Impact of Final Grades: Neural Network Reveals Attendance is Strong Indication

Jessica Albright

Master of Business Administration – Data Analytics Certificate
Missouri State University

Neural Networks for Computer Learning

Abstract

What if professors would accurately predict a student's final grade based on inputs that are already made readily available to them at the midterm point of the semester? Could one of the strongest predictive inputs, which influences the student's grade as a measure of over one full letter (1.5 points on a 5 points scale), be something that the student can easily control?

Environment of Legal Business (also referred to as Law231 at Missouri State University is one of the required core business class for students admitted to the College of Business. While this class is challenging, Professor Phillips strives to find the most efficient ways to reach her students and ensure they are receiving the highest level of education from that course. The research conducted for this paper included gathering data from 939 students from Professor Phillips recent past five semesters of teaching Law231 at Missouri State University. Inputs from student's information included their gender, class status (freshmen thru graduate student), midterm grade, percent of classes attended, where they sit in class each day, if they were able to use electronics in class, their class time, days of the week class was held, semester, and year. Using NNSOA, we can predict a student's final grade with on average a 50% accuracy using these inputs, and one input in particular has an impact on a student's grade of 1.5 points on a 5-point scale.

1. Introduction

   The importance of this research will span farther than the Professors of Law231 at Missouri State University campus. The inputs used in this research can easily be interpreted to different courses or used as a tool to determine which inputs had the most significant impact on the student's overall grade. By determining this input, the professors, and students, can better allocate their time and resources to make the educational process more efficient.

   To understand how the inputs where identified and analyzed, it's important to note the structure of the data. For the purpose of this paper, percentage attendance was calculated at the midterm part of the semester, during the Spring semester this was at the time of March 15th, and for the Fall semester it was calculated at October 15th. Since classes meet more or less than other throughout the semester (for example, a Monday night class only meets one day a week for three hours., where as a Monday, Wednesday, Friday class meets three days a week for one hour) the attendance was calculated by determining the students number of days attended at the midterm point, and divided by number of days that class met during that timeframe. Using this attendance percentage allowed all the classes to be analyzed without altering that input for different runs. Additionally, other inputs included their gender (as inputted by the Office of the registrar during their time of registration), the term (fall or spring semester), day of the week classes were held, whether they had permission to use electronics in class (if they sat in the front row), their class status (freshmen through graduate level), their midterm grade, the row they sat in the classroom (which was determined by the student the first week of classes), and class time.

   While Artificial Computer Learning and Neural Networks might sound new age and a future trend to some, it's importance to realize that Artificial Neural Network have been around for nearly as long as computers have. The three articles identified as background information for the introduction are not from this year, and one is source regarding the prediction of students GPA was published over twenty years ago.

Artificial Neural Networks is not a “new trend” nor should its validity be question by being new and experimentive. Neural networks are widely used for researchers, experimenters to prove the certainty of inputs being a valid predictor of outputs, using past data trained and analyzed for future inputs to be recorded.

1.1 Neural Networks Predictive of Student Success in Nigeria

Victor Oladokun conducted research to identify why there was a significant number of graduates from Nigerian universities with low levels of educational qualifications. Through a study using Acritical Neural Networks to predict the likely performance of a candidate being consider for admission to the university, he found that part of the problem was the inadequacies in the National University Admission Examination System. The research identified various inputs that influence a student’s performance, such as students scores, age on admission, parental background, gender, types/region of the secondary school they attended, and exam score. A model based on the Multilayer Perceptron Topology was developed and trained using data spanning five generations of graduates from an Engineering Department of University of Ibadan, Nigeria's first University. Test data evaluation shows that the ANN model is able to correctly predict the performance of more than 70% of prospective students. (Oladokum, 2000)

1.2 Predicting Students Grade Points Averages using Neural Networks

In the research conducted by Goor, Nagin, and Szcypula they discuss how a student’s GPA in a professional school can be predicted to an admissions committee by using neural networks.(Gorr, Nagin, & Szcypula, 1994)

1.3 Determining the fault of ineffectiveness in education in Bangladesh

The research Rahman conducted in Bangladesh was designed to predict why students were failing to succeed after what was considered proper education. Despite dropout rates declining in primary and secondary schools, the problem continued to rise in undergraduate education with soaring failure rates. The research analyzed socioeconomical and psychological factors that could have an impact on the students academic performances. By gathering data from 423 students, and designing a predictive model with algorithms that insectioned the relationship between the three criteria's in their research, they found eight strong predictors of students influences that impacted their academics. (Rahman, 2017)

