UNIT 6 U6/0/2024

Machine Learning

COS4852

Year module

Department of Computer Science

School of Computing

CONTENTS

This document contains the material for UNIT 6 for COS4852 for 2024.

1 OUTCOMES

In this Unit you will learn about the Bayesian approach to machine learning. Specifically you will study:

1. The mechanics of Bayes' Theorem.
2. How to build a model of related events using Bayes' theorem.
3. A theoretically ideal probabilistic model, the Bayes Optimal classifier.
4. A more practical model, the Naive Bayes Classifier.
5. How to depict the probabilities of related events using a Bayesian Belief Network.

After completion of this Unit you will be able to:

1. Understand and describe Bayes' Theorem.
2. Solve a given learning task using the Bayes Optimal Classifier.
3. Solve a given learning task using the Naive Bayes Classifier.

2 PREPARATION

2.1 Textbooks

Chapter 6 of Tom Mitchell's book discusses Bayes' Theorem in relation to Concept Learning, and goes on to the Bayes Optimal Classifier (a theoretical model of the ideal probabilities classifier), then on the Naive Bayes Classifier (which can be used in the real world) and also Bayesian Belief Networks, which is a useful way to visually depict the probabilities involved in event that are related to each other.

2.2 Online Material

Here is an excellent lecture on Bayes' Theorem, with several practical examples, worked through in detail. There are no mathematics in this video, just simple explanations using diagrams and highlighting the important things to keep in mind when thinking Bayesian. This should help you grasp the concepts behind Bayes' Theorem.

This page on BetterExplained gives a short, intuitive explanation on Bayes' Theorem using cancer tests and spam detection as examples.

This TDS article is an excellent, mathematically based discussion on Bayes' Theorem using a COVID test and its results as the example.

Allen Kim gives an interactive visual description to help you understand what happens to the posterior probabilities (the model) when the prior probabilities are changed in Bayes' Theorem.

This YouTube video gives a detailed and step-by-step work-through of the Naive Bayes Classifier.

3 INTRODUCTION

Thomas Bayes was a Presbyterian minister in the mid 1700's, and also a lay statistician, as so many scientists were in those days. He is famous for the theorem named after him. He worked on the problem of assigning a probability to an event (called an unobserved variable) of which we only have information on related events. His observation was that if you have some initial conditional probabilities (called beliefs in inferential statistics), and you then get new objective data on related events you can improve your initial probabilities (improved beliefs). He used his theorem to investigate why it seems that slightly more boys than girls are born (the actual ratio is about 106 boys born for every 100 girls born – 51.5%:48.5%). He spent decades collecting demographic data from around the world (this was the 1700s, and there was no Google), and using his theorem, concluded that this ratio is the same everywhere and determined by biology.

Bayes' Theorem also plays a central role in many machine learning algorithms. In most real-world problems involving large volumes of data, the models being built will give you a probability as output, or a set of classifications with probabilities assigned to them indicating the likelihood of each classification being correct.

3.1 Bayes' Theorem

Let's work through an example to see Bayes' Theorem in action.

Here are some more online material that will help you in the following discussion:

- A page on sensitivity and specificity.
- https://byjus.com/maths/bayes-theorem/
- https://betterexplained.com/articles/an-intuitive-and-short-explanation-of-bayes-theorem/
- https://en.wikipedia.org/wiki/Sensitivity_and_specificity
- https://microbenotes.com/sensitivity-specificity-false-positive-false-negative/

In the world we are experiencing now, people have become a lot more aware of medical test results. One question that most people would want to have answered: *How accurate is a test for an infection?*

There are terms often used in medical test results, and are vital in correctly interpreting a test result:

1. **Prevalence**: the ratio of the total population who is infected.
2. **True positive (TP)**: the ratio of the total tests that are accurately labelled as positive. This is dependent on the prevalence.
3. **True negative (TN)**: the ratio of the total tests that are accurately labelled as negative. This is dependent on the prevalence.
4. **Sensitivity**: (true positive rate TPR) the ratio of positive tests that are accurately labelled as positive. This is independent of the prevalence.
5. **Specificity**: (true negative rate TNR) the ratio of negative tests that are accurately labelled as negative. This is independent of the prevalence.
6. **Positive Predictive Value (PPV)**: the probability of an infection given a positive test result.
7. **Negative Predictive Value (NPV)**: the probability of no infection given a negative test result.

Consider the data on a medical test for SUPERBUG:

1. Out of every 10 000 people with a record of possible symptoms, more or less 100 people were diagnosed with SUPERBUG. These are confirmed cases, based on a combination of doctors' diagnoses, CT-scans, several different tests, and postmortem analyses.
2. It is known that for this test, 10 out of 100 positive test results are incorrect. This is the inverse *sensitivity* of the test.
3. It is known that for this test, 10 out of 50 negative test results are incorrect. This is the inverse *specificity* of the test.

