Practicing The Perceptron: Introducing Data Mining To The CS2 Classroom

François Neville
Department of Computer Science
Bemidji State University
Bemidji, MN 56601
fneville@bemidjistate.edu

Abstract

We describe a 2-week team project designed for a typical CS2 class, giving students experiences crafting a complex object-class called a Perceptron. The basic perceptron is a statistical machine learning model resulting from a training algorithm capable of distinguishing between two or more categories of input vectors. In the process, students are introduced to basic concepts of data mining and data visualization, and use the model they build to explore datasets of interest to them. Several semesters of assigning variations of this project to students of varying levels in our Computer Science program refined the process for introducing this topic to students within our CS2 curriculum so it appears both relevant and meaningful. Initial observations indicated that having students work in small teams enhanced their individual understanding, and thus the ultimate success of their project. Reports from past students indicated having a choice in final datasets to use, as well as the teamwork aspects made this project particularly compelling to computer science majors in our program, a potential positive indicator for our departmental retention goals.

Keywords: CS2, Machine Learning, Perceptron, Data Visualization, Data Mining

1. Introduction

A frequent theme in CS education involves the benefits students accrue from formulating assigned projects within a *context* [8] [13], preferably one relevant to them (or one they would be likely to encounter in the “real world”), and often within some academic discipline other than Computer Science [2] [3] [9] [16]. Introductory CS textbooks increasingly deliver material *in context*. For example, Miller & Ranum’s *Python Programming in Context* [11] presents programming constructs within specific projects drawn from subjects such as astronomy, earthquakes and population dynamics. Guttag’s introductory text *Introduction to computation and programming using Python* [7] communicates the same material within a data science context. Recent articles (and indeed many *Nifty Assignments*) place their lessons within contexts that range from Media Studies [16], to Data Analysis [1], to Sustainability Research [3].

A benefit to providing students an approximation of real-world experience is in better preparing them for careers after graduation [20]. Perhaps most notably, Steve Cunningham suggested in 2008 that creating a context for a course had a threefold effect: it could 1) help increase student engagement with the material, 2) allow for greater self-expression within the discipline, and 3) foster collaboration and a greater sense of community [4]. These are among the factors that Rich et al (2004) later identified whose absence seem to contribute to declines in retention of women within CS programs [16].

In the initial semesters of a projected four years of late nights spent within campus computer labs laboring over contrived exercises involving abstractions such as queues or hashtables, it can be difficult for students to imagine a professional career in computer science that might involve anything else – such as field work assessing water quality, or categorizing vegetation samples, for example. However, computer science is useful in many other academic disciplines, encompassing not only natural sciences and business administration, but extending through health sciences and into the liberal arts [18]. For computer science departments concerned with the retention numbers of students within their program, it might well seem beneficial to contextualize some portion of the curriculum within a framework attractive to populations of students that might not otherwise consider a CS major.

“Big Data” analysis as a narrow subdiscipline of computer science has participated perhaps like no other in recent years’ zeitgeist, garnering expressions of both public anxiety [12] and acclaim [14]. Given the growing attention paid to this topic, it is natural that an attempt should be made to place a series of CS2-level labs within the Big Data context. The Perceptron, a relatively simple, straightforward statistical model from machine learning was selected for this purpose.

Our initial observations after three years of iterative refinement to the Perceptron project as assigned to our CS2 classes are presented here. Subsequent sections deal with the preparation required to set the context for second or third semester CS students, descriptions of problem sets provided with their results, and possible extensions to the base project. Finally, we surveyed a past CS2 cohort having recently experienced this project. A standard set of fourteen questions was used to determine participants' reaction to various aspects of the project including the issues raised by Cunningham. Initial results suggest that student engagement with this project is high, and that opportunity to explore datasets of greater personal interest to students – as well as the opportunity to collaborate in teams – were also appreciated by participating students.

2. Context Setting

The CS2 curriculum at our four year liberal arts college is traditionally structured, extending upon basic knowledge gained in CS1 with more advanced data structures and algorithms. CS2 provides a second consecutive semester of instruction in the Python programming language. In labs, timing results of new algorithms are output to Excel readable *.CSV* files in order to subsequently illustrate and confirm theoretical big-O performance levels using Excel charts. This experience provided some familiarity with the Excel interface, and instilled student confidence in designing data visualizations to complement their growing understanding of new material from the required course text [10].