2. Neural Networks

This research attempts to predict a student's final grade in the class, given variables that are easily obtained at the midterm part of the semester, including midterm grade, class attendance, and location of seat in class, gender, class status, and time of class. In order to do this, we will utilize a neural network (NN), which has been found to be successful prediction tool for business problems as well as many other fields, such as technology, medicine, agriculture, engineering, and education. A simple Internet search using ArticleFirst produced over 11,000 articles on NNs. The NN used in this study incorporates the Neural Network Simultaneous Optimization Algorithm (NNSOA), a modified genetic algorithm, as its search technique. The Genetic Algorithm (GA) that was used as the base algorithm for the NNSOA, has been shown in comparisons with gradient search algorithms (variations of backpropagation) to outperform them in computer-generated problems as well as several real-world classification problems. (Sexton, Tunc , & JND, 2000) (Sexton & Dorsey, Reliable classification using neural networks: a genetic algorithm and backpropagation comparison, 2000) (Sexton, Johnson, & Dorsey, Toward a global optimum for neural networks: a comparison of the genetic algorithm, 1995) (Sexton & Gupta, Comparative evaluation of genetic algorithm and backpropagation for training neural networks., 2000) (Sexton & Gupta, Comparing backpropagation with a genetic algorithm for neural network training, 1999)

Modifications of the GA were made improving the algorithms ability to generalize to data in which it was not trained as well as giving it the ability to recognize relevant versus irrelevant input variables (NNSOA). This has the advantage of giving the researcher or manager additional information about the problem itself. In our case, we will be able to see which of the inputs we included in our data set that are actually helping predict the student's final grade. By doing so, we are one step closer to helping students and instructors see where a greater impact is being made in the students' efforts for academics. The NNSOA was shown to outperform the GA, 25 in which it was based, as well as several backpropagation variations. (Sexton, Dorsey, & Sikander, Simultaneous optimization of neural network function and architecture algorithm, 2002) Also, included in the NNSOA is the automatic determination of the optimal number of hidden nodes to include in the NN architecture. This feature alone saved us considerable time and effort from trial-and-error techniques usually employed by other NN programs to find optimal architectures.

2.1. Generalization

By using a search algorithm that identifies unneeded weights in a solution, this solution can then be applied to out-of-sample data with confidence that additional error cannot be introduced in the estimates. In current NN practices, every available input is included into the model that has the possibility of contributing to the prediction. While this method can result in fairly good models it has some obvious limitations. During the training process, if the connections (or weights) are not actually needed for prediction, the NN is required by its derivative nature to find nonzero weights that will essentially zero each other out for the training data. However, once this solution is applied to data that it has not seen in the training data set (out-of-sample), the unneeded weights are likely to not zero each other out and therefore add additional error to the estimate. This is a generalization problem. By using a search algorithm that is not based on derivatives, such as the NNSOA, we are allowed to have weights in our model that are hard zeros as well as modifying the objective function to add a penalty for every weight that is not a hard zero. In actuality, weights are really never removed only replaced with hard zeros, which in effect, removes them from the solution. In doing so, when applied to any data, whether training or testing there can be no net effect on the estimates. With the NNSOA, weights can be added and removed automatically at each stage of the optimization process. As weights are added or eliminated during the optimization process, discontinuities are introduced into the objective function. This precludes using search algorithms that require a differentiable objective function and, in fact, preclude most standard hill climbing algorithms. Previous studies have explored using gradient techniques that allow some of the weights to decay to zero 9 or which reduce the size of the network during training. (Baum & Haussler, 1989) (Burkitt, 1991) (Fogel, 1995) (Kamimura, 1993) (Prechelt, 1994) (Cottrell, Girard, Girard, & Mangeas, 1993) (Chan and Tong, 1986).