You have just been tested for SUPERBUG, but are still waiting for your results. Obviously you will want to know what a positive or negative result will tell you about the likelihood that you have been infected with SUPERBUG, so that you can decide how to deal with the result.

Define the variables. Let:

$$
\begin{array}{rcl}
\text{Bug} & \leftarrow & \text{a person is infected with SUPERBUG} \\
\neg\text{Bug} & \leftarrow & \text{a person is not infected with SUPERBUG} \\
\text{Pos} & \leftarrow & \text{a positive test result} \\
\text{Neg} & \leftarrow & \text{a negative test result} \\
 & = & \neg\text{Pos}
\end{array}
$$

The actual prevalence (base rate) is unknown. We can only know that number if everybody is tested, with a test that is 100% accurate, and everybody is tested in a very short space of time, or everybody get retested at short intervals. There is no test that is 100% accurate and mass testing is very expensive and complicated. We can therefore only use the frequency of the number of confirmed positive cases against the population size. This will be our best estimate for prevalence. Since we were given a population of 10 000, we will use this number as a convenient population sample to calculate the ratios.

$$\begin{aligned} P(\text{Bug}) &\leftarrow \text{prevalence} \\ &\approx {}^{100}/_{10\,000} \\ &= 0.01 \end{aligned}$$

and therefore:

$$\begin{aligned} P(\neg\text{Bug}) &\leftarrow \text{inverse prevalence} \\ &= 1 - 0.01 \\ &= 0.99 \\ &= {}^{9\,900}/_{10\,000} \end{aligned}$$

The known *sensitivity* and *specificity* tells us that:

$$\begin{aligned} P(\text{Pos}|\text{Bug}) &\leftarrow \text{sensitivity} \\ &= {}^{(100-10)}/_{100} \\ &= 0.9 \\ &= {}^{90}/_{100} \\ P(\text{Neg}|\neg\text{Bug}) &\leftarrow \text{specificity} \\ &= {}^{(50-10)}/_{50} \\ &= 0.8 \\ &= {}^{7\,920}/_{9\,900} \end{aligned}$$

We can also write down their inverses:

$$\begin{aligned} P(\text{Neg}|\text{Bug}) &\leftarrow 1 - P(\text{Pos}|\text{Bug}) \\ &= 1 - 0.9 \\ &= 0.1 \\ &= {}^{10}/_{100} \\ P(\text{Pos}|\neg\text{Bug}) &\leftarrow 1 - P(\text{Neg}|\neg\text{Bug}) \\ &= 1 - 0.8 \\ &= 0.2 \\ &= {}^{1\,980}/_{9\,900} \end{aligned}$$

Therefore, of the 100 infected people (out of the population of 10 000), 90 get positive results, and 10 get negative results. Of the 9 900 people not infected (out of the population of 10 000),

$0.8 \times 9\,900 = 7\,920$ get negative results, and $9\,900 - 7\,920 = 1980$ get positive results. Therefore $90 + 1\,980 = 2\,070$ came back positive, and $7\,920 + 10 = 7\,930$ came back negative.

We can also now write down the following probabilities:

$$
\begin{aligned}
\text{True positive} &\leftarrow \text{ratio of all tests correctly labelled as positive} \\
&= {}^{90}/_{10\,000} \\
&= 0.009 \\
\text{True negative} &\leftarrow \text{ratio of all tests correctly labelled as negative} \\
&= {}^{7\,920}/_{10\,000} \\
&= 0.792 \\
\text{False positive} &\leftarrow \text{ratio of all tests incorrectly labelled as positive} \\
&= {}^{1\,980}/_{10\,000} \\
&= 0.198 \\
\text{False negative} &\leftarrow \text{ratio of all tests incorrectly labelled as negative} \\
&= {}^{10}/_{10\,000} \\
&= 0.001
\end{aligned}
$$

We can represent these values in table form, as shown in Table 1 and Table 2. Figure 1 shows this visually, but the small ratios make this not so easy to see. Figure 2 shows another set of ratios, where the prevalence becomes 25%, which gives a better idea. As an exercise, repeat the calculations for a prevalence of 25%, to see if you can get the same numbers as in the figure.

	Bug	¬Bug	totals
Pos	90	1980	2070
Neg	10	7920	7930
totals	100	9900	10 000

Table 1: Ratios (out of 10 000) for SUPERBUG and it's test.