Proposed by Frank Rosenblatt in 1958 [17], a Perceptron is a conceptually simple binary classification model – that is, it adapts its inner weights through an inductive *supervised learning* process in order to successfully distinguish between two classes of *n*-dimensional points. The model iterates through a training dataset of two identified classes of points, until it converges to the coefficients of a linear structure separating most or all of the points of one class from most or all of the points of the other. In the 2-dimensional case, its results may be displayed in the familiar form of a standard $y = mx + b$ equation of a line – the only mathematical background necessary for this project – that can be readily rendered in Excel or any graphing application. In our experience, given our student's prior familiarity with graphing utilities, the context and required knowledge for this project can be adequately delivered in the course of one ninety minute lab session.

2.1 The Perceptron Rule

At its core, the perceptron is simply a ($d+1$)-length array of floating-point values. These values are used to construct a linear boundary in *d*-dimensional space between two sets of binary-classified points. The Perceptron model proposed in laboratory sessions is initialized with three matrices: a ($n \times (d+1)$) dataset (*data*) of classified *d*-dimensional points, to which a column of ones (i.e. constants) is appended; an *n*-length array (*category*) of the classification of each point in data – labeled as either +1 or -1; and a ($d+1$)-length array (*p-vector*) initialized to random (or zero) values. The sole specific functionality of the Perceptron is a decision structure known as the *Perceptron Rule*. This rule's essential thrust is that if the current model's *p-vector* derives an incorrect classification for a particular point during training (where binary classification is encoded as the sign of the returned model-value), then the *p-vector* must be additively modified, as demonstrated by the following pseudocode:

Perceptron Training Algorithm

```
FOR EACH row IN data:
   model-value = Dot_Product(row, p-vector)
       /* Perceptron Rule */
   IF Sign(model-value) <> Sign(class) // p-vector must be changed
          IF model-value < class        // not positive enough
             add row to p-vector
          ELSEIF model-value > class    // not negative enough
             subtract row from p-vector
```

Convergence to a correct solution for the training set has occurred when an entire pass through *data* results in no changes to *p-vector*. However, convergence will only occur if the two classes of points within the dataset are *linearly separable*. Should the two classes of points not be linearly separable (as will happen in natural datasets), training may stop at any point that a minimum error count has been reached.

3. Project Description

3.1 Introductory Gradebook Project

Students are introduced to the project concept through a collection of randomly generated quiz grades (*data*) for three separate quizzes – a 5-point quiz, a 10-point quiz and a 20-point quiz (we feel that mapping *total_points* to the x-axis and *points_received* to the yaxis in a chart produces the most intuitive visualization – so a grade of 4/5 is encoded as the point (5, 4)). As a group, students generate the *category* array by classifying each point as either *Passing* or *Failing* by respectively assigning it either a +1 or -1 (note that actual sign and magnitude of the classifier is immaterial, so long as the same arbitrary value is always used to represent the same category *(e.g. -1: Passing; +1: Failing)*). For this exercise, a deterministic classification can be made with the following pseudocode:

```
IF y/x >= 0.6 THEN
   class = -1
ELSE
   class = +1
```

The 2D point/grades may now be plotted on a chart.

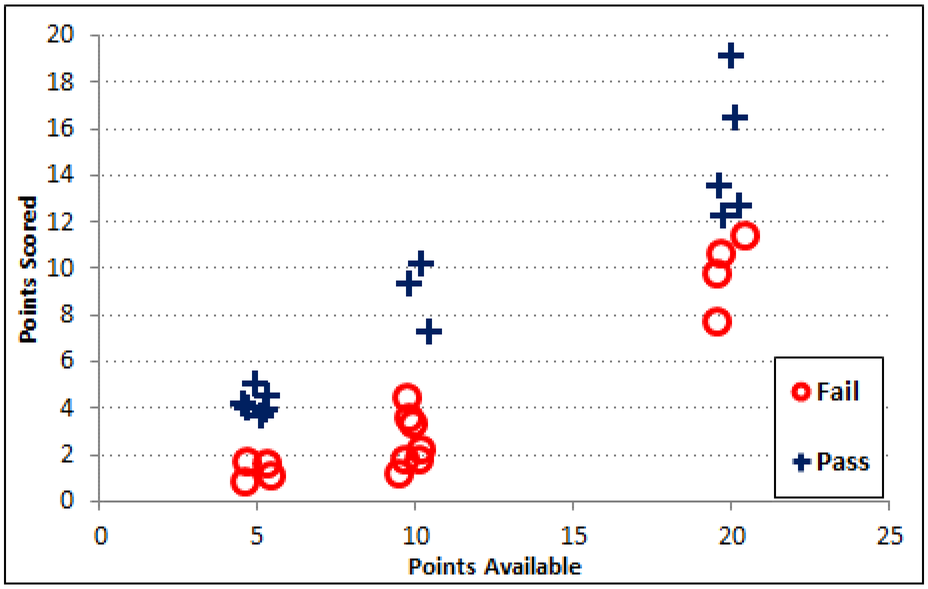

Figure 1: Example of 2D Gradebook Project data represented as a graph

The Perceptron Training Algorithm specified above may now be executed over this dataset until changes to the *p-vector* no longer occur. Although the number of solutions to a linearly separable problem is technically infinite, the result of the application of the Perceptron Rule over one or more passes through this dataset will typically result in a *p-vector* similar to [18, -10, -1], which translates to the linear equation:

$$0 = 18y - 10x - 1 \qquad \text{(Eq. 3.1a)}$$

or, more familiarly:

$$y = \tfrac{5}{9}x + \tfrac{1}{18} \qquad \text{(Eq. 3.1b)}$$

While this solution is correct with regard to the data provided, it is also very close to the general solution, which matches our intuitive understanding of the distinction between passing and failing grades:

$$y = 0.6x + 0 \qquad \text{(Eq. 3.1c)}$$

That is, points that are located on or above the line in Eq. 3.1c are classified as passing, while those below it are failing.

3.2 Fisher's Iris dataset (1936)

The introductory Gradebook exercise is followed by one using a subset of the well-known *Iris* dataset compiled by Ronald Fisher to aid in categorizing various species of iris flower [6]. The full dataset comprises three species of iris, one of whose individual measurements (*I. setosa*) are easily linearly separable from the other two (*I. virginica*, *I. versicolor*). The latter two are themselves not linearly separable from each other, and this is the subset provided to the students for analysis.

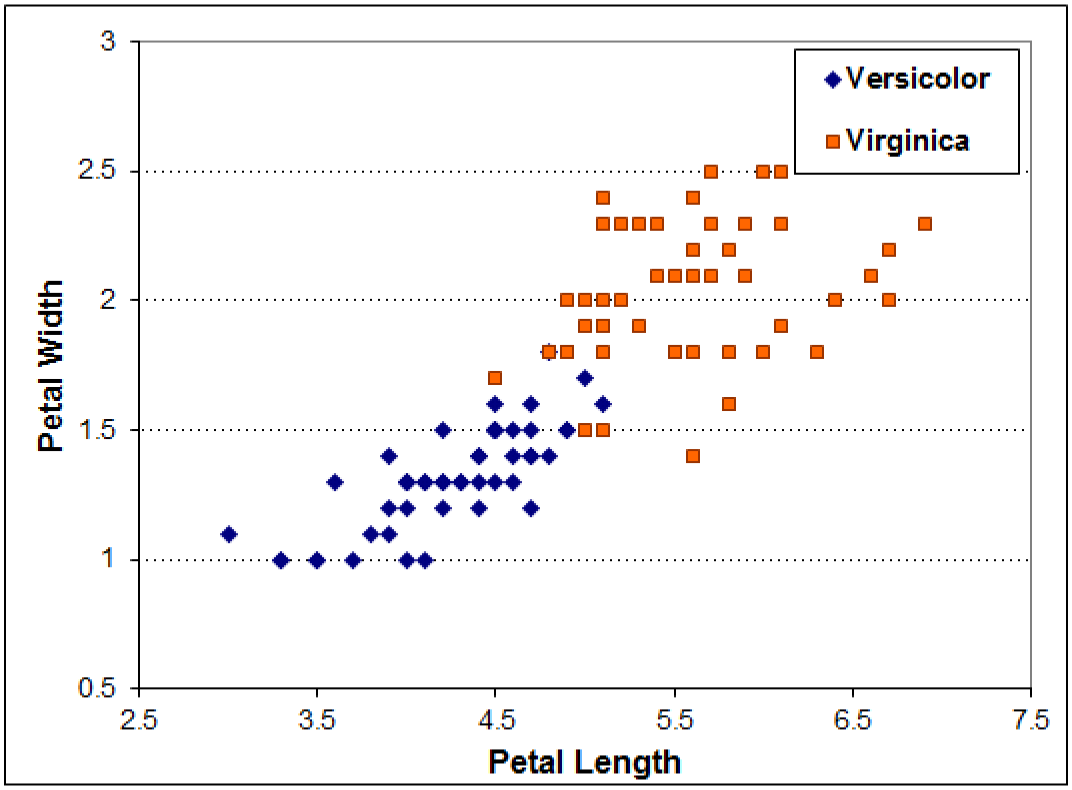

Figure 2: Illustration of the Iris exercise dataset

Fisher's original dataset collects four attributes for each iris instance, so while *Petal-length x Petal-width* is not the only possible 2D projection of the dataset, no 2D projection will be found that ensures linear separability between the two species. Students are given the choice to use any two of the four attributes provided. This choice brings complexity to the data analysis procedure, as students must determine a satisfactory resolution for a convergence process that may never converge on its own, but settle instead upon a minimum error solution. Many different lines, of positive or negative slope, may be determined that provide equivalent minimum error solutions, giving each student the opportunity to think critically as they provide rationales for their own solution, which may appear radically different from those of their peers. This multiplicity of valid solutions also provokes discussion between groups of students as they compare, evaluate and confirm each other's solutions as correct or incorrect.