These methods were found to have limited usefulness. Another alternative is to remove active weights or hidden nodes and then evaluate the impact. This method of weight reduction is basically trial-and-error and requires the user to retrain after every modification to the network. The NNSOA on the other hand is based on the genetic algorithm, which does not require a differentiable objective function and can handle discontinuities such as penalty value for each nonzero weight in the solution. The improvement of generalization has been the topic of much research. (Burkitt, 1991) (Fogel, 1995) (Kamimura, 1993) (Prechelt, 1994) (Dorsey, Johnson, & Mayer, A Genetic Alogrithm for the training of the feed forward neural networks, 1994) (Drucker & LeCun, 1992) (Karmin, 1990) (Kruschke, 1989) (Dorsey, Johnson, & Van Boening, The use of artificial neural networks for estimation of decision surfaces in first price sealed bid auctions, 1994) (Chan and Tong, 1986; Sexton et al., 1999).

2.2. Identification of Relevant Inputs

An additional benefit of being able to set unneeded weights to zero is the identification of relevant inputs in the NN model. After a solution has been found, an examination of these weights can be conducted in order to determine if any of the input variables have all of its weights set to zero. If a particular input has all of its input weights set to zero, we can conclude that this variable is irrelevant to the NN model since it will have no effect on the estimate. This is not to say the input has no relevant information. It just means that the NN found for this particular solution, to have no value for this input in helping with the prediction. This could mean two things. First, the variable has no value in predicting the output. In this case, if several different runs were conducted (changing the random seed to initialize the network's starting points) it is likely that this input would be identified as irrelevant every time.

The second case is not as clear as to the inputs relevancy. After several runs the input may or may not be included in the final solution. In this case it is likely and makes intuitive sense that the information contained in this input may be duplicated in other input variables. For example, let's say Inputs 1 and 2 have some of the same information contained in them. In one NN run, Input 1 is eliminated from the model. However, in the second run Input 2 is eliminated. A third run might include both variables as relevant, where it captured some of the relevant information from both variables. In either case, more information is gathered by this method, which gives the researcher or instructor a better understanding of the problem. By determining the relevant inputs in the model, a professor or instructor can now have a better understanding of the problem and will be better equipped in making decisions. Section 3 describes the NNSOA. This is followed by the Monte 35 Carlo study, results and conclusions.

3. The neural network simultaneous optimization algorithm

The following is a simple outline of the NNSOA. The NNSOA is used only to search for the input weights. Prior research has found that using ordinary least squares (OLS) for determining the output weights is more efficient and effective. (Sexton, Johnson, & Dorsey, Toward a global optimum for neural networks: a comparison of the genetic algorithm, 1995) (Dorsey, Johnson, & Mayer, A Genetic Alogrithm for the training of the feed forward neural networks, 1994) (Dorsey, Johnson, & Van Boening, The use of artificial neural networks for estimation of decision surfaces in first price sealed bid auctions, 1994) (Dorsey, Johnson, & Mayer, A Genetic Alogrithm for the training of the feed forward neural networks, 1994) (Sexton, Sriram, & Etheridge, Improving decision effectiveness of artificial neural networks: a modified genetic algorithm approach, 2003)

A formal description of the basic GA algorithm can be found in *Evolution of Dynamic reconfigurable neural networks: energy surface optimality using genetic algorithms*. (Dorsey & Johnson, Evolution of dynamic reconfigurable neural networks: energy surface optimality using genetic algorithems, 1997)

Unlike backpropagation (BP), which moves from one point to another based on gradient information, the NNSOA simultaneously searches in many directions, which enhances the probability of finding the global optimum. The following is an outline of the NNSOA used in this study.

3.1. The NNSOA outline

3.1.1. Initialization

A population of 12 solutions will be created by drawing random real values from a uniform distribution [−1, 1] for input weights. This will happen only once during the training process. The output weights are determined by OLS.

3.1.2. Evaluation

Each member of the current population is evaluated by an objective function based on their sum-of squared error (SSE) value in order to assign each solution a probability for being redrawn in the next generation. In order to search for a parsimonious solution, a penalty value is added to the SSE for each nonzero weight (or active connection). The following equation shows the objective function used in this study:

$$\text{Min } E = \sum_{i-1}^{N}(O_i - \hat{O}_i)^2 + C\sqrt{\frac{\sum_{i=1}^{N}(O_i - \hat{O}_i)^2}{N}}$$

Here N is the number of observations in the data set, O the observed value of the dependent variable, Oˆ the NN estimate, and C the number of nonzero weights in the network. The penalty for keeping an additional weight varies during the search and is equal to the current value of the Root Mean Squared Error (RMSE). Based on this objective function each of the 12 solutions in the population is evaluated. The probability of being drawn in the next generation is calculated by dividing the distance of the current solution's objective value from the worst objective value in the generation by the sum of all distances in the current generation.