	Bug	¬Bug	totals
Pos	0.009	0.198	0.207
Neg	0.001	0.792	0.793
totals	0.010	0.990	1.000

Table 2: probabilities for SUPERBUG and it's test.

What we cannot immediately calculate, are the following:

$$
\begin{aligned}
P(\text{Bug}|\text{Pos}) &\leftarrow \text{chance of having the bug given a positive test} \\
P(\text{Bug}|\text{Neg}) &\leftarrow \text{chance of having the bug given a negative test} \\
P(\neg\text{Bug}|\text{Pos}) &\leftarrow \text{chance of not having the bug given a positive test} \\
P(\neg\text{Bug}|\text{Neg}) &\leftarrow \text{chance of not having the bug given a negative test}
\end{aligned}
$$

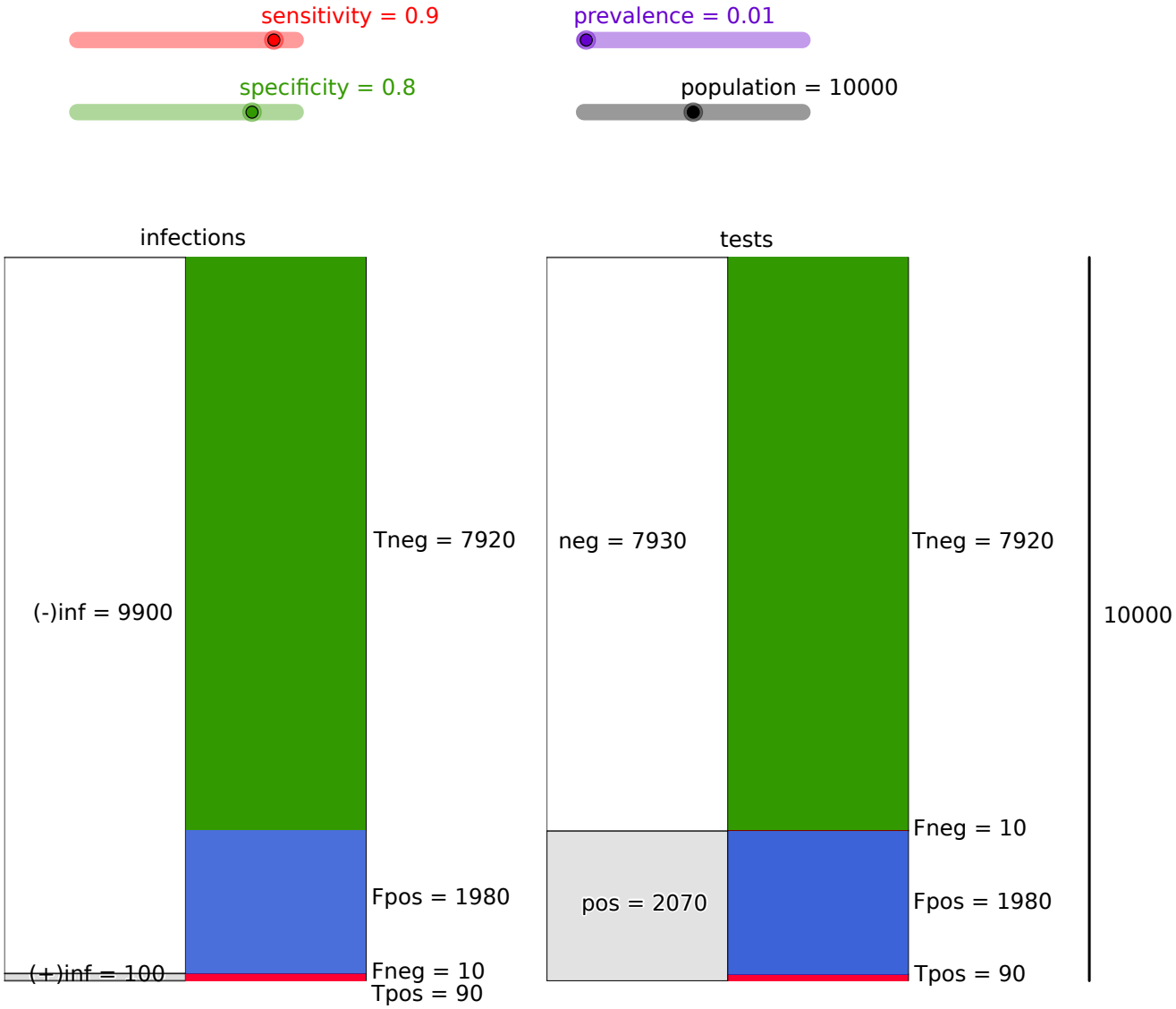

Figure 1: Bug test ratios shown visually for a prevalence of 0.01.

