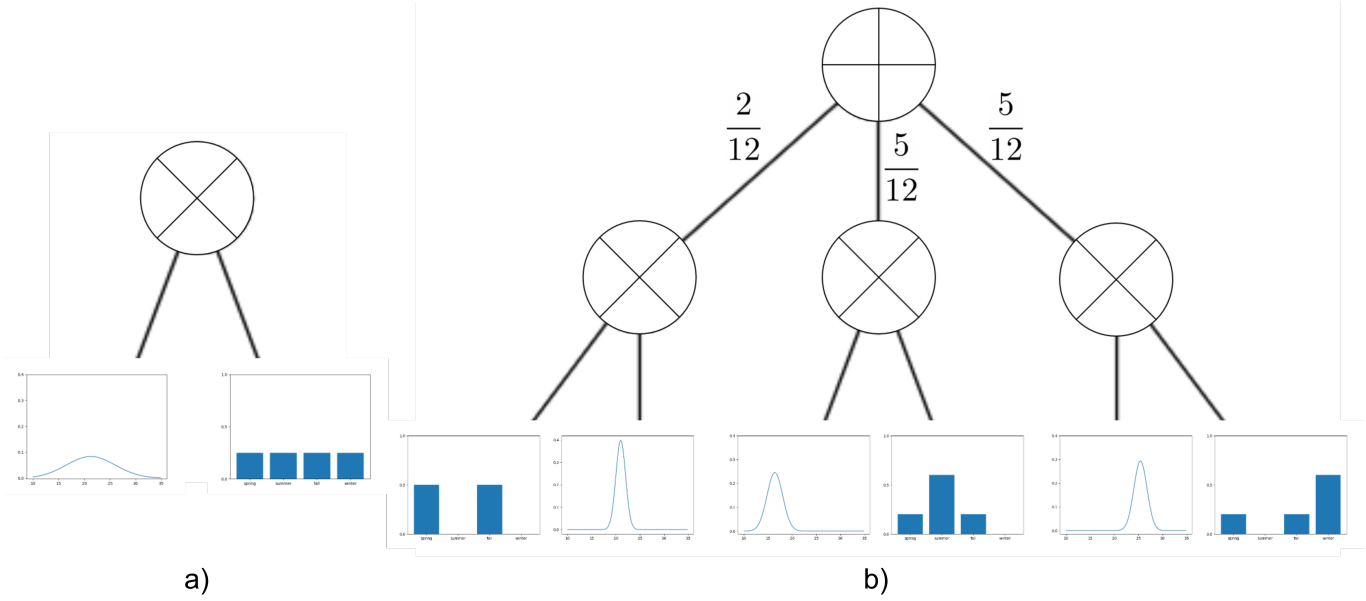

Figure 4.5.: Learning the structure of environmental SPNs
a) An SPN learned with the LearnMSPN algorithm using a min instances ratio $m = 1$. This flat SPN is the simplest that can be learnt and consists of just one product node that factorizes all present RVs. These leafs encode one probability distribution for each RV that summarize all respective values.
b) If $m < 1$, the structure learning algorithms can construct deep networks by finding clusters in subsets of RVs and assessing context-sensitive statistical independence. This SPN exhibits roughly similar parameters as the SPN learned by hand, but also some differences: LearnMSPN has split the clusters also in three parts, from left to right: one for midsize counts ($\mu = 21, \sigma = 1.0$)[3], one for low counts ($\mu = 16.4, \sigma = 1.625$), and one for high counts ($\mu = 25.4, \sigma = 1.356$). The clusters are not strictly separated by the associated season, but rather split by the count values. Therefore, the summer and winter clusters also contain instances with similar values from spring and fall.

LearnMSPN also has the advantage that we can easily incorporate environmental variables, as the use of the rdc-split algorithm allows to test for statistical independence between different types of random variables, in particular between count RVs and categorical RVs. To learn SPNs that additionally contain environmental information, the respective random variables just have to be added to the dataset used in the learning algorithm. Beside that, the procedure remains the same as described for the learning of count SPNs above.

Two examples for learned SPNs based on the example dataset that was used for the hand-designed SPN in figure 4.4 are shown in figure 4.5. Flat SPNs like the one on the left side of the figure will be used as a baseline

[3]This cluster only consists of two datapoints that both have a value of 21. Thus, the estimated Gaussian distribution would be degenerated with a standard deviation of 0. Such distributions can lead to problems during inference, as they declare every value that is not close to 21 as suspicious. Therefore, we correct the standard deviation by setting it to a minimum of 1. In fact, the correction is applied on the fly during the evaluation of new instances rather than in the SPNs itself.

to evaluate if more complex SPNs of health data can approximate the observed data better than flat ones, so if learning fine-grained hierarchical models like on the right side is benefiting for the task of disease outbreak detection.

The advantage of incorporating environmental variables explicitly is that we can condition on the set of observed environmental conditions of a day of evaluation. This leads to an SPN that models the distribution of data that was collected under the same circumstances and results in a more appropriate data-baseline for that day. This adjusted data-baseline should lead to earlier detection of disease outbreaks while decreasing the false alarm rate.

4.3. Syndromic Surveillance with Sum-Product Networks

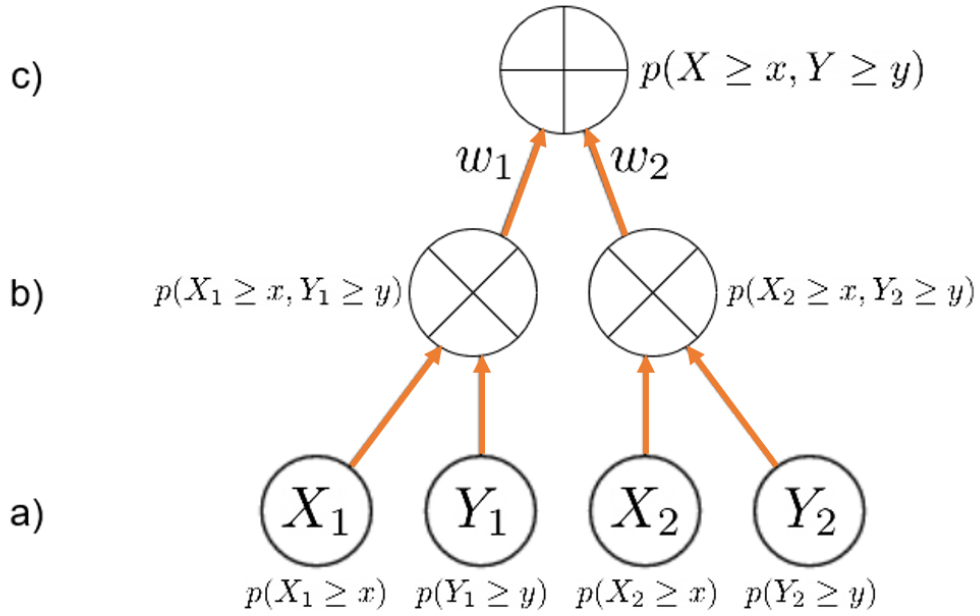

Figure 4.6.: Computation of p-values in Sum-Product Networks
a) In each leaf node, the p-value is computed and propagated to its according parent.
b) At product nodes, the independent tests need to be combined into one p-value.
c) At sum nodes, the weighted average yields a good approximation for combining tests on the same set of random variables.

The task of syndromic surveillance is to assess how likely new observations of syndromes are with respect to our learned data-baseline and if they indicate a possible disease outbreak. As disease outbreaks lead to an increasement of the number of infections (and therefore reported cases), we only have to test if a suspicious high number of counts of a syndrome are reported. For this purpose, we use one-sided hypotheses tests and construct the null hypothesis H_0: The observed value was generated by the same underlying distribution as our previous observations. If the probability of the observed value is low[4], we reject H_0 and conclude that a disease outbreak may have happened. To evaluate the null hypothesis, according p-values are computed,

[4]In standard hypotheses test, the significance value α has to be prespecified. We will follow a slightly different approach, which will be introduced in a moment.

which represent the probability that a value *at least as extreme as the observed ones* occur. For a RV X and its estimated probability distribution p_X, the one-sided right-tailed p-value is $p(X \geq x_{new})$.

The computation of p-values in SPNs follows roughly the same procedure as the inference methods presented in section 2.4.2 and is illustrated in figure 4.6, but differs in two aspects: In the leafs, we compute the right-tailed p-value of the respective leaf rather than the probability of occurrence. At product nodes, we cannot simply multiply the p-values, because the multiplication of independent hypotheses tests does not yield the p-value of the joint distribution. Instead, we need to use methods for combining p-values that return appropriate values that resemble a p-value for the distributions encoded by the respective nodes.

4.3.1. Combining p-values

A valid p-value has to represent the frequency ratio of rejected null hypotheses. For example, if the significance level α is 0.05, 5% of independent and identically sampled p-values should lead to the rejection of H_0. The frequency of rejected H_0 has to equal α, regardless of the number of tests. In a geometric interpretation, the relative area of the rejection zone has to equal α, regardless the number of dimensions. When evaluating hypotheses tests in SPNs, the p-values computed in the leaf nodes need to be combined at sum and product nodes. At product nodes, the standard operation of multiplying the values of the statistically independent children does not yield a correct p-value for the combination of those values and needs to be replaced. At sum nodes, the weighted average produces an outcome that approximates the p-value of the mixture model represented by the node.