3.1.3. Reproduction

Selecting solutions from the current population based on their assigned probability creates a mating pool of 12 solutions. This is repeated until the entire new generation, containing 12 solutions, is drawn. This new generation only contains solutions that were in the previous solutions. The only difference in the new generation and the old generation is that some of the solutions (the ones with higher probabilities) may appear more than once and the poorer solutions (the ones with lower probabilities) may not appear at all.

3.1.4. Crossover

Once reproduction occurs giving us some combination of solutions from the previous generation, the 12 solutions are then randomly paired constructing 6 sets of parent solutions. A point is randomly selected for each pair of solutions in which the parent solutions will switch the weights that are above that point, generating 12 new solutions or the next generation.

3.1.5. Mutation

For each weight in the population a random number is drawn, if the random value is less than .05, the weight will be replaced by a randomly drawn value in the entire weight space. By doing this, the entire weight space is globally searched, thus enhancing the algorithm's ability to find global solutions or at least the global valley.

For each weight in a generation a random number is drawn, if the random value is less than .05, a hard zero will replace the weight. By doing this, unneeded weights are identified as the search continues for 9 the optimum solution. After this operator is performed, this new generation of 12 solutions begins again with evaluation and the cycle continues until it reaches 70% of the maximum set of generations.

3.1.6. Convergence enhancement

Once 70% of the maximum set of generations has been completed, the best solution so far replaces all the strings in the current generation. Each weight in the population of strings is varied by a small random amount. These random amounts decrease to an arbitrarily small amount as the number of generations increase to its set maximum.

3.1.7. Termination

The algorithm will terminate on a user specified number of generations.

4. Hidden node search

The number of hidden nodes included in each NN is automatically determined in the following manner. Each NN begins with 1 hidden node and trained for a user-defined set of generations or MAXHID. After every MAXHID generations, the best solution at that point is saved as the BEST solution and an additional hidden node is included into the NN architecture. The NN is reinitialized by using a different random seed for drawing the initial weights and trained again for MAXHID generations. The BEST solution is also included in this new generation by replacing the first solution with its weights. Since an additional hidden node creates more weights than is found in the BEST solution, these weights will be set to hard zeros. This way, we keep what we have learned so far from previous generations. Upon completion of this training the best solution for this architecture is compared with the BEST solution. If this solution is better than the BEST solution, it now becomes the BEST solution and is saved for future evaluation. This process continues until a hidden node addition finds no solution better than the BEST solution. Once this occurs, the BEST solution and its corresponding architecture is trained with an additional user defined number of generations or MAXGEN, which completes the training process. Although two solutions could achieve the same value for the objective function, they may differ in their architecture. Dorsey et al. demonstrated that the NN could have a variety of structures that will reduce to the same equivalent structure. (Dorsey, Johnson, &

Mayer, A Genetic Alogrithm for the training of the feed forward neural networks, 1994) (Drucker & LeCun, 1992) (Karmin, 1990) (Kruschke, 1989)

5. Classification problem and experiment

The objective of this study is to determine the effectiveness of using a NNSOA trained neural network to predict a student's final grade in a Law231 course, and identify variables that can be controlled by the instructor or student to influence a better grade in the course.

Data was collected from 960 students total, however after omitting students who dropped the class at the midterm point, or received a grade other than A, B, C, D, or F ; (such as an IC, or W), we analyzed data on 939 observations. Variables examined included their gender (as recorded by the office of the registrar), their class status (freshmen through graduate level), their midterm letter grade, their percent attendance at the midterm point, the row they sat in classroom, whether or not they had the opportunity to use electronics in class, their class time, day of the week schedule, seasonal term, and year. Data was collected from the most recent five semesters (Spring 2016 through Spring 2018).

The total number of ten inputs were evaluated with the neural network and represented in the graphs that follow - one input has a significant influence on the output of the final grade, with a weight of 1.5, which translates into one letter grade and half of their final grade.

