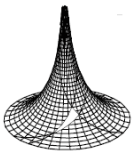

18/4 (2020), 307-316
DOI: 10.5485/TMCS.2020.0494
tmcs@science.unideb.hu
http://tmcs.math.unideb.hu

Teaching
Mathematics and
Computer Science

Integrating elements of data science into high-school teaching: Naïve Bayes-classification algorithm and programming in Python

ÖDÖN VANCSÓ AND PÉTER PRINCZ

Abstract. Probability theory and mathematical statistics are traditionally one of the most difficult chapters of mathematics to teach. One of the authors, Péter Princz has experience in teaching various topics via computer programming of the problem at hand as a class activity. The proposed method is to involve programming as a didactic tool in hard-to-teach topics. The intended goal in this case is to implement a naïve Bayes-classifier algorithm in Python and demonstrate the machine-learning capabilities of it by applying it to a real-world dataset of edible or poisonous mushrooms. The students would implement the algorithm in a playful and interactive way. The proposed incremental development process aligns well with the spirit of Tamás Varga who considered computers as modern tools of experimental problem solving as early as in the 1960s.

Key words and phrases: Bayesian classifier, computer programming, guided discovery learning.

MSC Subject Classification: 97D40, 97D50, 97K50, 97K99, 97M60, 97P40, 97P50, 97U50.

Historical and theoretical background

Probability theory and mathematical statistics are traditionally among the most difficult fields of mathematics to teach. In Hungary, at least probability theory is present in the secondary education. Elements of inferential statistics have been planned to be included in the final exam, but they have been omitted from the final revision because of reduction in the hours devoted to the subject of mathematics. This historical progression is documented in Vancsó (2015). There are new approaches ever since: there is an ongoing work within the MTA-ELTE *Recent Complex Mathematics Education Research Group* that aims to develop new, experimental educational material with the involvement of professor M. Borovcnik from Klagenfurt and two PhD students as well as with high-school teachers. The concept is described in detail in Borovcnik, Fejes-Tóth, Jánvári, and Vancsó (2020).

One of the authors, Péter Princz has experience in teaching various topics via computer programming as a class activity. A frequent observation during such programming classes is how students digest a new, complex concept, such as gravity or trajectory: first, they grasp it intuitively and on the surface only, but shortly after, they arrive at a deeper understanding, compared to the frontal-only teaching method. They are forced to transfer the concept to practice when using it in a program and then they can play with it and its effects interactively, in short feedback loops.

It is important to emphasise the role of the computer in this approach: students do not only run a computer program prepared in advance, or do not only change parameters in the simulations. These are all useful activities, but instead, the students are writing the working code of the problem as a class or pair or individual activity. The ambition is to involve computer programming as a didactic tool in hard-to-teach topics. The method is influenced by the seminal work of Tamás Varga, but also by Seymour Papert's Logo, by Guido van Rossum's Python programming for everybody (van Rossum, 1999), and to some extent even by Donald E. Knuth's literate programming (Knuth, 1992). One such example is the author's own experiment: *Programming the projectile motion for ninth graders*, which is taught with certain regularity (Princz, 2018).

Why the naïve Bayes classifier?

As we stated earlier, probability is not easy to understand. The Bayesian approach seems to be a good choice to develop the concepts and the toolkit of probability in

grades 9 – 12. Its theoretical background (Law of Total Probability, Bayes' Theorem) is not as simple as it may seem, with reasoning pitfalls, resulting in lamentable scores even among students at university level (Bordács, 2002).

The Bayesian way of problem statement is obvious and easy to understand for students. For example, "Is this mushroom edible or poisonous?" Both the answer to the questions posed and the way the classification is made are natural for this age group. This can engage even those students who dislike mathematics and can help the teacher in gaining their interest and active participation in the classroom activities. Finally, students will acquire practical and useful knowledge, all in a collaborative way, when attempting to solve a Bayesian problem via coding. Apart from these, it is an industry algorithm, suitable for a first approximation in classification problems. The term "naïve" is an indicator of the assumed independence of the attributes in the dataset. With a proper training dataset, the classification can be as accurate as 70 – 75%. Of course, this precision is not enough for a life-death decision as in this case, but it is good enough for a first classification. It has therefore a role in machine learning as well.

