Hristina Hristova

Machine Learning with Naïve Bayes

Course Notes

365√DataScience

Table of Contents

Abstract

Naïve Bayes algorithms are supervised machine learning algorithms that are most often used in classification tasks. Python's **sklearn** library offers a variety of Naïve Bayes algorithms depending on the input data – whether it is numerical or categorical, balanced or imbalanced, dense or sparse. Their implementation is based on the famous Bayes' theorem, named after the English statistician Thomas Bayes, who has contributed significantly to the topic of probability theory.

Naïve Bayes algorithms are known to be fast learners, solve with ease problems in real-time, as well as handle sparse data. That is why they are preferred when faced with tasks involving text analysis, such as spam filtering, categorizing articles, sentiment analysis, etc.

These course notes give, in a very compact and comprehensive manner, the essentials for understanding Bayes' theorem and the Naïve Bayes algorithm. They serve as a complement to the video lectures by describing in further detail the theoretical concepts introduced there. The course notes could also be used as a quick reference to formulae, definitions, and key points.

Keywords: machine learning algorithm, Naïve Bayes, classification, Bayes' theorem, probability, spam, confusion matrix, accuracy, precision, recall, F_1 score

1. Motivation

In this section, we present a summary of the advantages and disadvantages of Naïve Bayes classifiers (Table 1) together with a couple of the most common use-cases.

Table 1: Pros and cons of Naïve Bayes classifiers.

Pros	**Cons**
Learns well from small datasets	Dependencies between features are not considered
Makes predictions in real-time	Not suited for regression tasks
Well-suited to non-linear problems	Probability estimates are not to be completely trusted
Good predictor	
Irrelevant features do not affect the performance	

Below, we list the most common applications to the Multinomial and Complement Naïve Bayes algorithms. The latter is implemented to handle imbalanced datasets better.

- Text analysis
    - Spam filtering
    - Categorizing documents
    - Categorizing news articles
    - Sentiment analysis
- Recommendation systems

2. Bayes' Thought Experiment

In 1763, two years after the passing of the British statistician Thomas Bayes, his friend Richard Price sends a letter[1] to the British physicist John Canton containing Bayes' lecture notes on probability theory. In this letter, Price discusses the importance of his friend's work. In Section II of the famous lecture notes, Bayes describes a thought experiment that we will recreate below in a simpler fashion.

Consider a square table divided into smaller squares in a 9-by-9 grid (Figure 1). The rules are as follows. Each cell in the grid can be occupied by a single ball. The balls cannot occupy two or more cells simultaneously, that is, once they fall onto the table, they should fully occupy one of the 81 cells.

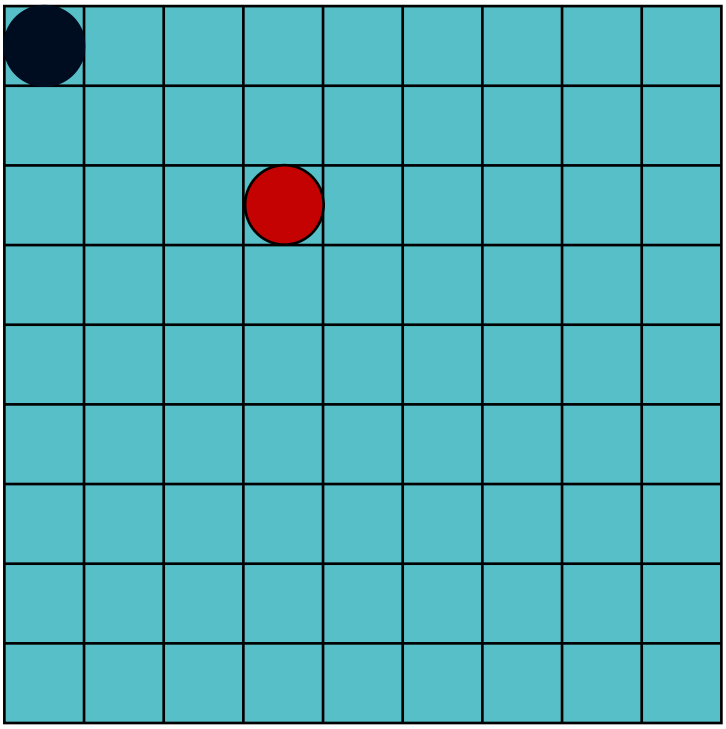


Figure 1: A square table divided into 81 squares. Imagine a red ball is tossed and lands somewhere on that table. Based on the rules and the conditions for the blue ball given in the text, where would you put the red ball?

[1] https://royalsocietypublishing.org/doi/pdf/10.1098/rstl.1763.0053

The orientation of the table is shown in Figure 2.