Combining p-values at Product Nodes

When we want to conduct multiple tests on the same data, we encounter the multiple hypotheses testing problem. This states that if we perform a number of tests, some will undercut the prespecified rejection threshold α by chance alone, e.g. given $\alpha = 0.05$ and 1000 performed tests, 50 of them should be rejected. In other words, the true value for the rejection threshold of multiple tests is $\alpha = 1 - (1 - \alpha)^n$, where n is the number of performed tests.
Let's take a look at three approaches to combine p-values of independent tests and how well they approximate the rejection zone of α for joint hypotheses tests. First, as stated above, we could multiply the p-values at product nodes. But this has a major disadvantage: H_0 is rejected if the p-value $< \alpha$, and multiplying values in the range (0, 1) (let's assume here that all events (more than one) can happen) always yields a lower value for the product than for each multiplier. If we multiply two p-values, one of 0.05 and one near 1, then we could always find that the occurrence of both together is statistically significant. Moreover, if both tests are clearly not statistically significant and have a p-value larger than 0.05, then the multiplication also can yield a p-value that is lower than this threshold. So, for some RVs $\mathbf{X} = \{X_1, ..., X_n\}$ and observations $\mathbf{x} = \{x_1, ..., x_n\}$,

$$p(\mathbf{X} \geq \mathbf{x}) \neq \prod_{i=1}^{n} p(X_i \geq x_i) \tag{4.1}$$

Tippett [42] proposed to not multiply multiple p-values, but take the minimum value of them. When we consider the two problematic cases of multiplication, we can rule one of them out: The combined p-value can

now only be significant, if at least one of the tests is significant. But if only one is significant and the other p-values are near 1, the same problem as with multiplication still persists.

$$p(\mathbf{X} \geq \mathbf{x}) \approx min(\{p(X_i \geq x_i), \forall i \in [1, n]\}) \tag{4.2}$$

Fisher's method [8] yields a good approximation for combining p-values of independent tests that were conducted under the same null hypothesis. As product nodes factorize their scope into statistically independent sets of random variables and in this case every computation within the SPN tests for the same H_0, we can use it to aggregate the p-values of the children into one proper test-statistic. For n independent tests, Fisher's method constructs a χ^2 distributed test-statistic with $2n$ degrees of freedom. The resulting p-value represents an approximation of the probability that the set of null hypotheses $\{H_0^i\}$ is falsely rejected.

$$p(\mathbf{X} \geq \mathbf{x}) \approx \chi^2_{2n} \sim -2 \sum_{i=1}^{n} log(p(X_i \geq x_i)) \tag{4.3}$$

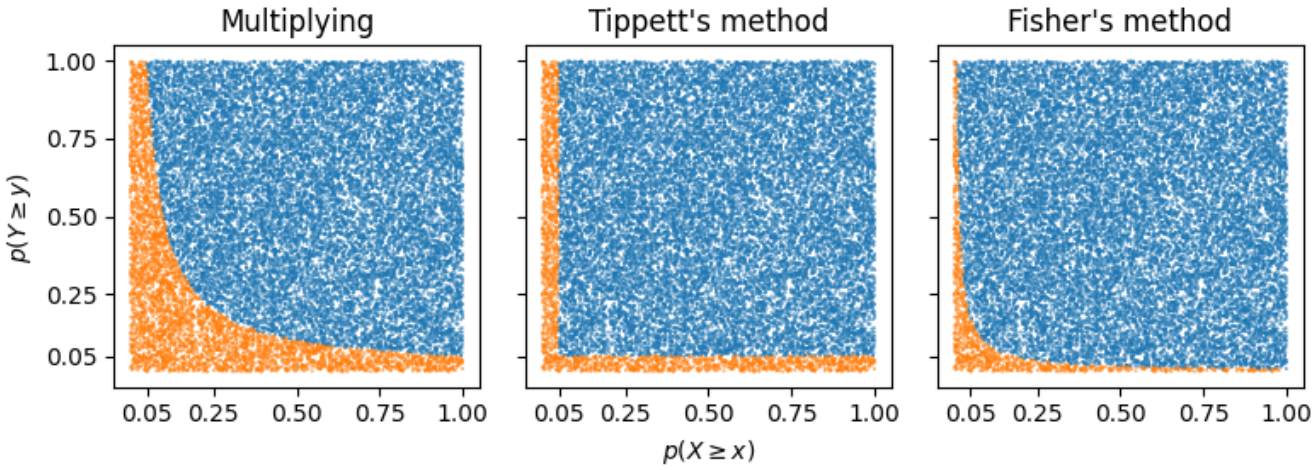

Figure 4.7.: Empirical evaluation of the rejection zones of methods for combining independent p-values

20.000 uniformly distributed p-values of two RVs X, Y. Orange points represent combined p-values whose value is lower than 0.05 and lead to the rejection of H_0.
Left: Multiplying p-values drastically understates the p-value for the combination of two independent tests.
Middle: Tippett's method takes the minimum of all tests. It's a better approximation to the true combined p-value, but still leads to too many falsely rejected H_0.
Right: Fisher's method for combining p-values yields the nearly correct proportion of rejected null hypotheses.

In figure 4.7, the rejection zones of the three presented methods are pictured on the basis of two independent sets that each contain 20000 uniformly distributed p-value samples. It can be easily seen that the multiplication of p-values yields a rejection zone that occupies much more than 5% of the total area, particularly 19.975% for this empirical evaluation. Tippett's relative area of rejection can be easily calculated, it is $1 - (1 - \alpha)^n$, where n is the number of tests. In the 2-dimensional case with $\alpha = 0.05$, this equals to 9.75%, which is close to the empirical value of 9.83%. This is half of the area compared to the multiplication of p-values, but still roughly twice than α. We can also see that Tippett's method is a reflection of the formula for the multiple

hypotheses testing problem given above and will perform worse when combining more than two tests, which eventually leads to a rejection ratio of nearly 100% for lots of tests. Fisher's method, which constructs a proper test-statistic based on the p-values of independent random variables, yields an area of 4.975%, which is very close to our specified rejection threshold and conserves α in the multi-dimensional case, leading to a robust technique for combining independent hypotheses tests.

Combining p-values at Sum Nodes

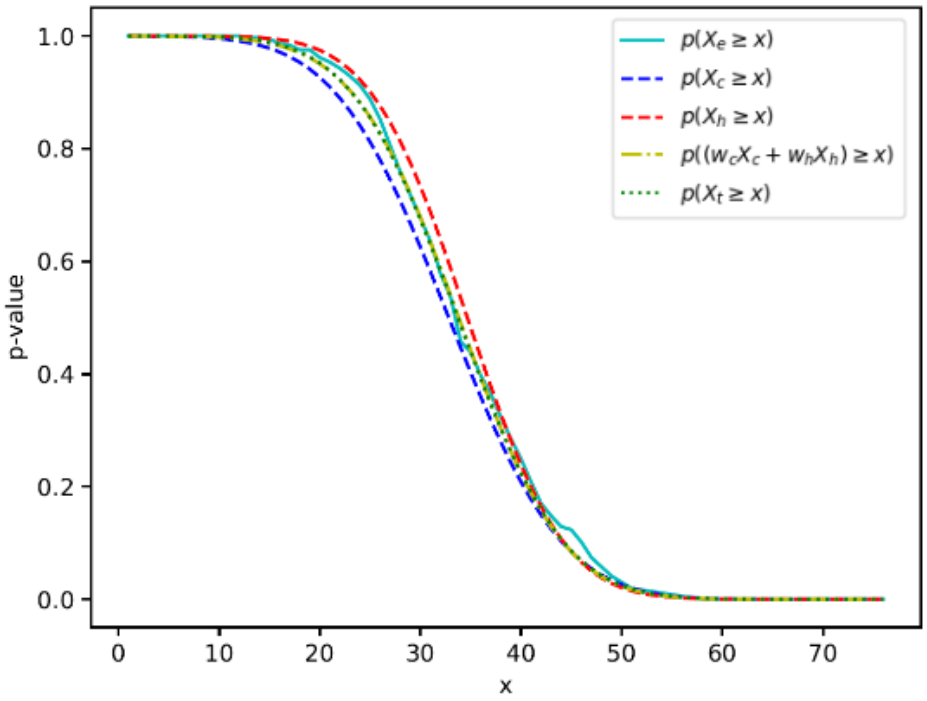

Figure 4.8.: Empirical evaluation of p-values of a mixture distribution
Comparison of p-values from an empirical distribution and the estimated mixture distribution with its cluster components. X_e is the empirical distribution of the total admissions in a simulated emergency room. This data was split into days with cold weather and hot weather, represented by $X_c = \mathcal{N}(32.82, 8.84)$ and $X_h = \mathcal{N}(34.66, 7.51)$. Their p-values under-/overstate the empirical distribution one's for most values of x in 3σ around the respective μ. $p((w_c X_c + w_h X_h) \geq x)$ is the mixture distribution of the cold and hot cluster with $w_c = 0.51$ and $w_h = 0.49$. $X_t = \mathcal{N}(33.73, 8.26)$ represents the estimated Normal distribution from all datasets. The mixture distribution and X_t are similar and represent the best approximation to the empirical p-value.