A ten-fold cross validation was conducted in order to add rigor to our study. We made 10 trainings and 10 corresponding test sets out of the 939 observations. We did this by first randomizing the order of the observations and then taking off the last 94 observations and saving them into a test file. The remaining 845 observations were saved into a training file. To make the next training and test files, we put the 44 test observations from the previous data set and put them at the top of the training observations from the previous data set. We took them off the last 94 observations and saved them as the second test file and also saving the remaining 845 observations as the second training file. We did this for 9 data sets and on the 10th data set we had to change the number in the training and testing sets because the total number of observations was not divisible by ten and we wanted to make sure that every observation appeared one time in the a test set. The last training and testing sets included 846 for the training observations and 93 for the testing observations.

6. Results

Once the 10 training sets finished training, we tested the solutions on their respective test sets. We calculated the hits rate of each grade (A,B,C,D,F) and found the sum of squared errors, mean of squared errors to calculate the root mean of squared errors. By compiling this per each run, for each respective grade hit, we could calculate the percent of times the neural network could accurately predict a student receiving that grade given all the inputs. In table one, it illustrates the calculations per each perspective grade and shows their standard deviation and average. The neural network can predict with accuracy of a student receiving an “A” final grade 47% of the time, a “B” final grade 70% of the time, a “C” grade 60% of the time, a “D” grade 33% of the time, and an “F” grade 43% of the time.

Table 1

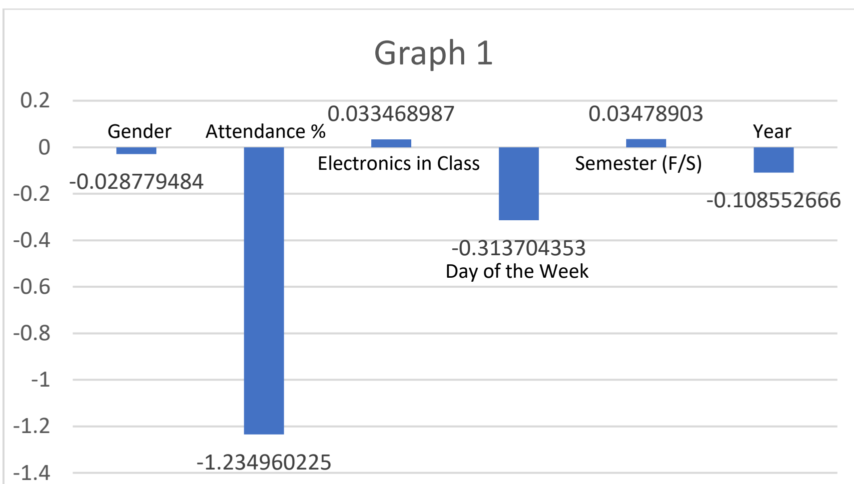
Runs	A	B	C	D	F
1	0.526316	0.682927	0.666667	0	0.125
2	0.47619	0.705882	0.5	0.857143	0.75
3	0.526316	0.806452	0.44	0.266667	0.25
4	0.5	0.552632	0.727273	0.111111	0.555556
5	0.4375	0.658537	0.8	0.444444	0.5
6	0.5	0.607143	0.545455	0.2	0.777778
7	0.4375	0.74359	0.6	0.222222	0.2
8	0.5	0.823529	0.677419	0.4	0.444444
9	0.388889	0.594595	0.678571	0.4	0.166667
10	0.411765	0.806452	0.413793	0.416667	0.5
STD	0.048491	0.096197	0.127286	0.235135	0.234923
AVG	0.470448	0.698174	0.604918	0.331825	0.426944

The ability to predict a student’s final grade was only one part of the study. The second part of the study is to determine which inputs have the most significant impact in influencing this final grade. This is important because by knowing the inputs with a significant influence, we can determine if the input can be controlled, or somewhat influenced, by the instructor or student.

The following graphs illustrate the severity of impacts an input has on the final results. Before examining the results, it's important to note two things:

1. The scale of the graph to indicate influence. Anything below 1 would mean that the input does not influence a student's grade by a full letter grade the scale is on 5 points, respective to letter grades.

2. The direction is not always intuitive. A negative number does not mean the input is negatively correlated with the outcome. The description with each chart will detail how the results should be read, due to coding with the neural network, the results seem counter-intuitive, and "A" is classified as a 1, "B" as a 2, "C" as a 3, "D" as a 4, and "F" as a 5.



Graph 1 examines the impact of six inputs, Gender, Attendance %, whether or not the student was able to use electronics in class, the day of the week classes were held, the fall or spring semester, and the year the inputs were gathered.