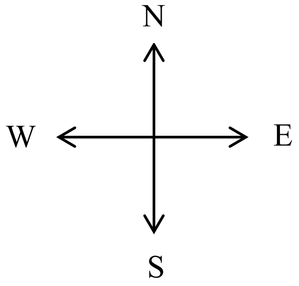


Figure 2: The orientation of the table.

Imagine a red ball is tossed and it lands somewhere on the table. The goal of this exercise is to guess the position of that ball based on the position of another, blue ball, with respect to the red one.

Example:

Consider the positions of the red and the blue ball in Figure 1 (not known a priori). The information you would be given, based on this arrangement, is:

"A blue ball is tossed, and it lands 2 squares north and 3 squares west of the red ball."

We are now prepared for the thought experiment. Five blue balls are tossed in a row and land in a cell on the table. The position of each one with respect to the red ball is recorded. Below, you will find those records.

1. A blue ball is tossed, and it lands 2 squares north and 1 square west of the red ball.
2. A second blue ball is tossed, and it lands 1 square north and 3 squares east of the red ball.

3. A third blue ball is tossed, and it lands 4 squares north and 3 squares west of the red ball.
4. A fourth blue ball is tossed, and it lands 5 squares north and 2 squares east of the red ball.
5. A fifth blue ball is tossed, and it lands 3 squares south and 5 squares east of the red ball.

Which cell do you think the red ball has landed in? You can find a discussion of this problem in the Appendix section.

3. Bayes' Theorem

3.1. The frequentist versus the Bayesian approaches

There are two ways to approach the topic of probability. The first one is the so-called *frequentist* approach. It is based on the idea of repeating an experiment a certain number of times and, as a result, estimating the probability of obtaining a particular outcome.

Example 1:

A novice darts player attempts to hit a target (Figure 3) consisting of three layers, with the outermost giving the least number of points and the innermost giving the maximum number of points.

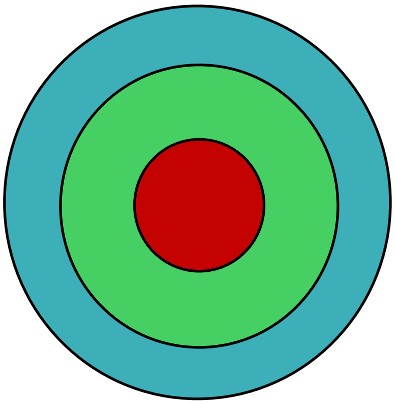


Figure 3: The target in a game of darts.

The darts player wonders what her chances of hitting the bull's eye are. She can resolve this problem by throwing the darts at the target 10 times. Assume that the dart ends up hitting the center twice. The player, therefore, concludes that 2 out of 10 times, or once every 5 throws, she will be awarded the maximum number of points. The larger the number of throws, the more data would be obtained, therefore the better the predictions would be. This example assumes, of course, that practice does not make the player better at throwing darts.

Through the *Bayesian* approach, on the other hand, we estimate the probability a certain hypothesis is true given past data (evidence). The difference with the frequentist approach is that the event whose outcome we are trying to predict will occur just once.

Example 2:

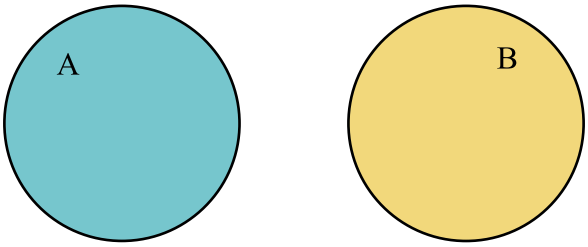
Consider the scenario from Example 1. This time, however, the already experienced darts player is competing in a World Championship. The audience is wondering what her chances of winning the game are. Unfortunately, this is not an experiment that can be conducted repeatedly – such championships are held only once every couple of years, so sufficient data would be hard to gather. Moreover, the outcome depends not only on the practice and experience she has gained but also on the practice and experience of the rest of the competitors.

The Bayesian approach is used to quantify the belief in an outcome that would occur only once. Relevant questions for Bayesian statistics could be:

- Will it rain tomorrow?
- Have I been diagnosed correctly?
- Who will win the presidential elections?

3.2. Marginal, joint, and conditional probabilities

Before we derive Bayes' theorem, we must introduce the relevant terminology. Consider two sets of events, A, and B.



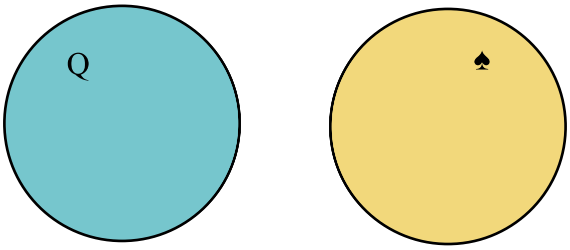
We define the following probabilities.