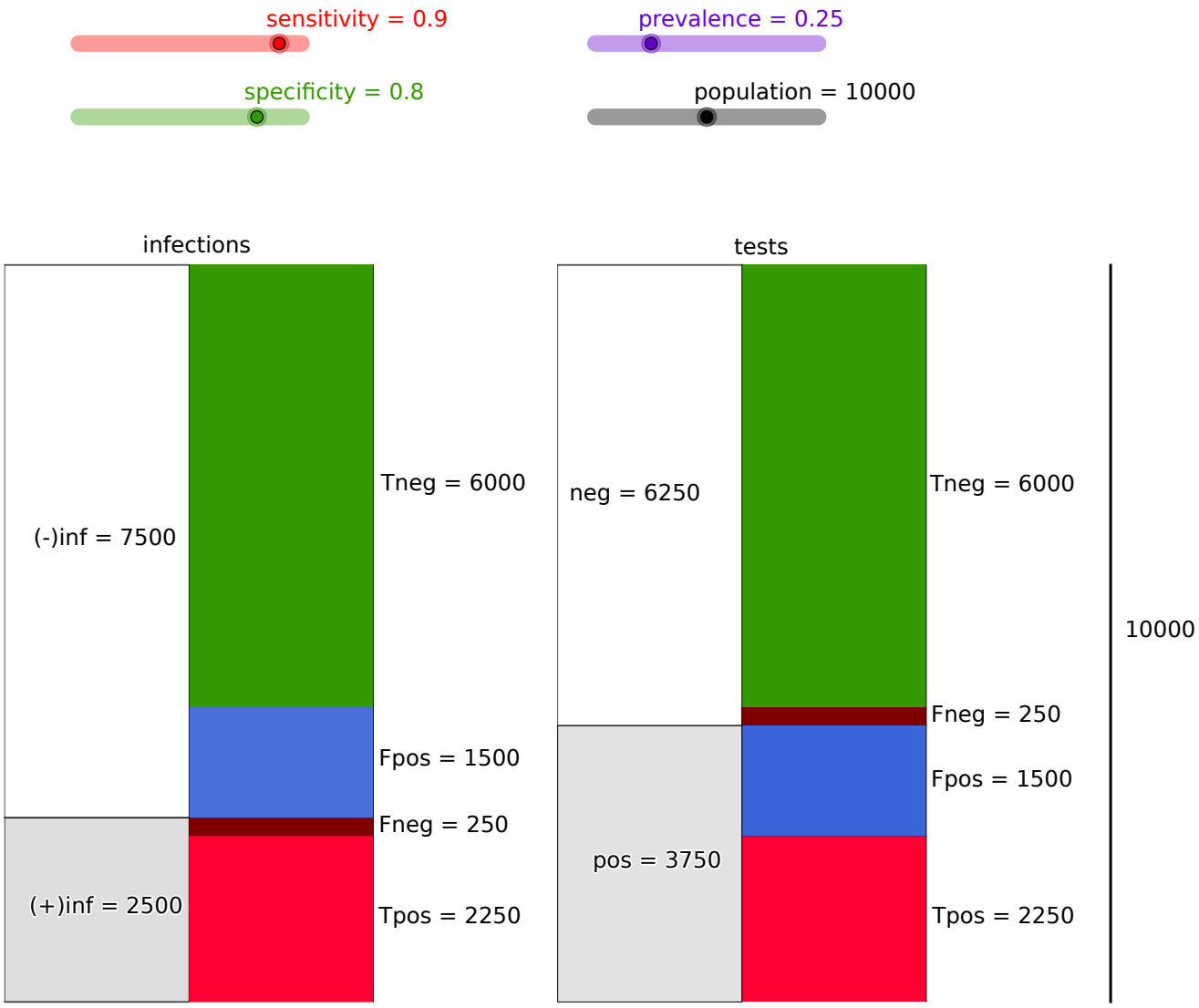

Figure 2: Bug test ratios shown visually for a prevalence of 0.25.