6.1 Gender was coded as either male or female as gathered information from the office of the registrar. Female was coded as 1, and Male was coded as 0, however with a -.0287 average the input is insignificant to predicting a student's final grade. It has an influence of less than 3% of one letter grade.

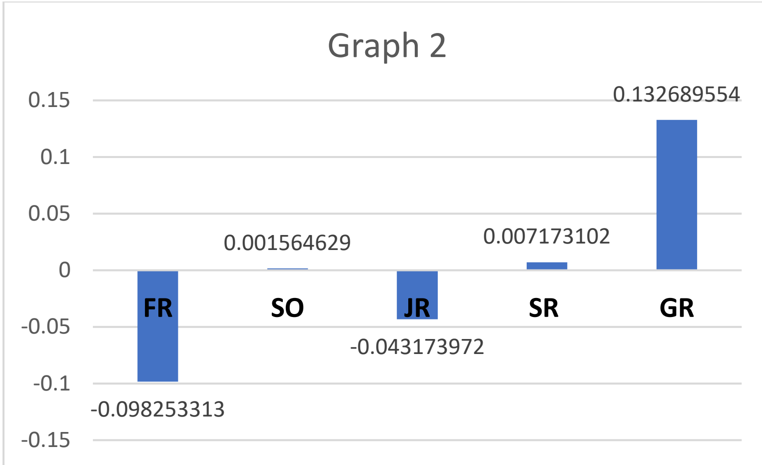
6.2 Attendance is the highest and most significant input to influence a student's final grade, as illustrated by the neural network. Attendance was gathered at the midterm point of each semester, and calculated as a percentage, since some classes meet more or less than other classes. The percentage was the number of days a student attended class, respective to the total number of days classes were held to the midterm point. Thus, a student with 100% attendance attended every class from the beginning of the semester till the midterm point. With over a -1.23 impact, this means that a student's letter grade will be influenced by over one full letter grade based on their attendance in the class. Recall that "A" is coded as 1, and "F" is coded as 5, with the other grades falling respective of their order, therefore when examining that attendance as -1.23 impact, it shows that as student's attendance decreases, their prediction of getting the higher coded grade (which again, is counter intuitive with the neural network, a "higher coded grade" would mean "lower grade", as 5 would be an F). Over one full letter grade is the highest single impact we have identified in this study, and will be discussed in the conclusion on the possible solutions both the students and the instructor can take to encourage students to attend class more often to increase their likeliness of achieving a higher final grade.

6.3 Electronics in class variable was identified by the students who sat in the first row of the classroom. Since the data was obtained through Professor Phillips class records, this input could be classified as she requires any student who uses an electronic in class to sit in the first row of seats, this could include a tablet or laptop for taking notes. Noted, not each student who sat in the front row actually used electronics in class, however the opportunity to use one was granted to those students, and this study does not conclude whether or not the student actually decided to use one in class, nor the frequency of when students used electronics in class. Given the average from the neural network, the opportunity to use electronics in class was on .033, which means it had less than a 3% impact of a full letter grade.

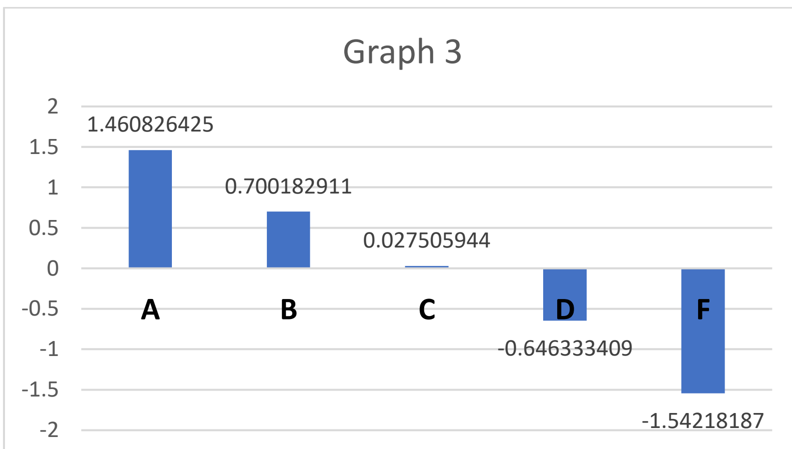
6.4 Day of the week input is the classification of which days the classes were held. This was coded either by Tuesday and Thursday classes as “-1”, Monday and Wednesday classes as “0” and classes that only met Wednesday evenings as “1” , but examining the results of this prediction, we can conclude that the days of the week classes meet is insignificant to the students grade, less than 30% of a full letter grade. The -.313 can be indication that as the classification gets greater, meaning 1 (Wednesday night classes) the grades increase (meaning getting closer to 1, with 1 representing A). Therefore, Wednesday night class students overall performed better than the other sections, and this needs to be considered when evaluating if this was indication of the neural network, or the student representation in that section, as will be discussed further when examining the results in Graph 5, illustrating class times.