Recall that sum nodes model a mixture model of the joint distribution of their scope and each child represents a disjunct set of instances that are described by the same set of random variables. p-values of mixture models differ from combined p-values of statistically independent probability distributions, as they are not aggregating probabilities of different random variables, but differently parametrized distributions of the same RVs. Accordingly, sum nodes have normalized weights which describe the probability of each cluster of the respective children. Therefore, multiplying the p-value of each child with its weight and adding those values up results in a valid probability. Figure 4.8 shows that the weighted average of p-values yields a good

approximation to the p-value of the mixture distribution and of the empirical distribution. Hence, we want to use the weighted average mean at sum nodes to evaluate the p-values of syndrome counts.

4.3.2. Evaluating Syndrome Counts

With appropriate methods for combining p-values in SPNs at hand, the question of how to evaluate syndrome counts for disease outbreak detection in detail is still open. An outline of the main idea is pictured in 4.6, but we still need to determine the data-baseline, the evaluation procedure and the alarm threshold.

First, the patient records of a day to test are collected and preprocessed to syndrome counts as described in section 4.1.2. These will be referred to as *current data*, *current day*, or *current syndromes*. As the number of possible combinations of personal attributes grow exponentially with the syndrome size, we limit the evaluation to syndromes of size 1 and 2, so we can detect anomalies in single characteristics like a particular age range or medical treatment, but also pairs of them. Depending on the number of modeled variables and the available computational resources, this can possibly be extended to syndromes of slightly larger size. The evaluation strategy now depends on the type of SPN we have learned. I will detail the procedure for SPNs with environmental information first and cover SPNs only containing syndrome counts later.

Evaluation in SPNs with Environmental Variables

In SPNs with environmental attributes, we can obtain a data-baseline that contains estimated distributions from observations that were collected under the same circumstances as the current day by conditioning the SPN on those environmental conditions. As the structure of reported cases in the emergency room changes throughout the year and some observed values as the count of respiratory symptoms depend on those environmental conditions, the conditioned SPN should represent a more accurate distribution for the respective setting. Computing the conditional p-value requires computing the marginal p-value of a syndrome $\mathbf{S}$ and the probability of a set of n environmental variables $\mathbf{E}$ and divide the joint of them by the latter.

$$p(S_i \geq s_i | E_j = e_j) = \frac{p(S_i \geq s_i, E_j = e_j)}{p(E_j = e_j)}, i \in \{1, 2\}, j \in 1, ..., n \tag{4.4}$$

However, the combination of p-values with probabilities would make the inference procedure cumbersome. Instead, we condition the SPN on the set of environmental observations, which results in an SPN that does not contain any environmental random variable anymore, and has adjusted weights such that it represents the estimated joint distribution of all syndromes for that specific environment. But we do not want to assess all syndromes simultaneously, but test for anomalies in the marginal distributions of all syndromes of size 1 and 2. An illustration of a more complex SPN containing multiple syndromes and environmental variables and the process of conditioning to obtain an SPN for the evaluation of syndromes is pictured in figure 4.9.

The conditioned SPN can now be used to conduct hypotheses tests on all syndromes of size 1 and size 2. We have to care that we do not evaluate syndromes that belong to the same personal attribute, so we may not evaluate the count of males and females $p(S_1 \geq s_1, S_2 \geq s_2 | E_1 = e_1, E_2 = e_2)$ together. All other syndrome counts that are not part of the current evaluation do not propagate values to their parent nodes and are

syndromes				environment	
gender	gender	age	symptom		
#female	#male	#working	#respiratory	weather	day of week
S_0	S_1	S_2	S_3	E_0	E_1
12	18	10	8	cold	weekday
18	25	20	17	cold	weekday
21	15	22	12	cold	weekday
20	15	12	12	cold	weekday
28	19	21	11	cold	weekday
21	18	17	10	cold	saturday
24	16	21	12	cold	sunday
12	9	10	7	hot	weekday
16	16	15	9	hot	weekday
14	15	14	6	hot	weekday
16	9	10	5	hot	weekday
8	11	11	5	hot	weekday
10	17	14	5	hot	saturday
12	20	15	8	hot	sunday

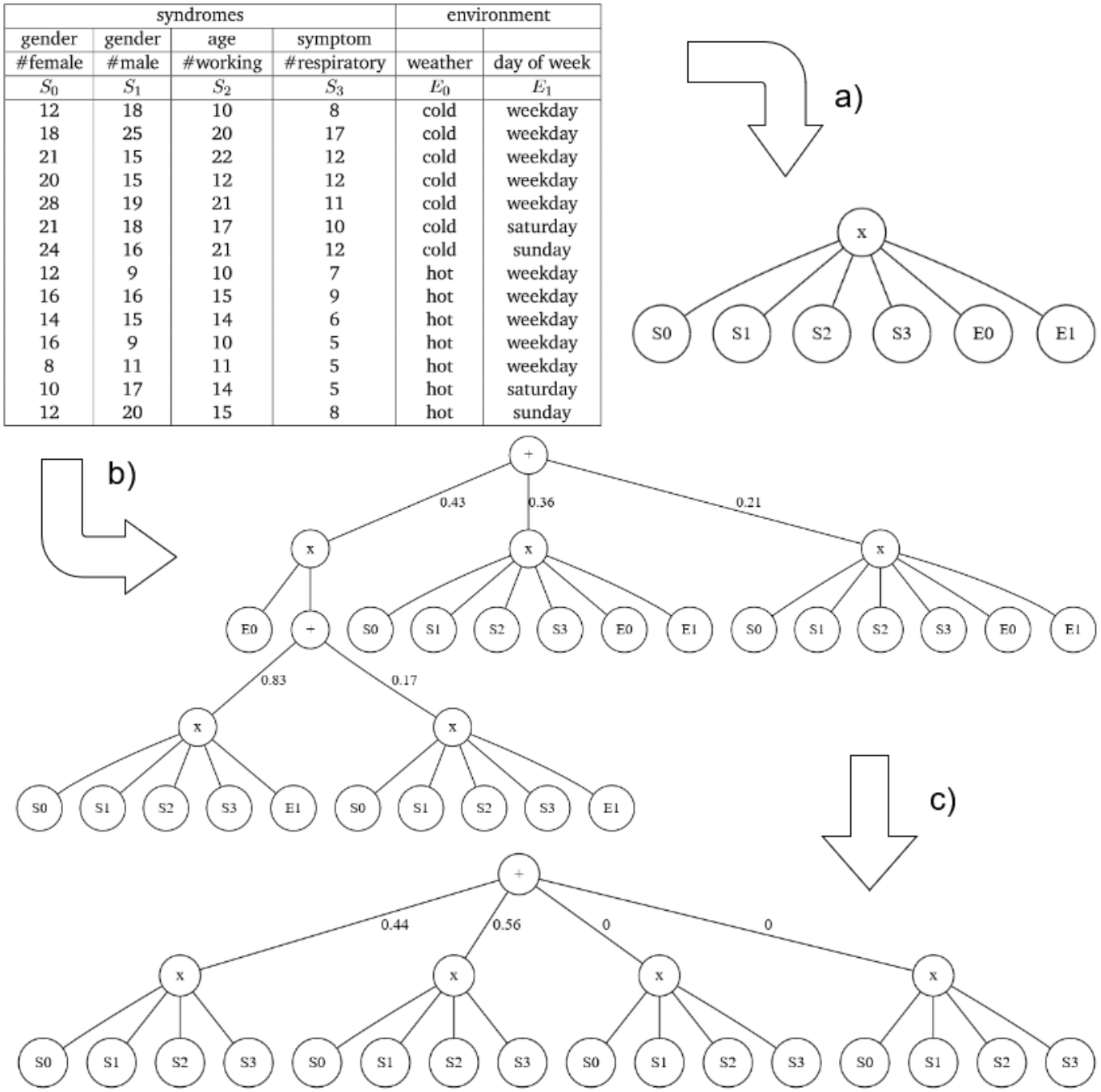

Figure 4.9.: Evaluation of syndromes in an SPN containing environmental variables
Top-left: Data from a simulated emergency room. It contains reported cases of one week in the winter and one week in the summer each. Top-right (a): The simplest SPN that can be learned is a fully factorized model that does not account for any dependency in the data and uses one probability distribution per RV. Conditioning on the environment would only eliminate E_0, E_1, but not alter the distributions of the syndromes.
Middle (b): A deeper SPN that models the splitting and clustering according to the environmental variables. The most left product node on the second level describes the cluster of cold days, represented by E_0 and the neighbored sum node. Bottom (c): Conditioning on a weekday with cold weather results in an SPN without environmental variables and adjusted weights according to those environmental observations. It models the appropriate data-baseline for the current day and can be used to evaluate p-values of syndromes of different sizes.

ignored. Given n syndromes and m environmental variables and $a_P(S_i)$, which maps the syndrome to its respective personal attribute, for each day we compute:

$$\begin{aligned} &\{p(S_i \geq s_i | E_k = e_k)\} \cup \{p(S_i \geq s_i, S_j \geq s_j | E_k = e_k)\} \\ &\forall i, j \in \{1, ..., n\}, i \neq j, a_P(S_i) \neq a_P(S_j), \forall k \in \{1, ..., m\} \end{aligned} \tag{4.5}$$

For the exemplary data in figure 4.9, in the conditioned SPN, we evaluate the p-values of the syndrome counts $\#\text{female}, \#\text{male}, \#\text{working}, \#\text{respiratory}, (\#\text{female}, \#\text{working}), (\#\text{female}, \#\text{respiratory}), (\#\text{male}, \#\text{working}), (\#\text{male}, \#\text{respiratory})$ given the respective observations of the current day, and collect all results.