- **Marginal probability** – the probability of an event from either set occurring. It is denoted as follows

$$P(A)$$

<u>*Example:*</u>

Set A represents the event of drawing a Queen from a deck of cards. The event of drawing the Queen of Spades, or the Queen of Clubs, or any other Queen from the deck, belongs to set A. Set B represents the event of drawing a Spade. The event of drawing the Queen of Spades, or the King of Spades, or any other Spade from the deck, belongs to set B.



Since there are 4 Queens and 13 Spades in a deck of 52 cards, we can write down the marginal probabilities

$$P(A) = \frac{4}{52}$$

$$P(B) = \frac{13}{52}$$

- **Joint probability** – the probability of an event from set A and an event from set B occurring simultaneously. It is represented by the intersection of the two sets and is denoted as follows

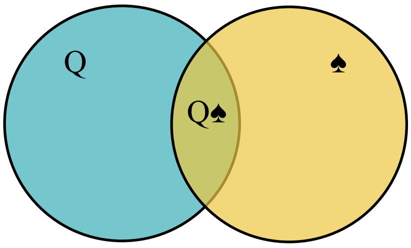
$$P(A \cap B)$$

Note that

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

<u>Example:</u>

The intersection of events A and B would be the event of drawing the Queen of Spades from a deck of cards.



Since there is 1 Queen of Spades in a deck of 52 cards, we can write down the joint probability

$$P(A \cap B) = \frac{1}{52}$$

- **Conditional probability** – the probability of an event occurring *given* that another event has taken place. It is denoted as follows

  $$P(A|B)$$

  The vertical bar in the parentheses is read as "given".

  Note that

  $$P(A|B) \neq P(B|A)$$

  for dependent events.

  Example:

  "A given B" would be the event of drawing a Queen given that the set of cards to choose from are all Spades. There is only one Queen in a set of 13 Spades. Therefore

  $$P(A|B) = \frac{1}{13}$$

  "B given A" would be the event of drawing a Spade given that the set of cards to choose from are all Queens. There is only one Spade in a set of 4 Queens. Therefore

  $$P(B|A) = \frac{1}{4}$$

3.3. The product rule

To derive the product rule, let's first make the following comment. Marginal probabilities are estimated by taking the ratio between the favorable outcomes and all outcomes:

$$P(A) = \frac{\text{favorable outcomes}}{\text{all outcomes}}$$

In a very similar fashion, we can define conditional probabilities. In such a case, the favorable outcomes are those entering the intersection between the two sets A and B. The probability of obtaining a favorable outcome is $P(A \cap B)$. All outcomes, on the other hand, are those for which event B takes place, no matter whether event A occurred or not. This probability is $P(B)$. Therefore,

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

Analogously,

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

Rearranging these equations, we obtain the **product rule**

$$P(A \cap B) = P(A|B)P(B)$$

and analogously

$$P(A \cap B) = P(B|A)P(A)$$

This result lies at the heart of proving Bayes' theorem.

3.4. Proof of Bayes' theorem

The result from the previous subsection showed that the intersection of two events can be expressed in two ways

$$P(A|B)P(B) = P(B|A)P(A)$$

Upon dividing both sides by $P(B)$, we obtain the celebrated **Bayes theorem**.

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

The result is spectacular in the sense that it connects the conditional probabilities of two events – knowing one of them can give us the other.

In the context of Bayesian statistics, event A is referred to as a hypothesis, denoted by H, whereas event B is referred to as evidence, denoted by E. The above result can then be rewritten as

$$P(H|E) = \frac{P(E|H)P(H)}{P(E)}$$

Therefore, to obtain the probability of a hypothesis being true, given some piece of evidence, we would need to estimate the probability of the evidence being true, given that our hypothesis holds. We would additionally need to provide the marginal probabilities of the hypothesis and the evidence being true individually.

3.5. Prior, posterior and likelihood

In the context of Bayesian statistics, each term in Bayes' theorem bears a specific name.

$$P(H|E) = \frac{P(E|H)P(H)}{P(E)}$$

- **Prior probability** – the probability of the hypothesis being true *before* seeing the evidence. In Bayes' theorem, it is denoted by $P(H)$.
- **Posterior probability** – the probability of the hypothesis being true *after* seeing the evidence. In Bayes' theorem, it is denoted by $P(H|E)$.