3.3 Student Choice Dataset

Once students are familiar with their models and how to deal with complex datasets, as well as understanding what makes a dataset compatible with the Perceptron approach, they are encouraged to choose datasets of their own. Many different extensions to the base project may occur here, as the perceptron model readily extends to higher dimensional problems if students so desire. Additionally, the use of *m-1* perceptron models will allow to distinguish among *m* categories of points. Students' creativity and self-expression may be given full rein as they use datasets they care about to engage with their new modeling tool. For students requiring suggestions of where to look for datasets of interest, repositories such as that maintained at the UCI Machine Learning Repository [19] may be provided. A helpful feature of the UCI repository is that it specifies those datasets that are particularly fit for classification tasks, and therefore for use within this open-ended portion of the project.

4. Potential Outcomes of Data Analysis with Perceptrons

In the course of these exercises, students learn some basic tenets of data analysis, in addition to computer science best practices. As an example: randomizing the order of the dataset with each new pass through leads to significantly faster convergence onto either a solution, or a minimum error state. Without randomization between passes, the entirely deterministic algorithm takes a particular static path through the problem-space, which may or may not be efficient for the state in which the Perceptron's *p-vector* finds itself. While the difference in performance may not be noticeable on smaller problems such as the Gradebook exercise, order randomization can make a radical difference in running time for larger, more complex problems.

Since the basic model approaches its solution completely deterministically, one model may be used as a validation of another model's results should they both begin in the same initial state. Different weather forecast models, for example, are employed to validate each other's results and incidentally serve as a baseline performance for the other to compare against. We have observed a pair of students working on the same dataset use one model's intermediate results to find a bug in the other's, and in the process find improvements to be made to the former.

From the students' perspective, we have received anecdotal reports suggesting increased engagement due to factors ranging from a math major's appreciation of implementing a model related to her field, to appreciation of choosing one's own dataset to explore, to appreciation of actively being encouraged to work with others to accomplish one's goals with a dataset. Given continued perceptions of the CS major as being asocial and uninviting, with homework projects concerning material perceived as irrelevant to non-majors [5], this feedback seems to support Cunningham's view of general project properties that might appeal to those not otherwise being retained within the major.

Whereas we hope to encourage healthy collaboration in lab assignments as well as class assignments in general, this project appears to engender more collaboration than most. Due to the open-ended nature of the final phase of the project, the introduction of a wide variety of datasets into a population that has been indoctrinated with this new classification technique allows for experimentation with a range of approaches to "solve" a particular dataset. We have found that allowing two person teams to work on this project tends to improve the final results (as expressed not only by final grades but also student understanding and appreciation for the material) for individual participants, as compared to students working on it individually. Moreover this collaborative spirit tends to extend outside the local team to other groups whose potentially improved results will have no effect on the collaborator's scores. The sense of having mastered a novel technique would appear to have increased the engagement of at least some of the participants, and in the process relationships are formed with a wider number of classmates than an individual might otherwise have been comfortable with. This in turn can lead to a greater sense of *esprit de corps*, or a sense of belonging, which may affect a student's decision to ultimately remain with a major or not, thereby potentially impacting a program's retention numbers.

While many of these statements of potential benefit rest solely on the evidence of a single instructor's observations over several iterations of this project over the years, they find some additional anecdotal support in independent observations from that instructor's colleagues of increased social cohesiveness in recent graduating cohorts – though it is important to note that this project takes place among a series of experiences students have in collaborative settings throughout the initial courses of the major at our institution. Although it would not make sense to ascribe this increased cohesiveness to any one project or assignment, it might not be unreasonable to expect to make more anecdotal observations of this type were the percentage of projects in one's program increased that displayed elements encouraging engagement, self-expression and collaboration.

A preliminary Qualtrics [15] survey of fourteen respondents from a recent CS2 cohort was taken to measure student perceptions of the above-mentioned elements. Although it suggested some potentially interesting differences between the perceptions women and men had of the project, especially in the areas of student choice and collaboration, these observations remain tempered by the fact that the sample size of the women respondents was often only two (compared to eleven for men), and for most responses these two groups' scores were therefore not significantly different.