Now, we have different options to aggregate the p-values and decide if an alarm should be given or not. We can either take the minimum result of the syndromes according to Tippett's method and throw an alarm if this value is lower than a specified α, e.g. 0.05. We could also present a subset of n most suspicious syndromes. Alternatively, we could combine the p-values with Fisher's method on either all values or a subset only. However, combining all results into one statistic has the disadvantage that we loose the information of which specific syndrome is suspicious and therefore responsible for the alarm.
I followed the first approach and took the minimum value and its associated syndrome of all evaluated tests. This could lead to an increasement in falsely rejected null hypotheses and therefore false alarms, but as we have test datasets available, α can also be calibrated afterwards by evaluating the false positive rate on the test data. After all, the task of this systems is not to compute the exact probability that values as extreme as the observed ones occur, but to deploy a surveillance support system that helps to determine if there are combinations of syndromes that seem to be suspicious and are worth an investigation by the medical staff. Therefore, optimizing the alarm threshold by means of the false alarm rate on a test dataset is a reasonable approach.

Evaluation in SPNs without Environmental Variables

When the data does not exhibit any environmental information and cannot be easily enriched with either, we can still try to detect disease outbreaks in SPNs that model only count data. The syndromes of different sizes can be evaluated in the same manner as in SPNs with environmental information, but the step of conditioning on the non-existent environmental variables is omitted. This leads to marginalized SPNs that only model the clusters of the syndromes over all estimated distributions and does not account for possible dependencies of the circumstances under which the data was collected. This should lead to a worse performance as suspicious high counts of a symptom that occur frequently in one time of the year, but not another, may not be detected in the latter.
SPNs are capable of modeling the context-sensitive dependencies between the syndrome counts. When evaluating syndromes of size 2, another evaluation strategy would be to condition on one of the syndromes and evaluate the other syndrome with respect to the conditioned distribution. When the evaluated syndromes exhibit dependencies among them that is due to the circumstances the data was collected under, conditioning on one syndrome and evaluating another can implicitly account for the latent environmental information. However, this method may be less robust and accurate and should only be used when the collected health data cannot be enriched with at least some environmental information.

5. Results

Before coming to the results of the experimental evaluation of the just presented approaches, I will introduce a suitable evaluation metric for detecting disease outbreaks early and also walk through experiments that were conducted based on heuristics before the proper methods for combining p-values were incorporated. We will also examine how well the learned Sum-Product Networks approximate the respective training datasets. Depending on the used training data and structure learning algorithms, a range of different SPNs were learnt with a set of pre-specified parameters that control the network size. For each type of SPN, experiments evaluating the procedures presented in 4.3.2 will be discussed.

I used SPFlow for the implementation of experiments, a Python-framework for learning and evaluating Sum-Product Networks that was developed by Molina et al. [28]. SPFlow provides a domain specific language for designing SPNs by hand and computationally efficient implementations of a range of structure and parameter learning algorithms and inference procedures. Particularly, with SPFlow we can easily learn Poisson and Mixed SPNs, and can compute joint, marginal and conditional probabilities or even manipulate SPNs to represent respective distributions by altering their structure and adjusting the cluster weights. The framework also allows to alter the behavior of these algorithms and customize the evaluation-behaviour of SPNs to fit our needs.

5.1. Evaluation Metric

As we want to detect disease outbreaks as early as possible, our evaluation metric should be based on the delay between the onset of a known disease outbreak and the day on which an alarm was given by the system. But the detection delay could be simply reduced to the optimal value of 0, when the system alarms every day. Hence, the metric should also incorporate the false alarm rate. We cannot guarantee that no false alarms are given, but try to bring down the FAR the trustworthiness of such a surveillance support system also builds upon the correctness of the alarms and the amount of work induced by an investigation that follows an alarm. With a corresponding metric, we can optimize the threshold for giving an alarm based on the p-value computed for an evaluation instance and determine the detection delay of different methods with respect to the number of false alarms that the respective methods gives.

5.1.1. Activity Monitoring Operating Characteristic

For the general purpose of evaluating algorithms that aim to detect changes in monitored data, Fawcett et al. introduced the *Activity Monitoring Operating Characteristic* [7], short *AMOC*. The AMOC analysis is a method to assess the quality of anomaly detection systems in the setting of activity monitoring. This means that a constant stream of new instances arriving in fixed time slots is evaluated against an estimated distribution of the monitored objectives that is determined to be a normal status. Syndromic surveillance is a type of

activity monitoring problem and in our case, we receive new instances daily and want to compare them with a outbreak-free distribution to detect changes in the number of infections within a region.

The AMOC analysis is similar to the wide-spread Receiver Operating Characteristic (ROC) analysis and derived from it. ROC curves are an evaluation method for binary classification algorithms by comparing the false positive rate (FPR) with the true positive rate (TPR, also called sensitivity or recall), both in the range of $[0, 1]$. An optimal classifier would return a TPR of 1 and a FPR 0. To compare ROC curves for different algorithms, the area under the ROC curve is calculated, where the area of 1 resembles an optimal classifier and 0.5 a random classifier. Detection-ROC curves could also be used to evaluate syndromic surveillance systems by comparing the detection rate, the number of outbreaks for which an alarm was actually given, with the FPR. But as the AMOC analysis builds on a similar concept as the ROC analysis, the experiments are evaluated based on the detection delay rather than the detection rate.
To construct the corresponding metric for our models, first, an AMOC-curve has to be computed according to the algorithm given below (adapted from [7]):

Algorithmus 3 : AMOC($\{t, p\}, o, d$)

Input : Set of tuples $\{(t, p)\}, o, d$, where p is the p-value evaluated at day t, o is the day of the outbreak and d is the duration of the outbreak in days
Output : R: Set of points on AMOC and according alarm threshold
$S = d$ /* detection delay in days, set to maximum of d */
$F = 0$ /* false alarm rate */
$R = \{(0, 14, 1)\}$
sort $\{(t, p)\}$ in descending order by p
for $(t, p) \in \{(t, p)\}$ **do**
  if $o \leq t \leq (o + d)$ **then**
    /* t is in the period of the outbreak */ $S = \min(S, (t - o))$ /*
  end
  else
    $F = F + 1$
  end
  Add point (F, S, p) to R
end
for $(f, s, p) \in R$ **do**
  $f = \frac{f}{F}$ /* normalize false alarms */
end

The set of points R can be used to plot the according AMOC-curve. The evaluation metric for the comparison of models is the area under curve (AUC) of the AMOC, which returns the average detection delay in days with respect to the false alarm rate. But as we can decrease the detection delay to 0 when we give an alarm for every day and syndromic surveillance systems rely on a low false alarm rate to be applicable in the real world, we compute the area up to the 5% false alarm rate. This value reflects the average detection delay, when on average up to every 20th alarm is given falsely. The optimal value of AMOC-AUC5 is 0 days and the worst 14 days, which is the maximum an outbreak lasts. The selected FAR threshold can then be used to determine the respective alarm threshold α, for which the daily p-values have to fall below in order to give an alarm. All values regarding the results of syndromic surveillance experiments are given as AMOC-AUC$\leq 5\%$.

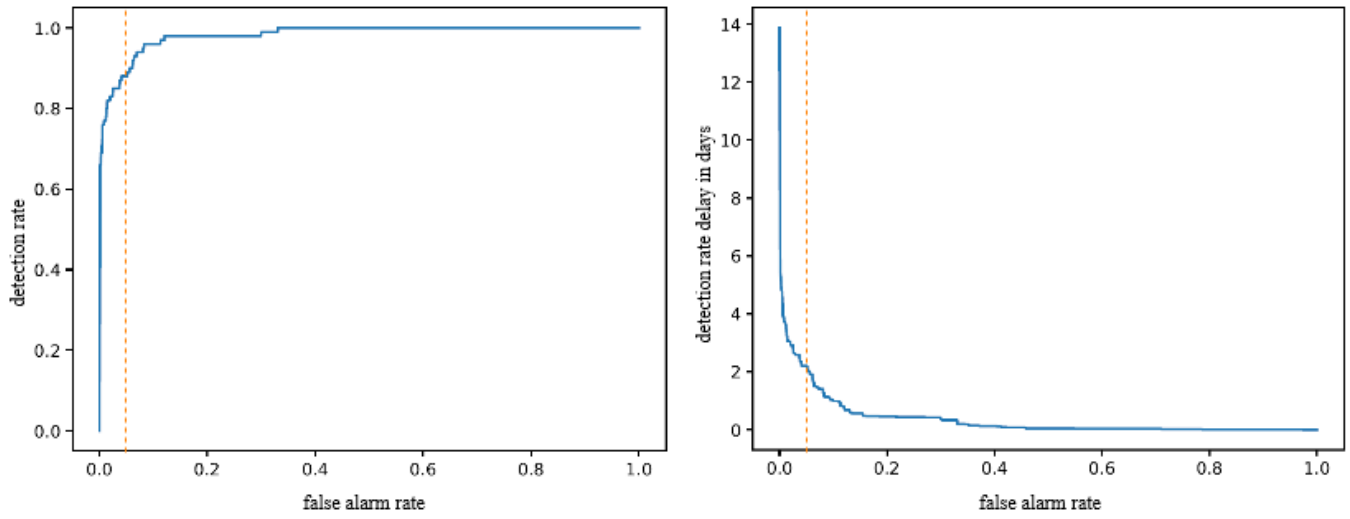

Figure 5.1.: Exemplary dROC and AMOC curves
Left: The detection-ROC-curve shows the FAR versus the detection rate.
Right: The AMOC-curve shows the FAR versus the detection delay in days.
The two metrics are often anti-proportional for most of false alarm rates. The orange dotted vertical line represents the 5% FAR and the left-side area-under-curve will be used as evaluation metric. For the AMOC, this value gives us the average detection delay for FAR$\leq 5\%$.