Bayes' rule can be used to calculate these probabilities, and is expressed using the prior (known) probabilities $P(A)$, $P(B)$, and the probability of B given that A is true, $P(B|A)$. Bayes' theorem gives us the probability of A given B, $P(A|B)$ as:

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

where

$$P(B) = P(A)P(B|A) + P(\neg A)P(B|\neg A)$$

To summarise the known probabilities:

$$\begin{aligned} P(\text{Bug}) &= 0.01 \\ P(\neg\text{Bug}) &= 0.99 \\ P(\text{Pos}|\text{Bug}) &= 0.90 \\ P(\text{Neg}|\text{Bug}) &= 0.10 \\ P(\text{Neg}|\neg\text{Bug}) &= 0.80 \\ P(\text{Pos}|\neg\text{Bug}) &= 0.20 \end{aligned}$$

To test the posterior probability of having SUPERBUG, given that the test results came back positive, let:

$$\begin{aligned} A &= \text{Bug} \\ B &= \text{Pos} \end{aligned}$$

Plugging these into Bayes' theorem, we get:

$$\begin{aligned} P(B) &= P(A)P(B|A) + P(\neg A)P(B|\neg A) \\ P(\text{Pos}) &= P(\text{Bug})P(\text{Pos}|\text{Bug}) + P(\neg\text{Bug})P(\text{Pos}|\neg\text{Bug}) \\ &= (0.01 \times 0.90) + (0.99 \times 0.20) \\ &= 0.009 + 0.198 \\ &= 0.207 \end{aligned}$$

$$\begin{aligned} P(\text{A}|\text{B}) &= \frac{P(\text{A}) \times P(\text{B}|\text{A})}{P(\text{B})} \\ P(\text{Bug}|\text{Pos}) &= \frac{P(\text{Bug}) \times P(\text{Pos}|\text{Bug})}{P(\text{Pos})} \\ &= \frac{0.01 \times 0.90}{0.207} \\ &= \frac{0.009}{0.207} \\ &= 0.04348 \end{aligned}$$

To test the posterior probability of *not* having SUPERBUG, given that the test results came back negative, let:

$$\begin{aligned} A &= \neg\text{Bug} \\ B &= \text{Neg} \end{aligned}$$

$$\begin{aligned} P(B) &= P(A)P(B|A) + P(\neg A)P(B|\neg A) \\ P(\text{Neg}) &= P(\neg\text{Bug})P(\text{Neg}|\neg\text{Bug}) + P(\text{Bug})P(\text{Neg}|\text{Bug}) \\ &= (0.99 \times 0.80) + (0.01 \times 0.10) \\ &= 0.792 + 0.001 \\ &= 0.793 \end{aligned}$$

$$\begin{aligned} P(\text{A}|\text{B}) &= \frac{P(\text{A}) \times P(\text{B}|\text{A})}{P(\text{B})} \\ P(\text{Bug}|\text{Neg}) &= \frac{P(\neg\text{Bug}) \times P(\text{Neg}|\neg\text{Bug})}{P(\text{Neg})} \\ &= \frac{0.99 \times 0.80}{0.000792} \\ &= \frac{0.792}{0.793} \\ &= 0.9987 \end{aligned}$$

This means that a positive test results indicated that you only have a 4.3% likelihood of being infected with SUPERBUG. The reason for this very low number is related to the low figure we have for the prevalence of the infection in the population. This means that most people who are testes, and gets a positive result have not been infected. In a pandemic this may be acceptable, since you want to prevent the spread of the infection. In a situation where the majority of people are negative, mass testing will not be helpful. When the testing regime is changed to only test people with a high likelihood of being positive (such as having known symptoms, or having been exposed to a known person with the disease), the prevalence in the tested population will go up, and the PPV will also increase.

Something else to consider is that the prevalence is related to the population being *tested*. If the tests are done randomly, you would see the sort of figures shown here. However, in reality people are only tested when there is a valid suspicion that they may be positive, such as having most of the related symptoms, or if they have been exposed to a known positive person. In such a population, the prevalence will go up significantly, with a 25% prevalence being about right. Even in such a case, there are still very many false positives, but the false negatives are very low.

Similarly, we can calculate the posterior probability of not having SUPERBUG, given a negative test result:

$$\begin{aligned} A &= \neg\text{Bug} \\ B &= \text{Neg} \end{aligned}$$

Bayes' theorem therefore tells us:

$$\begin{aligned} P(\neg\text{Bug}|\text{Neg}) &= \frac{P(\neg\text{Bug}) \times P(\text{Neg}|\neg\text{Bug}))}{P(\text{Neg})} \\ &= \frac{0.9891 \times 0.98}{0.9694} \\ &= 0.9999 \end{aligned}$$

which means that a negative test results indicates that you have an almost 100% likelihood of not being infected with SUPERBUG.

Keep in mind that the numbers used here are purely hypothetical, and does not reflect any real figures. False positive and false negative rates are as problematic in a new test as the prevalence numbers would be. In the real world, you don't look at the test results in isolation. First, there is the uncertainty of the actual frequency of the population who has the infection. When looking at something like cancer tests, where their is a long history of medical data to work with, the figures about prevalence is very accurate, and becomes more so as more data is collected. At the start of a pandemic there simply is not enough data to get the prevalence accurately enough, and estimates are used, with various models developed specifically for this.