Russell and Norvig's (2010) seminal book on Artificial Intelligence (AI) pays a special tribute to Thomas Bayes. Its cover is a piece of art in itself: a careful selection of people and concepts fundamental to modern AI. Thomas Bayes has a special place in the top row. The book even explains why (p. ix):

"...Kasparov is shown at the top. To his left is the Asimo humanoid robot and to his right is Thomas Bayes (1702–1761), whose ideas about probability as a measure of belief underlie much of modern AI technology."

Theoretical background

As one of the frontal-teaching elements of a Bayesian computer-programming curriculum, the teacher has to discuss the theorem, maybe outline its proof or give intuitive reasons what the formula does and how the constituents can be interpreted, and work through at least one simple textbook example with the students. Vancsó (2004) already incorporated the double-tree diagram notation to depict probability problems as a lesson learned from Bordács (2002) and other sources.

Expanding on Bayes theorem, the remaining piece of the theoretical background is the naïve Bayes-classifier algorithm itself. It is used for a first approximation to classification problems in machine learning: Given is a *training dataset* of observations

of *known* classes (i.e., it is known to which class the statistical unit related to the data belongs), and a new *test observation*. The problem is to identify into which of the classes a new test observation falls. Some real-world examples where a naïve Bayes classifier performs surprisingly well are:

- have a training set of known spam e-mails; decide whether a new e-mail in the inbox is legitimate or spam;
- have a training set of people's features such as height, weight, shoe size and gender; decide whether a new (height, weight, shoe size) data relates to a boy or a girl;
- have a dataset of mushroom features including the expert classification whether they are edible or poisonous; decide whether a new mushroom is edible or poisonous (even if it is a never-seen-before mushroom).

A *probabilistic classifier* in machine learning is a classifier that can predict, given an observation as input, a probability distribution over a set of classes. That is, for each class, a probability that the given object belongs to this class. Hence, its output is not only the most likely class to which the observation belongs to, but also a probability for *every* class. In our example of mushrooms, a probabilistic classifier would compute two probabilities for every test object (mushroom), i.e. a probability for being edible and a probability for being poisonous for this specific mushroom. The final judgement for a given test object is the class that attains the maximum of the probabilities.

Naïve Bayes classifiers are a family of probabilistic classifiers applying Bayes' theorem to their training and test sets. "Naïve" relates to the fact that they assume the features to be independent from each other. Hence their other names: simple Bayes classifiers or independence Bayes classifiers. For example, they regard height, weight and shoe size as independent features of a person. In reality, taller people tend to be heavier and have a larger shoe size, so that the features in the training set are in fact dependent. Nevertheless, these classifiers perform surprisingly well even if the independence assumption fails (see Russell & Norvig, 2010). The teacher must communicate these details for the students in a proper way.

The mushroom problem

Key to machine-learning algorithms is to provide a good-quality *training dataset*. In practice this is not a trivial problem; in most cases, the raw data must be cleaned, normalised, the missing data to be filled in with zeros, default or estimated values. The data practitioners refer to these preparation activities as *data wrangling* and in some cases, they can be more complex than the actual model fitting and the visualisation. When it comes to bringing elements of data science into the classroom, it makes sense to touch this topic briefly, maybe by showing a before-and-after example but not to waste too much time and effort on it. As we will see, even the adjusted training set of mushrooms needs some preparation before the students can work with it. The didactic goal with the naïve Bayes classifier is to find a problem that is interesting for the students and to provide a relatively large but simple training set of good quality, preferably a comma-separated text file that can be loaded into Excel or any other spreadsheet for quick ad-hoc analyses. After some investigations, we discovered the mushroom classification challenge.

Data-science practitioners have an online community hosted by Google, named Kaggle. They are regularly organising competitions incentivised by prize money. The challenge provides a dataset reviewed by scientists that is to be used by all competitors as training dataset to train their models; then they must apply their model to make predictions for new data. In 2016, they organized the mushroom classification competition (Kaggle, 2016).