5.2. Syndromic Surveillance Experiments

In the following section, I detail the different Sum-Product Networks that were learned and evaluate how well they approximate the training datasets. Afterwards, the conducted experiments and their underlying ideas are presented and their results discussed.
The synthetic emergency room data consists of 100 datasets that cover two years each and expose a disease outbreak of up to 14 days in the second year. The first year of each dataset is used to learn respective SPNs, and the second year is evaluated on the same. The SPNs do not get updated and are not relearned, instead the training is used to evaluate all instances of the test year. For each of the 100 datasets, SPNs with different parameters are learned. These parameters are the threshold for the splitting algorithm r and the ratio of minimal instances m that are used to learn leafs in the SPN. To compare the SPN regarding the advantage of building deeper models, r will be set to 0.3 for all learned networks, whereas $m \in \{1.0, 0.7, 0.5, 0.4, 0.3, 0.2, 0.1, 0.05\}$. For the experiments with environmental variables, the SPNs with $m \in \{0.7, 0.4\}$ are missing due to late evaluation and limited time. The SPNs with $m = 1.0$ only consist of one product node whose children are single distributions in the leaf learned over all instances of each random variable, which is equal to a fully factorized model. Those SPNs will serve as baseline to determine if learning deep probabilistic models improves upon the simple modeling approach.

5 types of SPNs were learned: 3 modelling only syndrome counts, where one only consists of Gaussian nodes learned with *LearnMSPN*, and two only consist of Poisson Nodes, one learned with *LearnMSPN* and one with *LearnPSPN*, which differ in the splitting algorithms for assessing statistical independence between RVs. Learning one SPN for each m results in 800 SPNs per type. As the Negative Binomial distribution cannot be estimated with MLE, the according evaluation will be executed on the fly by approximating the NB parameters from the learned Gaussian SPNs.

As estimating the distribution of syndrome counts *and* environmental variables requires splitting between different types of RV, only LearnMSPN implementing the rdc-splitting algorithm can be applied. Therefore, 600 SPNs each were learned with Categorical and Gaussian respective Poisson nodes were learned.

The k-means algorithm was used as clustering operation for LearnMSPN. A small value for k that minimizes the sum of the squared distances of the observations to the k clusters is determined iteratively. The resulting cluster centers separate the datapoints into disjunct segments that either are used to learn a leaf with a respective univariate probability distribution estimated with MLE if the segments only contain one RV or their number of instances is smaller than m. If not, the structure learning algorithms try to split the clusters into approximate independent sets of RVs via the rdc-split or poisson-split operation illustrated in section 2.4.1.

5.2.1. Relative Goodness of Fit

The average goodness of fit of each type and size of SPN is determined by computing the log-likelihood for each instance of the training year and averaging all values. Then, the average log-likelihood is again averaged over all 100 datasets. The results can only be compared among models that were learned from the same data, so the tables are split into count-only SPNs 5.1 and SPNs with environmental variables 5.2. The average size of the SPNs in the format (#leafs, #sum nodes, #product nodes) is also displayed.

	Gaussian SPN LearnMSPN		Poisson SPN LearnMSPN		Poisson SPN LearnPSPN	
learning parameters	avg LL	avg size	avg LL	avg size	avg LL	avg size
$r = 0.3, m = 1.00$	-63.03	(25|0|1)	-63.34	(25|0|1)	-63.34	(25|0|1)
$r = 0.3, m = 0.70$	-59.69	(50|1|2)	-	-	-59.34	(50|1|2)
$r = 0.3, m = 0.50$	-58.63	(74|2|3)	-58.43	(75|1|3)	-58.43	(74|1|3)
$r = 0.3, m = 0.40$	-57.93	(98|2|4)	-	-	-57.89	(99|2|4)
$r = 0.3, m = 0.30$	-57.32	(136|2|5)	-57.58	(137|1|5)	-57.58	(135|3|7)
$r = 0.3, m = 0.20$	-56.90	(173|2|7)	-57.35	(175|1|7)	-57.36	(169|6|10)
$r = 0.3, m = 0.10$	-55.48	(374|2|15)	-56.64	(379|1|15)	-56.73	(323|21|47)
$r = 0.3, m = 0.05$	-53.23	(760|3|31)	-56.02	(766|1|30)	-56.55	(478|51|125)

Table 5.1.: Log-likelihoods and average sizes of count-only SPNs

	Gaussian Counts LearnMSPN		Poisson Counts LearnMSPN	
learning parameters	avg LL	avg size	avg LL	avg size
$r = 0.3, m = 1.00$	-69.45	(30|0|1)	-69.75	(30|0|1)
$r = 0.3, m = 0.50$	-64.65	(88|2|3)	-64.41	(89|1|3)
$r = 0.3, m = 0.30$	-63.07	(161|3|6)	-63.18	(164|1|5)
$r = 0.3, m = 0.20$	-62.47	(205|3|8)	-62.70	(210|1|7)
$r = 0.3, m = 0.10$	-60.24	(435|3|16)	-60.97	(446|1|15)
$r = 0.3, m = 0.05$	-57.47	(878|6|33)	-59.68	(900|3|31)

Table 5.2.: Log-likelihoods and average sizes of SPNs with environmental variables

Log-likelihoods that are nearer to 0 are better. We can see that more complex hierarchical distributions modeled by larger SPNs approximate the training data better than simpler ones for each learned type. For count-only SPNs, the LearnMSPN algorithm with k-means and rdc-split tend to find only few clusters. LearnPSPN leads to deeper and more fine-grained Poisson SPNs, which perform slightly worse than the Poisson SPNs learned

with LearnMSPN. Also, the Gaussian SPNs approximate the training data better than the respective Poisson SPNs while being similar in size.

5.2.2. Evaluation Strategies in Count-only SPNs

First, I learned and evaluated count-only SPNs and tried to detect disease outbreaks with heuristic strategies, which I will present in detail, before coming to experiments that use the previously presented methods to combine p-values at internal nodes. While incorporating environmental information helps to determine according data-baselines, conditioning on observation of syndrome counts rather than the environmental conditions can also yield low detection delays that are competitive with the latter.

Find the Most Suspicious Cluster

As we have learned a fine-grained hierarchical clustering of the joint distribution of the emergency room data, the first ides was to find the most suspicious cluster of all syndromes in the SPN given the observations of a new instance. Count-only SPNs don't contain any environmental information, and to account for the latent dependencies between syndrome counts given the circumstances the training data was collected under, we condition the SPN on one syndrome and want to find the cluster in the conditioned SPN with the lowest p-value. So, for each syndrome count in the test instance, the SPNs are individually conditioned onto the respective value. Then, all other observations of syndrome counts are set at the according leafs and the p-values for each univariate distribution are computed. Sum and product nodes are both replaced with min nodes that simply propagate the minimal p-value to their parents. For each syndrome count per day, one p-value $p(S_i \geq s_i | S_j = s_j)$ is returned. We take the minimum of the p-values of all evaluated syndromes as the p-value for the test instance.

learning parameters	Gaussian MSPN	Poisson PSPN	NegBinom from GSPN
$r = 0.3, m = 1.00$	0.8678	1.2713	0.9735
$r = 0.3, m = 0.70$	1.1710	1.7989	1.8617
$r = 0.3, m = 0.50$	4.1956	2.7907	2.8828
$r = 0.3, m = 0.40$	5.0041	2.5951	2.6963
$r = 0.3, m = 0.30$	6.6995	2.6916	3.3474
$r = 0.3, m = 0.20$	11.1648	3.1427	3.6728
$r = 0.3, m = 0.10$	12.8569	6.5964	5.2711
$r = 0.3, m = 0.05$	13.4364	13.0379	13.0852

Table 5.3.: Results of Min-Min Networks conditioned on one observed count

The baselines perform well, but the larger the SPN, the worse the detection delay and rate. This has the following reasons: When we split the training data into small clusters, for each syndrome count, the single clusters components are differently parametrized. This can also be seen in figure 4.8. If we observe many low values in the summer and many high in the winter, the SPN would learn one distribution for each season with the respective parameters. As the clusters with lower values will *always* return the smallest p-value for that syndrome count, this value is always propagated upwards (until smaller ones are present). This means that we're always comparing the syndrome counts to the same cluster with the lowest parameter values, no matter if it is an appropriate choice or not. Therefore, larger SPNs perform worse, as they model more clusters

that cover less values and therefore have distributions with relative low parameters. We can clearly see that evaluating the syndrome counts in larger SPNs does not work with this approach.