- **Likelihood function** – the probability of the evidence being true *given* the hypothesis holds. In Bayes' theorem, it is denoted by $P(E|H)$.
- **The normalization constant** – this term makes sure that, once all hypotheses have been considered and their conditional probabilities calculated, the sum of all posterior probabilities will be 1 (or 100%). Mathematically, this is expressed as follows

  $$\sum_{i=1}^{i=n} P(H_i|E) = 1$$

  where n is the number of possible hypotheses. In Bayes' theorem, the normalization constant is denoted by $P(E)$. It is calculated as follows

  $$P(E) = \sum_{i=1}^{i=n} P(H_i|E)P(H_i)$$

When applying the Naïve Bayes machine learning algorithm, one is typically interested in the hypothesis with the largest conditional probability rather than the exact value of the conditional probabilities of each hypothesis. Therefore, Bayes' theorem is usually applied in the following form

$$P(H|E) \propto P(E|H)P(H)$$

This avoids calculating the normalization constant $P(E)$.

4. The Ham or Spam Example

The setting in which the Naïve Bayes machine learning algorithm is used the most is text analysis. The example below is a continuation of the example from "The ham-or-spam example" video lecture, as well as the discussion from "The YouTube Dataset: Classification" video lecture.

<u>*Example:*</u>

Consider a training dataset consisting of 20 ham and 20 spam messages. The marginal (prior) probabilities of an email being ham or spam are therefore

$$P(\text{ham}) = P(\text{spam}) = \frac{1}{2}$$

Let's further consider the following vocabulary dictionary summarized in Table 2.

Table 2: Vocabulary dictionary before applying the Laplace smoothing parameter.

Word	Count in all ham messages	Count in all spam messages
dear	5	3
deadline	3	1
lecture	7	10
notes	9	9
assignment	6	7
student	15	0
Total word count	45	30

According to the table, the word "student" has not appeared in any of the 20 spam messages. Therefore, even if an incoming message that contains the word "student" is spam, it will not be considered as such by the model, as the conditional probability of the message being spam would always be 0. The remedy is to introduce the so-called smoothing parameter, α. Let's set its value equal to 1 (Laplace smoothing). The purpose of this parameter is to increase the count of each word so that the new counts become the ones in Table 3.

Table 3: Vocabulary dictionary after applying the Laplace smoothing parameter.

Word	**Count in all ham messages**	**Count in all spam messages**
dear	6	4
deadline	4	2
lecture	8	11
notes	10	10
assignment	7	8
student	16	1
Total count	51	36

Let's now apply Bayes' theorem in the known way.

$\mathrm{P}(\text{ham}|\text{dear, deadline, student}) \propto \mathrm{P}(\text{dear, deadline, student}|\text{ham})\mathrm{P}(\text{ham})$

$\mathrm{P}(\text{spam}|\text{dear, deadline, student}) \propto \mathrm{P}(\text{dear, deadline, student}|\text{spam})\mathrm{P}(\text{spam})$

Therefore

$$P(\text{ham}|\text{dear, deadline, student}) \propto \frac{6}{51}\cdot\frac{4}{51}\cdot\frac{16}{51}\cdot\frac{1}{2} \approx 1.4\times 10^{-3}$$

$$P(\text{spam}|\text{dear, deadline, student}) \propto \frac{4}{36}\cdot\frac{2}{36}\cdot\frac{1}{36}\cdot\frac{1}{2} \approx 8.6\times 10^{-5}$$

To substitute the proportionality sign with an equal sign, we need to further divide by $P($dear, deadline, student$)$. The way we calculate this quantity is as follows

$$\begin{aligned} &P(\text{dear, deadline, student}) \\ &\qquad = P(\text{dear, deadline, student}|\text{ham})P(\text{ham}) \\ &\qquad + P(\text{dear, deadline, student}|\text{spam})P(\text{spam}) \end{aligned}$$

Therefore, we need to add the two results we have just obtained:

$$P(\text{dear, deadline, student}) \approx 1.4\times 10^{-3} + 8.6\times 10^{-5} = 1.486\times 10^{-3}$$

The conditional probabilities for the message to belong to the ham or spam classes is the following

$$P(\text{ham}|\text{dear, deadline, student}) \approx \frac{1.4\times 10^{-3}}{1.486\times 10^{-3}} \approx 94\%$$

$$P(\text{spam}|\text{dear, deadline, student}) \approx \frac{8.6\times 10^{-5}}{1.486\times 10^{-3}} \approx 5.7\%$$

If the two results are now rounded, their sum would indeed equal 100%. In this way, not only did we learn which class the message belongs to, but we also managed to calculate the probabilities of the message belonging to either class.

5. Important Steps to Creating a Model

In this section, we outline the most important steps that need to be executed when creating a machine learning model. It is important that these steps are executed in the order given below.