We downloaded the adjusted dataset from there. It is a simple csv file and includes a description of mushrooms of the Agaricus and Lepiota Family (Lincoff, 1981, pp. 500-525). This field guide identifies each species as definitely edible or poisonous, or of unknown edibility and not recommended. Kaggle combined the third class with the poisonous one so that two classes are left. The guide clearly states that there is no simple rule for determining the edibility of a mushroom. The dataset (see *Figure 1*) consists of more than 8000 data rows with twenty-two feature columns. There is an additional column for the classification (the very first in the text file), stating whether a given mushroom is edible (“e”) or poisonous (“p”). Approximately half (52%) of the mushrooms is edible.

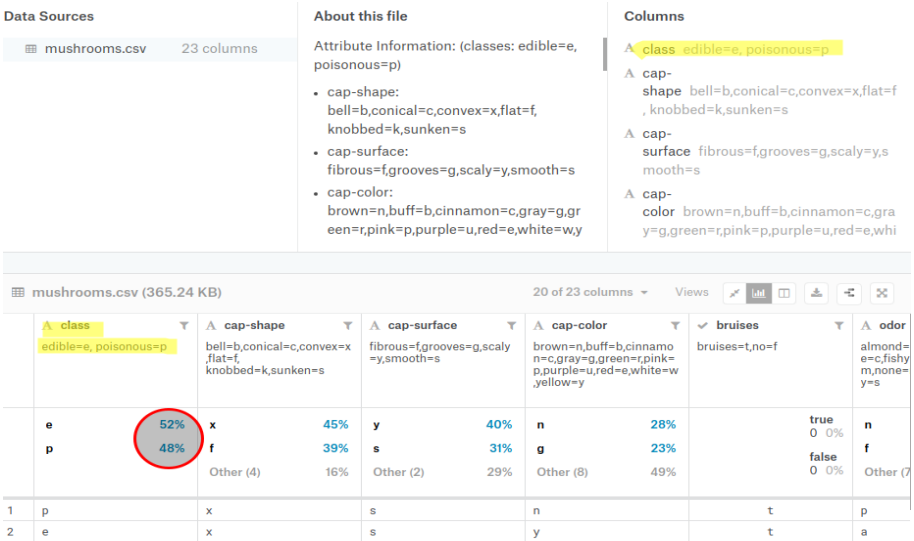

Figure 1. Excerpt of the mushroom competition dataset

The attributes as well as the categories in the text file contain alphanumerical codes, known as *categorical values*. Some examples follow:

1. cap-shape: bell = b, conical = c, convex = x, flat = f, knobbed = k, sunken = s
2. cap-surface: fibrous = f, grooves = g, scaly = y, smooth = s
3. cap-color: brown = n, buff = b, …, grey = g, green = r ,…, yellow = y

...

21. population: abundant = a, clustered = c, numerous = n, …, solitary = y
22. habitat: grasses = g, leaves = l, meadows = m, paths = p, …, woods = d

The problem with categorical values is that one cannot directly use them in algorithms, so that data wrangling is needed. The operation needed here is called *one-hot-encoding*; its purpose is to replace the categorical values of one feature by several binary features. For example, the first attribute above is the cap shape, which can attain six different values (*b*, *c*, *x*, *f*, *k* or *s*). With one-hot-encoding, one replaces this column of cap shape by six distinct columns (the number of possible categorical values). For every data row, a value of 1 in the new column means the given mushroom's cap shape coincides with that category. The other five added columns attain the value 0, which means that the cap shape is different from that category. Consistently, one replaces all categorical values by a bit vector with the length of the possible categories for each. In *Figure 1*, the first two columns of the first row contain the data "*class = p, cap shape = x*",

which means that this is a poisonous mushroom with a convex cap shape. In the wrangled file, the cap shape is represented by:

cap-shape_b = 0, cap-shape_c = 0, **cap-shape_x = 1**, cap-shape_f = 0, cap-shape_k = 0, cap-shape_s = 0.