Tippett's Method with Averaging

As taking the minimum p-value at every internal node of the SPNs does not work, I conducted experiments where the product nodes are still substituted with min nodes, but the sum nodes operate as usual computing the weighted average of the children. Taking the minimum value of the statistically independent children at product nodes results in applying Tippett's method. For the sum-nodes that have leafs as children, which represent mixture models, the weighted average results into a good approximation of the p-value of the mixture model. When we only condition the SPNs on one observed count (for each RV), but do not marginalize the SPNs on syndromes with fixed size, it is not clear what the p-value at internal nodes actually represents. As we have networks containing all syndrome counts that get aggregated at the sum nodes, the resulting value was possibly computed from a lot of different clusters of syndromes and we cannot identify a single suspicious syndrome, but only tell if the test instance is suspicious with respect to the SPN.
The table 5.4 shows results from SPNs which were not conditioned at all, but only marginalized on the syndromes of size 1 or size 2. Hence, in the marginal approach, we can identify the single syndrome that is responsible for an alarm. The table 5.5 uses SPNs that were conditioned and evaluated in the same manner as in the previous experiment that attempts to find the most suspicious cluster. This leads to a value that combines an unknown set of RV, but is still capable of detecting disease outbreaks.

learning parameters	Gaussian MSPN	Poisson PSPN	NegBinom from GSPN
$r = 0.3, m = 1.00$	1.1527	2.1269	1.3789
$r = 0.3, m = 0.70$	1.1348	1.6698	1.5389
$r = 0.3, m = 0.50$	1.1109	1.5360	1.3925
$r = 0.3, m = 0.40$	1.2304	1.4578	1.3380
$r = 0.3, m = 0.30$	1.2720	1.3878	1.3365
$r = 0.3, m = 0.20$	1.2917	1.3975	1.3637
$r = 0.3, m = 0.10$	1.3722	1.2244	1.3382
$r = 0.3, m = 0.05$	1.3853	1.2443	1.3406

Table 5.4.: Results of Sum-Min Networks marginalized on syndromes

learning parameters	Gaussian MSPN	Poisson PSPN	NegBinom from GSPN
$r = 0.3, m = 1.00$	0.8592	1.3121	0.9640
$r = 0.3, m = 0.70$	0.8258	1.0532	0.8864
$r = 0.3, m = 0.50$	0.7436	0.9747	0.8249
$r = 0.3, m = 0.40$	0.8326	0.9613	0.8119
$r = 0.3, m = 0.30$	0.8249	0.9277	0.8936
$r = 0.3, m = 0.20$	0.8075	0.9640	0.8676
$r = 0.3, m = 0.10$	0.9804	0.8817	1.0526
$r = 0.3, m = 0.05$	1.8413	0.9456	2.1376

Table 5.5.: Results of Sum-Min Networks conditioned on one observed count

The experiment in the marginalized SPNs already show better results than the previous approach. The baselines perform slightly worse, but a low detection delay of around 1 to 2 days is accomplished for every

type of SPN. SPNs with Gaussian nodes still are not able to improve the detection quality with increasing size, but are more robust than only using min nodes. The Negative Binomial experiment, in which the parameters were derived from the Gaussian, perform somewhat similarly. The baseline of the Poisson SPN has a detection delay up to twice as many days as the other networks, but is the only one that does perform better when we learn larger SPNs until $m = 0.1$.
The second experiment using SPNs containing the distributions of all syndromes conditioned on one of the observed counts eventually shows that the fine-grained clustering learned by the SPN can help to improve the detection delay for all types of leaf distributions. The Poisson SPN again performs better in larger SPNs up to $m = 0.1$. The Gaussian and Negative Binomial networks show a lower detection delay for medium-sized models, but forfeit that advantage when learning even larger SPNs. This may happen due to degenerated Gaussian distributions in the leafs that were estimated from only few training instances, leading to low variances in each cluster.

Fisher's Method with Averaging

Tippett's method is a rather bad approximation to the true p-value as stated in section 4.3.1 and illustrated in figure 4.7. When Tippett is used to combine many p-values, the probability of falsely rejecting H_0 and therefore giving an alarm approaches 1. As we use the average detection delay for $FAR \leq 5\%$ as evaluation metric, the specific computed p-values play a minor role within these experiments, because only the relation between the values for each test instance really matters. However, if we want to the resulting values to resemble approximate p-values, especially when comparing the values of each day to a set α, Fisher's method should be applied. To compute proper p-values of independent tests, Fisher's method is deployed in product nodes. At sum nodes, we still use the weighted average of the children. When we specify an alarm threshold α and alarm if the p-value of the test instance undercuts this threshold, it becomes important to compute good approximations to the true p-value of combined syndromes.
Again, the first experiment 5.6 computes the p-values of syndromes of size 1 and 2 in respective marginalized networks. In contrast to the previous approach with Tippett's method, the experiment with conditioned SPNs showed in table 5.7 operates differently. Instead of only conditioning on one syndrome count and then searching for the most suspicious respective combination, we also marginalize the conditioned SPN on the other syndrome when we evaluate those with size 2. Hence, the resulting p-values are always stemming from the predefined set of syndromes with limited size instead of combining arbitrarily many RV, allowing to tell which syndrome actually is suspicious.
This experiment also compares the results for SPNs with Poisson nodes learnt with LearnMSPN. The difference between the Poisson MSPN and the Poisson PSPN lies in the different clustering and splitting algorithms and therefore their structures and estimated distributions.

learning parameters	Gaussian MSPN	Poisson MSPN	Poisson PSPN	NegBinom from GSPN
$r = 0.3, m = 1.00$	1.1479	2.1155	2.1155	1.4040
$r = 0.3, m = 0.50$	1.1186	1.5656	1.5656	1.3708
$r = 0.3, m = 0.30$	1.2475	1.3919	1.3948	1.2856
$r = 0.3, m = 0.20$	1.2832	1.3982	1.4056	1.3637
$r = 0.3, m = 0.10$	1.4138	1.3312	1.2079	1.3749
$r = 0.3, m = 0.05$	1.4655	1.2905	1.2045	1.3390

Table 5.6.: Results of Sum-Fisher Networks marginalized on syndromes

The experiments using marginalized SPNs only perform very similar to the experiments using Tippett's method

learning parameters	Gaussian MSPN	Poisson MSPN	Poisson PSPN	NegBinom from GSPN
$r = 0.3, m = 1.00$	0.8592	1.3121	1.3121	0.9640
$r = 0.3, m = 0.50$	0.8721	1.0480	1.0480	0.9009
$r = 0.3, m = 0.30$	0.9670	0.9637	0.9785	0.9014
$r = 0.3, m = 0.20$	0.9707	0.9409	0.9609	0.8992
$r = 0.3, m = 0.10$	1.0578	0.9255	0.8492	0.9198
$r = 0.3, m = 0.05$	1.0652	0.8915	0.9748	0.8819

Table 5.7.: Results of Sum-Fisher Networks conditioned and marginalized on syndromes

at product nodes. For larger SPNs with Poisson nodes, LearnPSPN leads to more robust models than LearnMSPN. The second experiment, for which syndromes of size 2 condition the SPN on the first and marginalize the conditioned network on the second syndrome, cannot be directly compared to the respective experiment with Tippett's method. The new evaluation approach combined with Fisher's method for independent tests at product nodes yields low detection delays for all types of learnt SPNs, for the best models less than 1 day on average.

5.2.3. Including Environmental Variables

Chapter 4 mainly focused on learning and evaluating SPNs learned from health data containing environmental information. Elaborating the experiments and according strategies presented so far used up most of the time and evaluating the method based on conditioning on the environmental setting was only conducted just before the end. Therefore, the results of only one experiment are available and discussed.

Fisher's Method with Averaging

All SPNs in this section were learned on training data from the emergency room datasets containing 25 RV describing the counts of syndromes of size 1 and 5 categorical RV describing the condition of the environment. The latter are the season, the weather, the day of week, the flu-level in the simulated city and the month obtained from the dates of instances. For each day of evaluation, the SPNs are conditioned on the whole set of observed values for the environmental variables. The resulting networks still model the joint distribution of all syndrome counts and are separately marginalized on the syndromes of size 1 and 2. The detailed process is described in section 4.3.2 and pictured in figure 4.9.

learning parameters	Gaussian MSPN	Poisson MSPN	NegBinom from GSPN
$r = 0.3, m = 1.00$	1.1479	2.1155	1.4040
$r = 0.3, m = 0.50$	1.0242	1.4551	1.2504
$r = 0.3, m = 0.30$	1.0754	1.1815	1.1821
$r = 0.3, m = 0.20$	1.1177	1.1511	1.2042
$r = 0.3, m = 0.10$	1.1464	1.1373	1.1224
$r = 0.3, m = 0.05$	1.1657	1.0819	1.1039

Table 5.8.: Results of Sum-Fisher Networks with environmental variables marginalized on syndromes

While all types of SPNs perform well, the results for evaluation with Gaussian and Negative Binomial leafs are not monotonously decreasing. Whereas the Poisson SPN accomplishes lower detection delays the larger the SPNs grow in size and therefore is robust regarding this inference procedure. In contrast to the outcome I expected, conditioning on the environmental variables does not yield better results than conditioning on syndromes in the count-only SPNs. This may be due to the environmental variables selected, e.g. it may not be necessary to preprocess the date to months and we could just leave this attribute out. As the presented experiments take up days of computation and the limited time, a feature analysis has not been realized, but should be conducted before deploying such systems in the real world.
Although the results are not better than some of the previous experiments on less complex data, the detection delay for all SPNs are relatively low and still show that disease outbreak detection in such SPNs is possible.