The timing of the test for a viral infection has an important part to play in the accuracy of test results. Viral tests, for example, are dependent on the patient having enough viral particles for the test to extract sufficient material for the test, and a day makes a huge difference in the viral load. Further, RNA tests are highly sensitive, due to the RNA being unique for every entity. Such a test usually has a very low false positive rate, but the false negative could be very high due to timing, mishandling of the sample, lab errors, and so on.

The prevalence would also change as the outbreak progresses, but will get more accurate in time. In the case of some outbreaks, there are people who are positive, but does not experience any symptoms. Most of these people will not get diagnosed, nor tested. This would therefore mean that the prevalence rate is underestimated. Antibody tests may indicate whether somebody have had the infection in the past, which could then be added to the prevalence numbers. Again, such tests will not be done on everybody, for practical and economic reasons, and people may lose their immunity in time. In short, the prevalence rate is at bet an estimate.

In a pandemic situation, where the bulk of the population does not have the infection, we find that the probability of a positive test being correct is much lower than a negative test being correct. Here is an excellent page that explains why it is important to use a test that has high specificity.

3.2 Bayesian Optimal Classifier

Bayes Optimal Classifier is a technique that will maximise the probability that a new instance is classified correctly, if the same data, hypothesis space, and prior probabilities over the hypothesis space are used. It will effectively outperform any other classification method in terms of accuracy. The model is described as a classification technique, but the same principles can be applied to a regression task.

In practice, however, this model is difficult to use, as the computational cost is too high. It serves as the basis for other, more practical models, such as the Naive Bayes Classifier.

Bayes Theorem provides a principled way for calculating conditional probabilities, called a posterior probabilities. This is used in the Maximum a Posteriori (MAP) framework that finds the most probable hypothesis that describes the training data-set. Bayes Optimal Classifier is a probabilistic model that finds this most probable prediction for a new instance, using the training data and its hypothesis space.

Lets' start with the question:

> *What is the most probable hypothesis, given the training data?*

There are two statistical approaches to answering this question:

- Maximum a Posteriori (MAP)
- Maximum Likelihood Estimator (MLE)

MAP follows a Bayesian approach, while MLE tackles the problem from a frequentist angle. Both fits an optimisation model to the data, and classifies a new instance by searching for the most probable distribution and set of parameters that describes the training data.

This blogpost on Medium shows you how to calculate MLE and MAP for a data-set using Bayes' Theorem.

We can use Bayes' Theorem to estimate the proportional hypothesis and parameter (θ) that explains the data-set X. This can be written as:

$$P(\theta|X) = P(X|\theta) \cdot P(\theta)$$

By maximising this probability over a range of θ values, we can estimate the central tendency of the posterior probability, i.e. build a model of the distribution. This technique of maximising the posterior probability, is called the *maximum a posteriori* estimation, or MAP for short.

The MAP technique tells us which is the most probable hypothesis for a new instance given the training data-set. The real question we want to ask is: *What is the most probable classification of the new instance given the training data-set?*. At first it may seem as we simply apply MAP to the new instance, it is possible to do better.

Consider a hypothesis space consisting of 3 hypotheses, h_1, h_2, and h_3. The posterior probabilities of these hypotheses given the data-set are 0.4, 0.3, and 0.3, respectively (note that they add up to a total probability of 1.0). Since the posterior probability of h_1 is the highest (maximum), this means the h_1 is the MAP hypotheses for this data-set.

A new instance x is classified as positive ($\oplus$) by h_1 and negative ($\ominus$) by both h_2 and h_3. If we consider the posterior probabilities, the most probable classification is $\ominus$ since their combined probability is 0.6, while h_1 only has a 0.4 probability for x being $\oplus$. The most probable classification is different from the MAP result.

We can generalise this. To get the most probable classification of a new instance we can combine the classification of all hypotheses and weight them by their posterior probabilities. Therefore, if the new instance can be classified as any value v_j from a set V (only $\oplus$ and $\ominus$ in the above example), the conditional probability $P(v_j|D)$ (D is the data-set) that the correct classification for the new instance is v_j is:

$$P(v_j|D) = \sum_{h_i \in H} P(v_j|h_i) \cdot P(h_i|D)$$

The optimal classification of the new instance is the value v_j, for which

$$\arg\max_{v_j \in V} \sum_{h_i \in H} P(v_j|h_i) \cdot P(h_i|D)$$

This is the Bayes' Optimal Classification.