6.5 Semester inputs is the term that students were enrolled in LAW231, either Fall or Spring Semester, with the coded classification being 0 and 1 respectively. While this variable does not play a significant impact in the student’s grade, less than 3%, it does show a small influence that spring semester results in higher grade classification (meaning lower grades). The fall semester students can predict better grades.

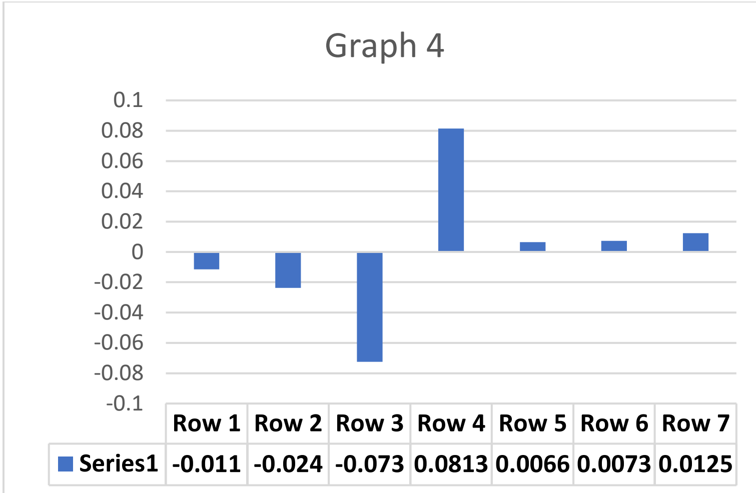
6.6 The year is a time variable that needed to be accounted for in the study, however as the results indicate had a very small influence in the overall prediction. The data was compiled from a relatively small timeframe, from 2016 thru 2018. The average impact on the year was less than 10% of a full letter grade for the students, however it indicates that since the years were coded, 2016 as -1, 2017 as 0, and 2018 as 1, that as the year becomes greater, the students classification number becomes greater, meaning the grades themselves are declining by a marginal amount.



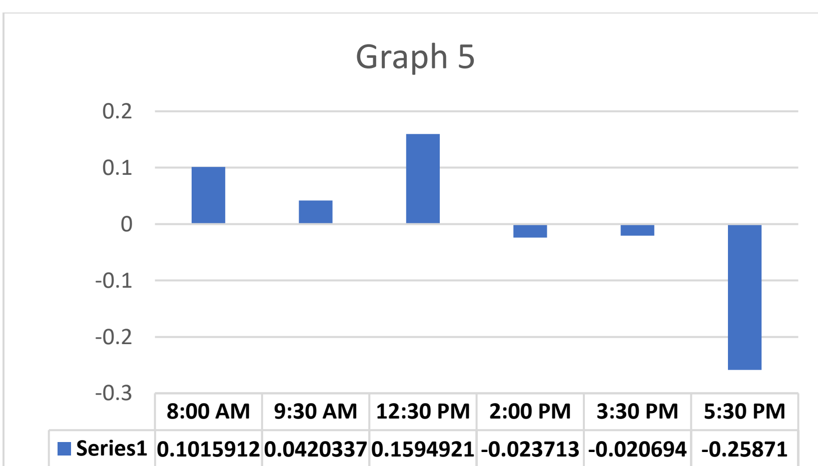
6.7 Graph 2 examines the influence a student class status has on predicting their final grade in the class. Not a single class status has a strong indication of having an impact on the outcome. It should be noted that the sample size of graduate students taking Law231 was relatively small, which created a larger average for this study.



6.8 Graph 3 examines how the midterm grade a student receives impact the outcome of the final grade they receive in the class. This part of the study is the least surprising, because as initiative would tell us, students who have higher grades at the midterm part of the semester will have higher grades at the final part of the semester, relatively. It's important to note, however, the significance of the inputs for the extremities. For example, having an "A" midterm score or an "F" as a midterm score will have a stronger prediction average in the student's final grade than students who receive a B, C or D.