5.2.4. Comparison of Evaluation Strategies

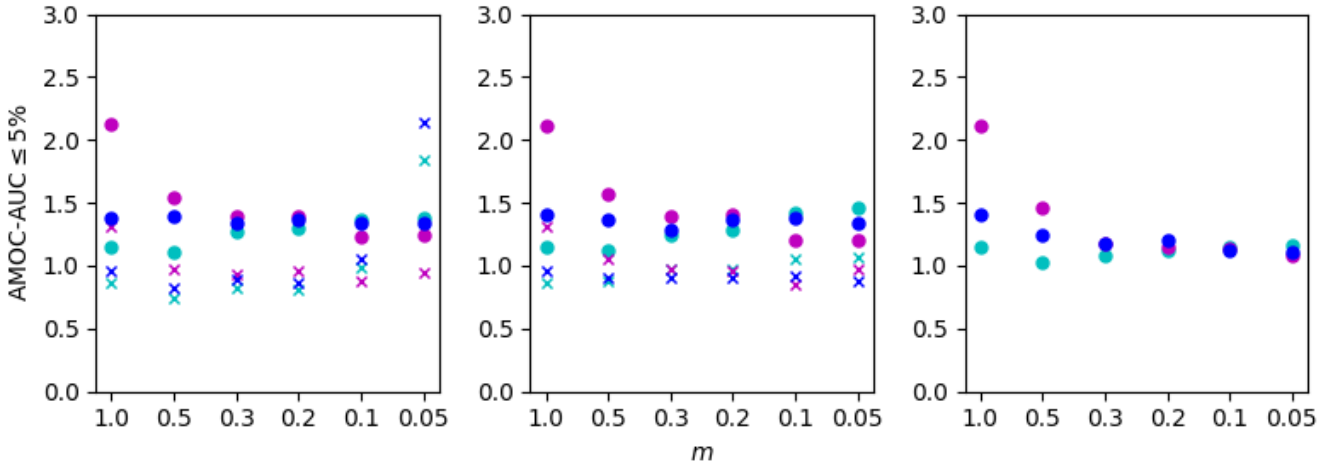

Figure 5.2.: Comparison of evaluation strategies based on AMOC-AUC5
All plots show the average detection delay in days for an false alarm rate less than 0.05. SPNs with Gaussian leafs are colored in cyan, Poisson leafs in magenta and Negative Binomial leafs in blue. Evaluation of marginalized SPNs is depicted with circles and of conditioned SPNs with X's.
Left: Results of Sum-Min Networks from the experiments with Tippett's method.
Mid: Results of count-only Sum-Fisher Networks.
Right: Results of Sum-Fisher Networks with environmental variables.

In figure 5.2, the results of all presented experiments (except the first from 5.3) are illustrated. It is positive to note that each of the displayed experiments accomplish detection delays lower than 2.2 days for at most 5% false alarms. In case of the experiment in SPNs with environmental information, starting from mid-size SPNs, each network can detect the simulated disease outbreaks even in less than 1.5 days. Large SPNs with Gaussian and Negative Binomial leafs using Tippett's method perform worse than mid-size models. This may be due to the estimation of degenerated distributions from clusters only containing few instances.
All other experiments with count-only SPNs with Gaussian and Negative Binomial leafs result in similar values for the same respective evaluation strategies, even for the baselines. While these are among the best results, they are not proportional to the size of the learnt SPNs and therefore not proportional to their goodness of fit. In contrast, SPNs with Poisson leafs show decreasing trends depending on the size-regulating parameter m

and seemingly lead to robust models of different sizes. They show clear trends for all experiments. Hence, I conclude that mixtures and factorizations of Poisson distributions like in SPNs yield a good approximation for the many syndrome counts generated from patient records in emergency rooms and maybe could be applied in the real world. The other two types need further investigation regarding the behavior for large SPNs and long-term performance.
Lastly, Sum-Product Networks containing environmental variables show decreasing detection delays with growing network sizes. This holds for all three types of evaluated SPNs and indicate that enriching health data with environmental information can lead to more robust and possibly more performant syndromic surveillance systems.

6. Conclusion

This thesis introduced a novel method for detecting disease outbreaks using unspecific syndromic surveillance, a type of activity monitoring that aims to find any anomalous patterns in health data with respect to an outbreak-free data-baseline. For this purpose, electronic patient records from an emergency room were used to count the daily occurrences of possible disease patterns, called syndromes. The preprocessed data was used to learn Sum-Product Networks, which are capable of estimating context-sensitive hierarchical models of a joint distribution over different types of random variables and computing the probabilities of evidence, marginals and conditionals exact and computationally efficient. However, the standard inference routines of SPNs are not sufficient for the type of anomaly detection that is required to detect suspicious high counts in a set of observed syndromes. Hypotheses tests were applied to compute the p-values of all syndromes of size 1 and 2, that resemble how likely values as extreme as the observed ones are. Statistical methods for combining p-values between tests of statistically independent variables and in mixture models were evaluated. By replacing product nodes with appropriate procedures for combining independent tests, the p-values of all syndromes can be computed. The minimum p-value of all syndrome counts of a test instance is selected and test datasets can be used to calibrate the alarm threshold of the system with respect to a desired false alarm rate. Experiments on simulated emergency room datasets accomplished average detection delays between 2.2 and 0.9 days for a false alarm rate of up to 5%. When environmental information is incorporated into the learnt models, the average detection delay of SPNs with Gaussian, Poisson, or Negative Binomial distributions robustly approaches 1.1 days with growing size of the networks.

The original contribution of this thesis is the development of applying one-sided hypotheses tests in Sum-Product Networks with the goal of detecting disease outbreaks in emergency room data as early as possible. In theory, this method could also be applied for systems that generally monitor activities in streams of data and attempt to detect changes, or other anomaly detection problems.

The experiments showed promising results on simulated emergency room data, but assessment on real public health data remains for future work. The main challenge of developing syndromic surveillance systems is the lack of data describing outbreak-free distributions and outbreaks with known onset, whose preparation requires the support of medical experts. While Sum-Product Networks can compute some probabilistic queries exact, the models itself are approximate ones. Replacing the applied structure learning algorithms with more sophisticated approaches like Trapp's bayesian learning [43] could possibly lead to shorter detection delays. Lastly, the presented method evaluates test instances daily with no knowledge about the condition of previous days, and the estimated structure is not updated. Considering the suspicious syndrome counts of previous days and updating the data-baseline with evaluated instances could improve the robustness and performance of the presented syndromic surveillance system.

Bibliography

[1] Nathan Bollig et al. “Machine learning for syndromic surveillance using veterinary necropsy reports”. In: *Public Library of Science San Francisco ONE* 15.2 (2020).

[2] David L Buckeridge et al. “Algorithms for rapid outbreak detection: a research synthesis”. In: *Journal of biomedical informatics* 38.2 (2005), pp. 99–113.

[3] Centers for Disease Control and Prevention. “Biological and Chemical Terrorism: Strategic Plan for Preparedness and Response. Recommendations of the CDC Strategic Planning Workgroup”. In: *Morbidity and Mortality Weekly Report* 49 (RR-4 2000).

[4] Centers for Disease Control and Prevention. “Bioterrorism-related inhalational anthrax: the first 10 cases reported in the United States”. In: *Emerging Infectious Diseases* 7 (6 2001), pp. 933–944.

[5] Centers for Disease Control and Prevention. “Preventing emerging infectious diseases: a strategy for the 21st century. Overview of the updated CDC plan”. In: *Morbidity and Mortality Weekly Report* 47 (RR-15 1998).

[6] Hadi Fanaee-T and Joao Gama. “Eigenevent: an algorithm for event detection from complex data streams in syndromic surveillance”. In: *Intelligent Data Analysis* 19.3 (2015), pp. 597–616.

[7] Tom Fawcett and Foster Provost. “Activity monitoring: Noticing interesting changes in behavior”. In: *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*. 1999, pp. 53–62.

[8] Ronald A Fisher. *Statistical Methods for Research Workers*. 1925, pp. 103–111.

[9] Robert Gens and Pedro Domingos. “Learning the structure of sum-product networks”. In: *International Conference on Machine Learning*. 2013, pp. 873–880.

[10] Per H Gesteland et al. “Automated syndromic surveillance for the 2002 Winter Olympics”. In: *Journal of the American Medical Informatics Association* 10.6 (2003), pp. 547–554.

[11] Jeremy Ginsberg et al. “Detecting influenza epidemics using search engine query data”. In: *Nature* 457.7232 (2009), pp. 1012–1014.

[12] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. `http://www.deeplearningbook.org`. MIT Press, 2016.