1. Create the DataFrame

First and foremost, we need to create a **pandas** DataFrame where all inputs and targets are organized. Of course, a **pandas** DataFrame is not the only way to store a database, but it proves to be very useful. You are welcome to experiment with other means, but keep in mind that the **train_test_split()** method accepts the inputs and targets in the form of lists, **NumPy** arrays, **SciPy**-sparse matrices, as well as **pandas** DataFrames.

2. Data cleansing – check for null values

This step is part of the data pre-processing procedure. Before moving forward to the next step, do check for any null values in the data. There are various techniques to deal with this issue. One would be to remove the samples containing missing values altogether. This, however, can be done only if the number of such samples is much smaller than the number of all samples in the dataset. As a rule of thumb, if the number of samples containing null values is no more than 5% of the total number of samples, then removing them from the database should be safe. If that is not the case, statistical methods for filling up the missing values can be used instead.

3. Data cleansing – identify outliers

This step is part of the data pre-processing procedure. One should identify and remove any outliers in the data. The presence of samples with obscure values could cause misclassification of samples that would otherwise be classified correctly.

4. Split the data

Next, split the data into training and testing sets using, for example, **sklearn**'s **train_test_split()** method. An 80:20 split is very common. It would dedicate 80% of the data to the training set and 20% to the test set. Other splits such as 90:10, or 70:30 could, of course, be used as well. Use the training data to fit the model and the test data to evaluate its performance.

This step of splitting the data is one of the most common ways to avoid overfitting. Overfitting is a phenomenon where a model learns the data so well, that it also captures random noise in the data which affects its predictions. This is undesirable as random noise would inevitably be present in completely new datasets as well. An overtrained model would therefore perform poorly by misclassifying many of the points.

5. Data wrangling

This step is part of the data pre-processing procedure. In this step, we prepare the data for the classifier. Some classifiers, such as K-nearest neighbors, are based on distances between the samples. Such algorithms require standardized inputs, which usually imply transforming the data. Others, such as the Multinomial Naïve Bayes classifier, which is mainly used for text analysis, require vocabulary dictionaries in the form of sparse matrices. This would require transforming the inputs accordingly.

Such transformations carry information about the data. In the case of standardization, for example, the knowledge on the mean and standard deviation is gained. It is, therefore, dangerous to perform such transformations on the whole dataset, that is, before train-test splitting. Doing so could lead to *data leakage*.

6. Perform the classification

In this step, the appropriate classifier for the task is chosen, it is fit to the training data, and hyperparameters are tuned to achieve maximum performance.

7. Evaluate the performance of the model

Once the model is created and finetuned, it is time to test it on a new dataset. Metrics such as accuracy, precision, recall, and F_1 score are studied in the next section.

6. Relevant Metrics

In this section, we introduce some of the relevant metrics that could be used to evaluate the performance of a machine learning model dealing with a classification task.

6.1. The confusion matrix

A confusion matrix, C, is constructed such that each entry, C_{ij}, equals the number of observations known to be in group i and predicted to be in group j.

A confusion matrix is a square 2 × 2, or larger, matrix showing the number of (in)correctly predicted samples from each class.

Consider a classification problem where each sample in a dataset belongs to only one of two classes. We denote these two classes by 0 and 1 and, for the time being, define 1 to be the *positive* class. This would result in the confusion matrix from

Figure 4.

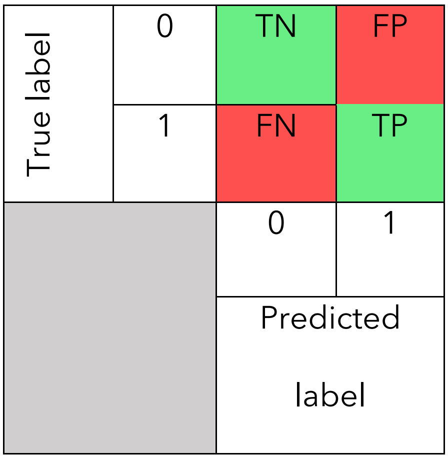


Figure 4: A 2 × 2 confusion matrix denoting the cells representing the true and false positives and negatives. Here, class 1 is defined as the positive one.

The matrix consists of the following cells:

- Top-left cell – **true negatives** (TN). This is the number of samples whose *true* class is 0 and the model has correctly classified them as such.
- Top-right cell – **false positives** (FP). This is the number of samples whose *true* class is 0 but have been incorrectly classified as 1s.
- Bottom-left cell – **false negatives** (FN). This is the number of samples whose *true* class is 1 but have been incorrectly classified as 0s.
- Bottom-right cell – **true positives** (TP). This is the number of samples whose *true* class is 1 and the model has correctly classified them as such.