5. Summary

Computer Science can be an intense, demanding discipline, and small CS departments may struggle to retain students in the program over the full four year span of a typical major. Women remain underrepresented in computer science, despite recruitment and retention efforts. Disinterest and discouragement with the major have been attributed to stereotypes of CS as uncreative, asocial, inapplicable and intimidating [5] [16].

The design of the Perceptron project was a direct attempt to incorporate these issues, and our preliminary evidence suggests that we were successful at engaging students with the assignment. Overall, students in the CS2 class have expressed above average to high levels of interest and engagement with this project, as well as appreciation for its opportunities for collaboration and self-expression. We plan to continue monitoring student sentiments during future assignments to ascertain to what degree they incorporate the positive properties that we would like to emphasize.

References

[1] Anderson, R. E., Ernst, M. D., Ordóñez, R., Pham, P., & Tribelhorn, B. A data programming CS1 course. In *Proceedings of the 46th ACM Technical Symposium on Computer Science Education*, ACM, 2015, 150-155.
[2] Barr, V. Disciplinary thinking, computational doing: promoting interdisciplinary computing while transforming computer science enrollments. *ACM Inroads*, *7*(2), 2016, 48-57.
[3] Brundiers, K., & Wiek, A. Educating students in real-world sustainability research: vision and implementation. *Innovative Higher Education*, 36(2), 2011, 107-124.
[4] Cunningham, S. Computer graphics in context: an approach to a first course in computer graphics. In *ACM SIGGRAPH ASIA 2008 educators programme*, 2008, p.1.
[5] Fisher, A., & Margolis, J. Unlocking the clubhouse: the Carnegie Mellon experience. *ACM SIGCSE Bulletin*, *34*(2), 2002, 79-83.
[6] Fisher, R. A. The use of multiple measurements in taxonomic problems. *Annals of eugenics*, *7*(2), 1936, 179-188.
[7] Guttag, J. V. *Introduction to computation and programming using Python (2nd Ed)*. MIT Press, 2016.
[8] Kumar, D., Blank, D. S., Balch, T. R., O'Hara, K. J., Guzdial, M., & Tansley, S. Engaging Computing Students with AI and Robotics. In *AAAI Spring Symposium: Using AI to Motivate Greater Participation in Computer Science*, 2016, 55-60.
[9] Maxwell, B. and Taylor, S.. Comparing Outcomes Across Different Contexts in CS1. In *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education*, SIGCSE '17, New York, NY, USA, 2017, 399-403.
[10] Miller, B. N., & Ranum, D. L. *Problem Solving with Algorithms and Data Structures Using Python (2nd Ed)*. Franklin, Beedle & Associates Inc., 2011
[11] Miller, B. N., & Ranum, D. L. *Python programming in context*. Jones & Bartlett Publishers, 2013.
[12] Moody, G. Snowden leaks reveal GCHQ and NSA snooped on in-flight mobile calls, retrieved March 5, 2017 from *arstechnica.com*: https://arstechnica.co.uk/tech-policy/2016/12/snowden-leak-gchq-nsa-spy-in-flight-mobile-calls/
[13] Murtagh, T. P. Weaving CS into CS1: a doubly depth-first approach. In *ACM SIGCSE Bulletin*, 39(1), 2007, 336-340.
[14] Patterson, S. M. "OK Facebook"—Why stop at assistants? Facebook has grander ambitions for modern AI, retrieved March 5, 2017 from *arstechnica.com*: https://arstechnica.com/information-technology/2017/01/the-origins-and-future-of-artificial-intelligence-at-facebook/
[15] Qualtrics. Retrieved March 5, 2017 from *qualtrics.com*: https://www.qualtrics.com/
[16] Rich, L., Perry, H., & Guzdial, M. A CS1 course designed to address interests of women. In *ACM SIGCSE Bulletin* 36(1), 2004, 190-194.
[17] Rosenblatt, F. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological review*, *65*(6), 1958, p.386.

[18] Straumsheim, C. Computer Science as Liberal Arts 'Enabler', retrieved March 5, 2017 from *insidehighered.com*: https://www.insidehighered.com/news/2016/02/23/liberal-arts-colleges-explore-interdisciplinary-pathways-computer-science
[19] UCI Machine Learning Repository. Retrieved March 5, 2017 from UC Irvine: http://archive.ics.uci.edu/ml/datasets.html
[20] Villarreal, E. E., & Butler, D. Giving computer science students real-world experience. In *ACM SIGCSE Bulletin* 30(1), 1998, 40-44.