[13] Fabian Hammes. “Vorhersagen von räumlich korrelierten epidemiologischen Zeitreihen mittels Methoden der Statistik und des maschinellen Lernens”. Master’s thesis. TU Darmstadt, Knowledge Engineering Group, 2019. URL: `https://www.ke.tu-darmstadt.de/lehre/arbeiten/master/2019/Hammes_Fabian.pdf`.

[14] Peter Hartmann. *Mathematik für Informatiker*. 5th. Vieweg+Teubner, 2012, pp. 420–422, 437–442, 445–451.

[15] Robert Heimbach. “Investigation of the Effect of Meteorological Data on Spatial Surveillance of Disease Outbreaks”. Bachelor’s thesis. TU Darmstadt, Knowledge Engineering Group, 2019. URL: `http://www.ke.tu-darmstadt.de/lehre/arbeiten/bachelor/2019/Heimbach_Robert.pdf`.

[16] Kelly J Henning. “What is syndromic surveillance?” In: *Morbidity and Mortality Weekly Report* 53 (Supplement 2004), pp. 7–11.

[17] Joseph M Hilbe. *Modeling Count Data*. 2014, pp. 35–38, 128–129.

[18] Remi Jedwab, Noel D Johnson, and Mark Koyama. “The Economic Impact of the Black Death”. In: *Journal of Economic Literature* Forthcoming (2020).

[19] Zalman Kaufman et al. “Using data on an influenza b outbreak to evaluate a syndromic surveillance system-israel, June 2004”. In: *MMWR (CDC)* 54 (2005), p. 191.

[20] Moritz Kulessa, Eneldo Loza Mencía, and Johannes Fürnkranz. “Improving Outbreak Detection with Stacking of Statistical Surveillance Methods”. In: *Workshop Proceedings of epiDAMIK: Epidemiology meets Data Mining and Knowledge discovery (held in conjunction with ACM SIGKDD 2019)*. 2019.

[21] Moritz Kulessa, Eneldo Loza Mencía, and Johannes Fürnkranz. “Improving the Fusion of Outbreak Detection Methods with Supervised Learning”. In: *16th International Conference on Computational Intelligence methods for Bioinformatics and Biostatistics*. 2019.

[22] John M Last. *A Dictionary of Epidemiology*. 4th. Oxford University Press, 2001, p. 61.

[23] David Lopez-Paz, Philipp Hennig, and Bernhard Schölkopf. “The randomized dependence coefficient”. In: *arXiv preprint arXiv:1304.7717* (2013).

[24] Colin McEvedy. “The Bubonic Plague”. In: *Scientific American* 258 (2 1988), pp. 118–123.

[25] Tom Mitchell. *Machine Learning*. McGraw Hill, 1997.

[26] Alejandro Molina, Sriraam Natarajan, and Kristian Kersting. “Poisson sum-product networks: A deep architecture for tractable multivariate poisson distributions”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 31. 1. 2017.

[27] Alejandro Molina et al. “Mixed sum-product networks: A deep architecture for hybrid domains”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 32. 1. 2018.

[28] Alejandro Molina et al. *SPFlow: An easy and extensible library for deep probabilistic learning using sum-product networks*. URL: https://arxiv.org/abs/1901.03704. (accessed: 07.01.2021).

[29] Todd K Moon. “The expectation-maximization algorithm”. In: *IEEE Signal processing magazine* 13.6 (1996), pp. 47–60.

[30] David M Morens and Robert J Littman. “Epidemiology of the plague of Athens”. In: *Transactions of the American Philological Association* 122 (1992), pp. 271–304.

[31] World Health Organization. *COVID-19 Weekly Epidemiological Update - 05.01.2021*. URL: https://www.who.int/docs/default-source/coronaviruse/situation-reports/20210105_weekly_epi_update_21.pdf. (accessed: 07.01.2021).

[32] World Health Organization. *Novel Coronavirus(2019-nCoV) Situation Report - 31.01.2020*. URL: https://www.who.int/docs/default-source/coronaviruse/situation-reports/20200131-sitrep-11-ncov.pdf. (accessed: 07.01.2021).

[33] World Health Organization. *Pneumonia of unknown cause in China - 05.01.2020*. URL: https://www.who.int/csr/don/05-january-2020-pneumonia-of-unkown-cause-china/en/. (accessed: 07.01.2021).

[34] Iago París, Raquel Sánchez-Cauce, and Francisco J Díez. *Sum-product networks: A survey*. 2020. arXiv: 2004.01167 [cs.LG].

[35] Robert Peharz et al. “On theoretical properties of sum-product networks”. In: *Artificial Intelligence and Statistics*. 2015, pp. 744–752.

[36] Marc Pfetsch. *Vorlesungsskript Mathematik III für Informatik*. 2018. URL: http://www2.mathematik.tu-darmstadt.de/~pfetsch/teaching.de.html. [not freely available].

[37] Hoifung Poon and Pedro Domingos. "Sum-product networks: A new deep architecture". In: *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*. 2011, pp. 689–690. DOI: 10.1109/ICCVW.2011.6130310.

[38] Ben Y Reis and Kenneth D Mandl. "Time series modeling for syndromic surveillance". In: *BMC Medical Informatics and Decision Making* 3.1 (2003), pp. 1–11.

[39] Hussin A Rothan and Siddappa N Byrareddy. "The epidemiology and pathogenesis of coronavirus disease (COVID-19) outbreak". In: *Journal of Autoimmunity* (109 2020), p. 102433.

[40] Marc Schneider. "Linking of emergency room and infectious disease data using machine learning approaches". Master's thesis. TU Darmstadt, Knowledge Engineering Group, 2019.

[41] European Parliamentary Research Service. *Economic impact of epidemics and pandemics - Briefing*. URL: https://www.europarl.europa.eu/RegData/etudes/BRIE/2020/646195/EPRS_BRI(2020)646195_EN.pdf. (accessed: 07.01.2021).

[42] Leonard H C Tippett. *The methods of statistics*. 1931.

[43] Martin Trapp et al. "Bayesian learning of sum-product networks". In: *Advances in Neural Information Processing Systems*. 2019, pp. 6347–6358.

[44] Geoffrey I Webb et al. "Characterizing concept drift". In: *Data Mining and Knowledge Discovery* 30.4 (2016), pp. 964–994.

[45] Weng-Keen Wong et al. "Bayesian network anomaly pattern detection for disease outbreaks". In: *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*. 2003, pp. 808–815.

[46] Weng-Keen Wong et al. "Rule-based anomaly pattern detection for detecting disease outbreaks". In: *AAAI/IAAI*. 2002, pp. 217–223.

[47] Worldbank. *China's trade balance, exports and imports by country and region, 2018*. URL: https://wits.worldbank.org/CountryProfile/en/Country/CHN/Year/LTST/TradeFlow/EXPIMP. (accessed: 07.01.2021).

A. Appendix

Probability Space

Definition A.0.1 (Sample space, event)**.** Ω is called a sample space and is the set of *all possible* outcomes ω. A subset $A \subseteq \Omega$ is called an event and occurs, if an outcome $\omega \in A$ is observed.

Definition A.0.2 (Event space, σ-algebra)**.** If Ω is a countable set, the power-set $\mathcal{P}(\Omega)$ is called an event space. A subset $\mathcal{A} \subseteq \mathcal{P}(\Omega)$ is called σ-algebra, if the following hold:

1. $\Omega \in \mathcal{A}$,
2. If $A \in \mathcal{A}$, then $\tilde{A} \in \mathcal{A}$,
3. For every series $A_1, A_2, ..., \in \mathcal{A}$ holds $\bigcup_{i=1}^{\infty} A_i \in \mathcal{A}$.

Definition A.0.3 (Probability measure)**.** A mapping $p : \mathcal{A} \to \mathbb{R}$ is called probability measure, if it satisfies the Kolmogorov axioms:

1. $p(A) \geq 0$ for $A \in \mathcal{A}$,
2. $p(\Omega) = 1$,
3. $p(\bigcup_{i=1}^{\infty} A_i) = \sum_{i=1}^{\infty} p(A_i)$ for pair-wise disjoint $A_1, A_2, ... \in \mathcal{A}$.

In other words, a well-defined probability measure assigns each event a probability between 0 and 1, and the probabilities of all events sum up to 1. The probability, that pair-wise disjoint events occur, is the sum of the respective probabilities. Definition A.0.3, condition 3., is also valid for finite, disjoint unions $\bigcup_{i=1}^{n} A_i$ by setting $A_i = \emptyset$ for $i \geq n + 1$:

$$p(A_1 \cup ... \cup A_n) = \sum_{i=1}^{n} p(A_i), \text{ if } A_1, ..., A_n \text{ pair-wise disjoint} \tag{A.1}$$

With the three previous definitions, we can now construct a well-defined probability space:

Definition A.0.4 (Probability space)**.** A probability space is a triplet (Ω, Σ, p) consisting of a sample space Ω, a σ-algebra Σ and a probability measure p.

When we deal with countable Ω, most often, the whole power-set $\mathcal{P}(\Omega)$ is defined as the σ-algebra (which is the largest possible σ-algebra of Ω) and we simply denote the probability space as a sample space and its associated probability measure (Ω, p). In some literature, a probability space is also called statistical model.