Example:

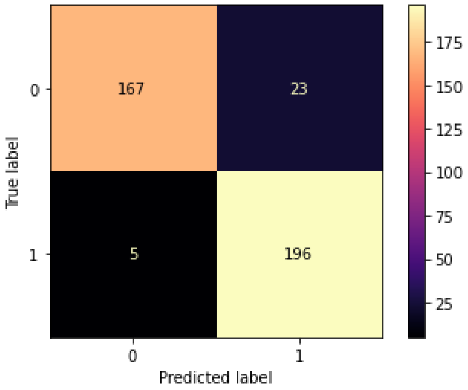


Figure 5: The figure shows a 2 × 2 confusion matrix in the form of a heatmap. A brighter color designates a higher number in the cell. We see that there are 167 true negative samples, 23 false positives, 5 false negatives, and 196 true positives.

Consider now a classification problem where each sample in a dataset belongs to one of three classes, 0, 1, or 2, with class 1 again defined as the *positive* class. This makes classes 0 and 2 *negative*. The confusion matrix would then look the one in

Figure 6:

True label	0	TN	FP	FN
	1	FN	TP	FN
	2	FN	FP	TN
		0	1	2
		Predicted label		

Figure 6: A 3 × 3 confusion matrix denoting the cells representing the true and false positives and negatives. Here, class 1 is defined as the positive one.

Example:

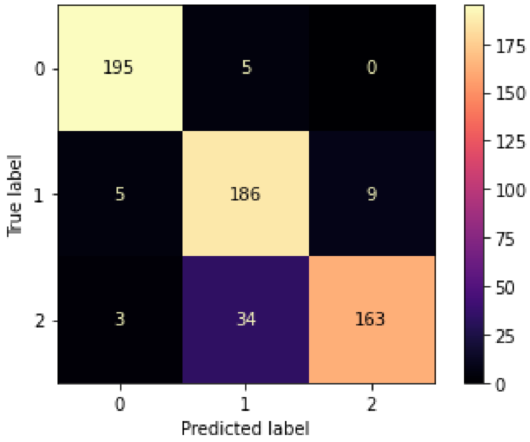


Figure 7: The figure shows a confusion matrix in the form of a heatmap. A brighter color designates a higher number in the cell.

Making use of these confusion matrices, we will introduce 4 useful metrics for evaluating the performance of a classifier. In Section 6.6, we construct a table summarizing all findings.

6.2. Accuracy

The ratio between the number of all correctly predicted samples and the number of all samples.

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

Example 1:

Using the confusion matrix from Figure 5, let's calculate the accuracy of the model.

$$\text{Accuracy} = \frac{167 + 196}{167 + 23 + 5 + 196} \approx 0.93$$

Example 2:

Using the confusion matrix from Figure 7, let's calculate the accuracy of the model.

$$\text{Accuracy} = \frac{195 + 186 + 163}{195 + 5 + 0 + 5 + 186 + 9 + 3 + 34 + 163} \approx 0.91$$

6.3. Precision

The ratio between the number of true positives and the number of all samples classified as positive.

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

Example 1:

Using the confusion matrix from Figure 5, let's calculate the precision of the model for both classes. To calculate the precision for class 0, we define that class as positive, while class 1 becomes negative. Applying the formula above, we obtain

$$\text{Precision}_0 = \frac{167}{167 + 5} \approx 0.97$$

Analogously, to calculate the precision for class 1, we define that class as positive, while class 0 becomes negative. Applying the definition, we obtain

$$\text{Precision}_1 = \frac{196}{196 + 23} \approx 0.89$$

Example 2:

Using the confusion matrix from Figure 7, let's calculate the precision of the model for all three classes. To calculate the precision for class 0, we define that class *as positive, while classes 1 and 2 become negative. Applying the definition, we obtain*

$$\text{Precision}_0 = \frac{195}{195+5+3} \approx 0.96$$

Analogously, the precisions for classes 1 and 2 are:

$$\text{Precision}_1 = \frac{186}{186+5+34} \approx 0.83$$

$$\text{Precision}_2 = \frac{163}{163+9+0} \approx 0.95$$

6.4. Recall

The ratio between the number of true positives and the number of all samples whose true class is the positive one.