One must apply the one-hot-encoding to all categorical features with more than two values. The resulting wrangled file is ready for processing by the students. Special library functions support this tedious preparation; yet, we do this step in advance to spare time and keep students' interest. In our case, the clean file is part of the educational material, ready to download. The teacher only has to highlight the concept of categorical values, the data wrangling, and the one-hot-encoding for the students so that they can pre-process any data file they would meet in the future.

Connection to the legacy of Tamás Varga

The way, the programming of the algorithm is conducted as a classroom activity recalls the perception of Varga on teaching mathematics. We collected two quotes on his methods, translated from Pálfalvi (2019, pp. 32-23):

"The biggest intrinsic motivator shall be the joy of discovery; it should be welcomed *to make mistakes while working* and one should not be blamed for it."

"For a number of seemingly trivial methodological aspects, it was difficult to accept and to get them accepted. Such principles are: getting motivated, differentiation, the fact that *the teacher is not the only source of knowledge*, *independent discovery is worth more than a drill commanded from above*, to be happy when a *student surprises the teacher* with an unplanned but good solution, etc. It was also a novelty that Varga advocated a diversity of teaching methods and various forms of work, including *group activities*, efficient application of *games* and gamification, free debate, and making mistakes by trial and error, over the then-prevalent frontal class work."

The similarity to programming a mathematical problem with students is striking. "Making mistakes during work is OK" is in Varga's spirit as programming is all about making small mistakes all the time and refining towards the solution of the problem. The emphasis in the second quote reflects how we conduct these programming workshops: the teacher is not the only source of knowledge, students experience independent discovery every minute by trial and error, they are surprising the teacher all the time, and this is a group activity signified by fun.

Tamás Varga has always been embracing novel approaches to education as well as even the latest gadgets of his time (see Fried, Fekete, & Princz, 2020). We believe both the mathematical content, i.e., the Bayesian approach to probability theory as well as the novel approach to teach this content in secondary school via computer programming is worthy of Varga's legacy in teaching mathematics. Varga brought in brand-new concepts, methods, and tools even from outside mathematics, such as flow charts, punch cards, etc., into elementary school. A notable example is his two-volume book for 8-14-year-old students: (Varga, 1972; 1976). He embraced algorithmic thinking and flowcharts well before the personal-computer era, probability at early ages in a playful manner with dice and coins, etc. Maybe, Varga would bring in elements of data science, machine-learning algorithms, and programming into today's classroom.

Conclusion

Which chapters of probability theory and mathematical statistics to cover and how to teach them, is still an ongoing discussion in the didactics of mathematics and among mathematics teachers. Frequentist vs. Bayesian interpretations of probability theory are debated. On this issue, see Carranza and Kuzniak (2008) or Vancsó (2009). In Hungary, Tamás Varga had a seminal role in bringing in probability theory into the national curriculum. However, the Bayesian approach and methods are still a novelty in a typical Hungarian high school, even as an extra-curricular activity. There are books approaching the problem from the educators' perspective, see Borovcnik (1992) and Batanero and Borovcnik (2016). Yet, teaching rarely covers Bayesian aspects adequately, and this applies worldwide.

The method presented in this paper is an attempt to bring in the Bayesian interpretation of probability into teaching mathematics. One of the novelties of the method presented here is to involve computer programming as a didactic tool in hard-to-teach fields of mathematics. During their professional development, the authors have tried several languages and tools, such as Logo, Scratch, and Processing as well. So far, the most valuable approach from a pedagogical point of view is the work of Seymour Papert and his Logo for simple visualisations as well as for pixel-by-pixel plotting of functions, i.e., trajectories. The work of Guido van Rossum (1999) and his Python programming language is also worth to mention here. Luckily, the Turtle module of Python combines the two: it is embedding Logo into Python. It is also important that

coding in Python is not much more verbose than the mathematical formula from which it translates. It is a prose-like code that is easy to read and thus very suitable for teaching purposes and it can even replace pseudocode when discussing algorithms.

Following the spirit of Tamás Varga, we believe that – embedded in technology – the modern pedagogical tool of discovery learning and guided experimentation in mathematics is computer programming.

References

Batanero, C., & Borovcnik, M. (2016). *Statistics and probability in high school.* Rotterdam: Sense Publishers.