Let's look at our example again. The possible classifications for x is $V = \{\oplus, \ominus\}$, and the conditional probabilities are:

$$\begin{aligned} P(h_1|D) &= 0.4 \\ P(\ominus|h_1) &= 0.0 \\ P(\oplus|h_1) &= 1.0 \end{aligned}$$

$$\begin{aligned} P(h_2|D) &= 0.3 \\ P(\ominus|h_2) &= 1.0 \\ P(\oplus|h_2) &= 0.0 \end{aligned}$$

$$\begin{aligned} P(h_3|D) &= 0.3 \\ P(\ominus|h_3) &= 1.0 \\ P(\oplus|h_3) &= 0.0 \end{aligned}$$

To get the Bayes' Optimal Classification for the instance x, we use these values and do the following:

$$\sum_{h_i \in H} P(\oplus|h_i) \cdot P(h_i|D) = 0.4$$

$$\sum_{h_i \in H} P(\ominus|h_i) \cdot P(h_i|D) = 0.6$$

and

$$\arg\max_{v_j \in \{\oplus,\ominus\}} \sum_{h_i \in H} P(v_j|h_i) \cdot P(h_i|D) = \ominus$$

No other classification method can outperform this one, given the same information. Given that there is uncertainty in the data (we are using probabilities after all), and that we have incomplete information about the domain and hypothesis space, the classifier will make mistakes (the so-called Bayes error, which is the best any model can do). The technique is optimal in the sense that, on average, it will have the lowest error rate on the classifications it makes.

A quick calculation on the above example will show you that this is a very expensive algorithm, since we have to calculate the posterior probability for every hypothesis, and combine these predicted classification, for every new instance. In real-world problems with incomplete and noisy data, we don't even have the complete hypothesis space and we cannot calculate the conditional distribution of the model output over its input.

Thus, the Bayes' Optimal Classifier is the unattainable ideal, and in practice we use less optimal variations of the Bayes' Optimal Classifier, or other algorithms using the same principles.

Two algorithms that does this are:

Gibbs Algorithm This is a simple algorithm, where we randomly pick a hypothesis from H, based on the posterior conditional distribution over H, to predict the classification of x. It does surprisingly well, with at most double the optimal Bayes error.

Naive Bayes Classifier Here we assume that the attributes in the input space are conditionally independent, given the target value, which simplifies the search space significantly. More on this in the next section.

As a comparison, the relatively straightforward k-NEAREST NEIGHBOURS algorithm often comes very close to the optimal Bayes error for certain domains.

3.3 Naive Bayesian classifier

Consider the learning task where each instance x is described by a conjunction of attribute values, and where the target function can take on any value from a finite set V. There is a set of training examples. The learner is tasked with classifying a new instance , described by the tuple $\langle a_1, a_2, ... a_n \rangle$. The Bayesian approach would be to assign the most probable target value, v_{MAP}:

$$v_{MAP} = \arg\max_{v_j \in V} P(v_j|a_1, a_2, ... a_n) \tag{1}$$

Using Bayes' Theorem we can re-write this expression as:

$$v_{MAP} = \arg\max_{v_j \in V} \frac{P(a_1, a_2, ... a_n|v_j)P(v_j)}{P(a_a, a_2, ... v_n)} \tag{2}$$

$$= \arg\max_{v_j \in V} P(a_1, a_2, ... a_n|v_j)P(v_j) \tag{3}$$

We need to estimate the two terms of Equation 3. Getting $P(v_j)$ can be done simply by counting the frequency of v_j in the training data. Estimating $P(a_1, a_2, ... a_n|v_j)$ is far more difficult. Getting a reliable estimate here can only be done if we have a very large data set where we see every instance in the instance space many times. This is not feasible in practice. We therefore need to make some simplifying assumptions. These simplifications are why we use the term *naive*. Here you can immediately see that there may be many possible simplification assumptions. Hence Naive Bayes is really more a family of learners than a single one, where the only real difference is in the assumptions being made.

We can make a simplification by assuming that the attribute values are conditionally independent of the target value - given the target value of a specific instance, the probability of observing the conjunction $a_1, a_2, ... a_n$ is simply the product of the probabilities of the individual attributes:

$$P(a_1, a_2, ... a_n|v_j) = \prod_i P(a_i|v_j) \tag{4}$$

Substituting Equation 4 into Equation 3, we get the Naive Bayes Classifier, namely:

$$v_{NB} = \arg\max_{v_j \in V} P(v_j) \prod_i P(a_i|v_j) \tag{5}$$

Learning in Naive Bayes occurs when we estimate all the $P(v_j)$ and $P(a_i|v_j)$ values, based on their frequencies in the training data. This set of estimates comprise the learned hypothesis. Each new instance is then classified by applying Equation 5.