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

Example 1:

Using the confusion matrix from Figure 5, let's calculate the recall of the model for both classes. To calculate the recall for class 0, we define that class as positive, while class 1 becomes negative. Applying the formula above, we obtain

$$\text{Recall}_0 = \frac{167}{167+23} \approx 0.88$$

Analogously, to calculate the recall for class 1, we define that class as positive, while class 0 becomes negative. Applying the definition, we obtain

$$\text{Recall}_1 = \frac{196}{196+5} \approx 0.98$$

Example 2:

Using the confusion matrix from Figure 7, let's calculate the recall of the model for all three classes. To calculate the precision for class 0, we define that class as positive, while classes 1 and 2 become negative. Applying the definition, we obtain

$$\text{Recall}_0 = \frac{195}{195 + 5 + 0} \approx 0.98$$

Analogously, the precisions for classes 1 and 2 are:

$$\text{Recall}_1 = \frac{186}{186 + 5 + 9} \approx 0.93$$

$$\text{Recall}_2 = \frac{163}{163 + 3 + 34} \approx 0.82$$

6.5. F1 score

The harmonic mean of precision and recall.

$$\text{F}_1 = \frac{2}{\frac{1}{\text{precision}} + \frac{1}{\text{recall}}}$$

The F_1 score can be thought of as putting precision and recall into a single metric. Contrary to taking the simple arithmetic mean of precision and recall, the F_1 score penalizes low values more heavily. That is to say, if either precision or recall is very low, while the other is high, the F_1 score would be significantly lower compared to the ordinary arithmetic mean.

In the examples below, we calculate the F_1 score for class 0 for both the 2 × 2 and the 3 × 3 confusion matrices. Calculating the F_1 score for the rest of the classes is left as an exercise to the reader. The answers are provided in Section 6.6 with a table summarizing the results from all metrics.

Example 1:

Using the confusion matrix from Figure 5, we calculated the precision and recall values for class 0 to be 0.97 and 0.88, respectively. Applying the definition, the F_1 score is

$$\mathrm{F}_1 = \frac{2}{\frac{1}{0.97} + \frac{1}{0.88}} \approx 0.923$$

In contrast, the arithmetic mean is

$$\text{Arithmetic mean} = \frac{0.97 + 0.88}{2} = 0.925$$

We see that the arithmetic mean is slightly larger than the F_1 score. The difference here is not big as both precision and recall are quite high in value. The discrepancy between the F_1 score and the arithmetic mean would be much more apparent if the difference between precision and recall is much larger.

Example 2:

Using the confusion matrix from Figure 7, we calculated the precision and recall values for class 0 to be 0.96 and 0.98, respectively. Applying the definition, the F_1 score is

$$\mathrm{F}_1 = \frac{2}{\frac{1}{0.96} + \frac{1}{0.98}} \approx 0.970$$

In contrast, the arithmetic mean is

$$\text{Arithmetic mean} = \frac{0.96 + 0.98}{2} = 0.97$$

When precision is almost equal to recall, the F_1 score and the arithmetic mean bear almost the same value.

6.6. Summary

In Table 4 and Table 5, we summarize the results of both confusion matrices. This is known as a classification report.

Table 4: A classification report for the model that has output the 2 × 2 confusion matrix.

Metric / Class	Precision	Recall	F_1 score	Accuracy
0	0.97	0.88	0.92	
1	0.89	0.98	0.93	
				0.93

Table 5: A classification report for the model that has output the 3 × 3 confusion matrix.

Metric / Class	Precision	Recall	F_1 score	Accuracy
0	0.96	0.98	0.97	
1	0.83	0.93	0.88	
2	0.95	0.82	0.88	
				0.91

Appendix

Bayes' Thought Experiment

In this simplified version of the problem, we aim at showing that, as new knowledge is gained, the predictions become more and more accurate.

With each toss of the blue ball, the number of possible spots for the red ball becomes smaller and smaller. In Figure 10 to Figure 12 below, you can see the possible choices going from 81 down to 4. In Figure 13, the original setup of the problem is revealed.

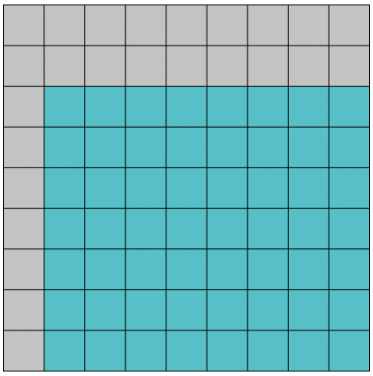


Figure 10: The possible spots for the red ball after gaining information on the position of the first blue ball with respect to the red ball.

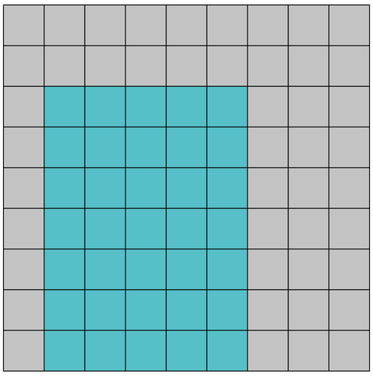


Figure 9: The possible spots for the red ball after gaining information on the position of the second blue ball with respect to the red ball.