Bordács, N. (2002). Bayesian-type problems in everyday life. An experiment conducted among university students. Thesis work guided by Ö. Vancsó. Eötvös Loránd University, Faculty of Sciences, Budapest.

Borovcnik, M. (1992). *Stochastik im Wechselspiel von Intuitionen und Mathematik.* Mannheim: Wissenschaftsverlag.

Borovcnik, M., Fejes-Tóth, P., Jánvári, Z., & Vancsó, Ö. (2020). Experimente zur Einführung von Ideen und Denkweisen statistischer Inferenz im Gymnasium. *Stochastik in der Schule, 40(1),* 18-27.

Carranza, P., & Kuzniak, A. (2008). *Duality of probability and statistics teaching in French education.* Joint ICMI/IASE Study: Teaching Statistics in School Mathematics. In: C. Batanero, G. Burrill, C. Reading, & A. Rossman (Eds.), Challenges for teaching and teacher education., Monterrey: ICMI & IASE. Obtenido de iase-web.org/Conference_Proceedings.php?p=Joint_ICMI-IASE_Study_2008

Fried, K., Fekete, I., & Princz, P. (2020). Better understanding mathematics by algorithmic thinking and computer programming. *Teaching Mathematics and Computer Science, 18*(4), 295-305. doi:10.5485/TMCS.2020.0486

Kaggle. (2016). *Mushroom Classification Safe to eat or deadly poison?* (UCI Machine Learning) Obtenido de https://www.kaggle.com/uciml/mushroom-classification

Knuth, D. E. (1992). *Literate programming.* Stanford, CA: Centre for the Study of Language & Information.

Lincoff, G. (1981). *National Audubon Society field guide to North American mushrooms.*

Pálfalvi, J. (2019). *Varga Tamás élete. A komplex matematikatanítási kísérlet. (The life of Tamás Varga. The experiment of complex mathematics education).* Budapest: Typotex.

Princz, P. (2018). *Lunar lander: a 4 hour Python workshop to teach programming and projectile motion*. Obtenido de Collection of public curricula: https://gitlab.com/princzp/4hr_python_workshop

Russell, S., & Norvig, P. (2010). *Artificial Intelligence. A modern approach. In: Prentice Hall Series in Artificial Intelligence.* (3rd ed.). New Jersey: Pearson Education, Inc. Obtenido de http://aima.cs.berkeley.edu

van Rossum, G. (1999). *Computer Programming for Everybody (Revised Proposal). A Scouting Expedition for the Programmers of Tomorrow.* Corporation for National Research Initiatives. doi:CNRI Proposal # 90120-1a.

Vancsó, Ö. (Ed.). (2004). *Matematika 11. Mathematics school book for grade 11 (in Hungarian only).* Budapest: Műszaki Könyvkiadó.

Vancsó, Ö. (2009). Parallel discussion of classical and Bayesian ways as an introduction to statistical inference. *International Journal of Science and Mathematics Education (IEJME), Vol. 4.*(Issue 3., pp. 291-322.).

Vancsó, Ö. (2015). Mathematics final exam in Hungary. (Die Mathematik Matura in Ungarn) Habiltationsschrift Alpen-Adria Universität Klagenfurt.

Varga, T. (1972). *Játsszunk matematikát! Let's play mathematics! Flowcharts – punch cards – probability* (Vol. I). Budapest: Móra Könyvkiadó.

Varga, T. (1976). *Játsszunk matematikát! Let's play mathematics! Space and plane – Probability – Logic and combinatorics* (Vol. II). Budapest: Móra Könyvkiadó.

ÖDÖN VANCSÓ
ELTE, FACULTY OF SCIENCE, INSTITUTE OF MATHEMATICS
1117 BUDAPEST, PÁZMÁNY PÉTER SÉTÁNY 1/C.
E-mail: vancso.odon@gmail.com

PÉTER PRINCZ
ELTE, FACULTY OF SCIENCE, INSTITUTE OF MATHEMATICS
1117 BUDAPEST, PÁZMÁNY PÉTER SÉTÁNY 1/C
E-mail: princzp@caesar.elte.hu