You can read more on the family of Naive Bayes classifiers in the Scikit-Learn documentation and this page at Datacademia.

3.3.1 An example of using Naive Bayes

The classic example data set here is the *PlayTennis* set, where we try to decide whether to play tennis or not, based on previously observed weather data (*Outlook*, *Temperature*, *Humidity*, *Wind*), and the current values of these weather variables. A CSV file with the data set is on Kaggle.

We have the new instance (today's weather data):

$$\langle Outlook = sunny, Temperature = cool, Humidity = high, Wind = strong \rangle$$

The task is to determine the target value, *PlayTennis* as either *yes* or *no*. Apply the frequencies of the data set to Equation 5:

$$v_{NB} = \arg\max_{v_j \in \{yes,no\}} P(v_j) \prod_i P(a_i|v_j) \tag{6}$$

$$= \arg\max_{v_j \in \{yes,no\}} P(Outlook = sunny|v_j) \cdot P(Temperature = cool|v_j) \cdot P(Humidity = high|v_j) \cdot P(Wind = strong|v_j) \tag{7}$$

The probabilities of the two possible target values can be estimated from the data:

$$P(PlayTennis = yes) = 9/14 = 0.64 \quad (8)$$

$$P(PlayTennis = no) = 5/14 = 0.36 \quad (9)$$

The conditional probabilities can also be calculated from the data:

$$P(Wind = strong|PlayTennis = yes) = 3/9 = 0.33 \quad (10)$$

$$P(Wind = strong|PlayTennis = no) = 3/5 = 0.60 \quad (11)$$

Plug these values into Equation 7 to get:

$$P(yes)P(sunny|yes)P(cool|yes)P(high|yes)P(strong|yes) = 0.0053 \quad (12)$$

$$P(no)P(sunny|no)P(cool|no)P(high|no)P(strong|no) = 0.0206 \quad (13)$$

The new instance is therefore classified as *PlayTennis* = *no*. To get the conditional probability that the target value is *no*, given the observed attribute values, normalise the quantities:

$$P(PlayTennis = no|a_i) = \frac{0.0206}{0.0206 + 0.0053} = 0.795 \quad (14)$$

You see here that probabilistic learners give us a more nuanced answer, instead of just a classification. The most likely classification is given as *no*, with a probability of 0.795 (about 80%) of being correct, assuming our simplification is valid.

To see this example in code, there is a worked Python example on Kaggle.

3.4 Bayesian Belief Networks

A Bayesian Belief Network (BBN) is a directed acyclic graph (DAG) that represent the conditional dependencies between variables related to a specific causal problem. We can use this paper by Bromley to illustrate a BBN.Bromley's paper applies BBNs to a real-world problem, but is also does a very good job of taking you step-by-step through the contruction of a BBN.

A BBN consists of a number of nodes linked to each other with transitions (directed arrows) that captures the causal relationship between variables (sometimes called facts). Bromley's paper discussed how BBNs can be used to facilitate water resource management and decision making. In his paper he shows that the *Annual River Flow* is dependent on the *Percentage Forest Cover* and the *Annual Rainfall*. If either or both of the last two change the *Annual River Flow* will also change. In its turn, the river flow determine how much water will be available for storage in reservoirs, as well as the size of the fish population. The size of the fish population then has an effect on the potential for anglers to catch fish. Similarly, forest cover inversely determines how much land is available for farming purposes. More forest means less farmland. The available farmland in turn determines agricultural production, which then determine farmers' income.

Eugene Charniak wrote an excellent introduction to BBNs that will also take you step-by-step through BBNs, their contruction, and their application.

Jason Brownlee also makes an attempt at a gentle introduction to BBNs. It is worth a read, though the paper by Charniak is the best references of the three given here.

4 ACTIVITIES

4.1 TASK 1 - STUDY THE NOTES

Find all the links referred to in this document, and study these in detail. In some cases more than one reference is given to give you a slightly different perspective or as an alternative.

4.2 TASK 2 - Bias and Variance

Look at the StatQuest YouTube on *Bias* and *Variance* in order to learn about this very important problem in statistics that has a huge effect on what is possible and not possible in Machine Learning models.

4.3 TASK 3 - Covid

This article on the Story of Mathematics (SOM) gives seven worked example of using Bayes' Theorem on real-world data. Three of these examples look at cancer in dogs and the eradication of smallpox. Study these. Then find open-source data on similar figures for Covid (transmission of the virus, figures related to mask wearing, etc) and Covid tests and vaccinations (test sensitivity, specificity, etc.) and repeat the exercise in Section 3.1 with your data.