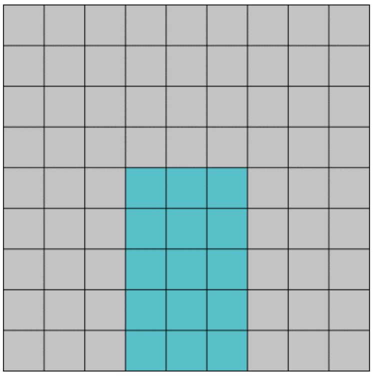


Figure 8: The possible spots for the red ball after gaining information on the position of the third blue ball with respect to the red ball.

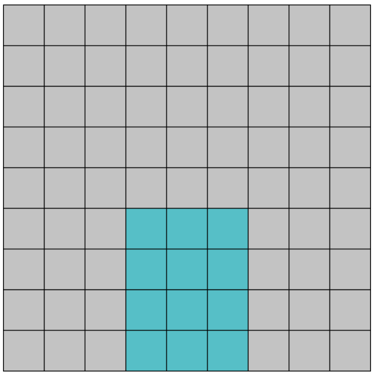


Figure 11: The possible spots for the red ball after gaining information on the position of the fourth blue ball with respect to the red ball.

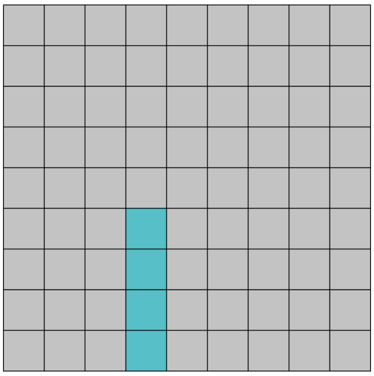


Figure 12: The possible spots for the red ball after gaining information on the position of the fifth blue ball with respect to the red ball.

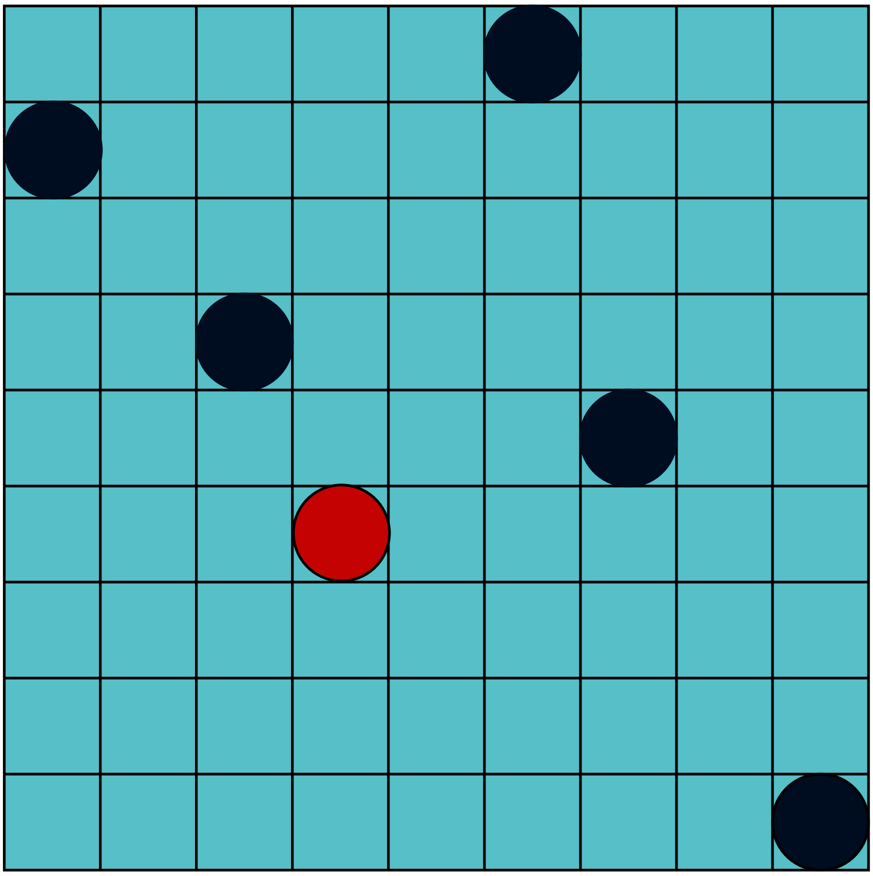


Figure 13: The original setup of the problem.

Hristina Hristova

Email: team@365datascience.com

365√DataScience