6.9 Graph 4 examines the impact of where a student sits in class each date relative to their final grade. Since the study only included students from Professor Phillip's classes, it's important to note that her policy is that students may choose where they sit on the first day of classes, and from their selection on the first day a seating chart is made and that student sits in that same location for the classes throughout the semester. Since the data was compiled from classes that met in different classrooms, sometimes the row 7 was not occupied in classrooms where only 6 rows of seats were present. Not a single row indicates a significant prediction of the student's final grade, however, the pattern of the graph is an interesting correlation. Notice that rows 1, 2, and 3 have a negative average, which means that the student's final grades who sat in rows 1, 2 and 3 could be predicted as nearing a 1 classification, meaning a better grade.



6.8 Graph 5 examined if the time of class meeting would influence a student's final grade in the class. There was not a single class time that showed a significant impact, the largest impact that is illustrated is the 5:30pm evening class that meetings weekly.

7. Conclusion

The NNOSA is shown to preform exceedingly well for optimizing a neural network while simultaneously eliminating unnecessary weights in the neural network structure during the training process for our predictions of students final grades in a Law 231 class. By decreasing the structure of the neural network, generalization is likely to improve because the shortage of weights forces the algorithm to develop general rules to discriminate between the input patterns, instead of memorizing special cases. While the neural network could only predict final grades with a 50%, it does provide valuable insight to the factors that influence a student's grade. While attendance is known to have an influence on a student's grade, it will eye opening to students, and instructors alike, to understand how much of an impact it can have, a full letter grade. By understanding the significance of this input, professors can communicate this factor to students to encourage higher attendance in lecture. Likewise, by students understanding the impact of this input, they can be more motivated that their regular in attendance will correlate with predicted higher grades in the class overall.

Limitations of this study include areas where future research is warranted. Additional studies should be conducted in other classes, across the university to see if the subject, instructor, material content, or lecture style play a significant role in identifying the final grade in the course as predicted by the neural network.

Works Cited

Baum, E. B., & Haussler, D. (1989). What size net gives valid generalization? . *Neural Computation*, 151-160.

Burkitt, A. (1991). Optimisation of the architecture of feed-forward neural nets with hidden layers by unit elimination. *Complex Systems, 5*, 371-380.

Cottrell, M., Girard, B., Girard, Y., & Mangeas, M. (1993). Time series and neural network: a statistical method for weight elimination. *European Symposium on Artificial Neural Networks*, 157-164.

Dorsey, R. E., & Johnson, J. D. (1997). Evolution of dynamic reconfigurable neural networks: energy surface optimality using genetic algorithems. (E. W. : Levine DS, Ed.) *Optimality in biological and artificial networks*, 185-202.

Dorsey, R. E., Johnson, J. D., & Mayer, W. J. (1994). A Genetic Alogrithm for the training of the feed forward neural networks. . *Advances in artificial intelligence in economics, finance, and management, 1*, 93-111.

Dorsey, R. E., Johnson, J. D., & Van Boening, M. V. (1994). The use of artificial neural networks for estimation of decision surfaces in first price sealed bid auctions. *New Direction in Computational Economics.*, 19-40.

Drucker, H., & LeCun, Y. (1992). Improving generalisation performance using double backpropagation. *IEEE Transactions on Neural*, 991-997.

Fogel, D. (1995). A comparison of evolutionary programming and genetic algorithms on selected constrained optimization. *Simulatoin*, 399-406.

Gorr, W., Nagin, D., & Szcypula, J. (1994, June). Comparative study of artificial neural network and statistical models for predicting student grade point averages. *International Journal of Forecasting, 10*(1), 17-34. doi:https://doi.org/10.1016/0169-2070(94)90046-9

Kamimura, R. (1993). Internal representation with minimum entropy in recurrent neural networks minimizing entropy through inhibitory connection. *Network Computation in Neural Systems*, 423-440.

Karmin, E. (1990). A simple procedure for pruning back-propagation trained networks. *IEEE Transactions on Neural Networks*, 239-242.

Kruschke, J. (1989). Distributed bottlenecks for improved generalization in back-propagation networks. *International Journal of Neural Networks Research and Applications*, 187-193.

Oladokun, Victor & T Adebanjo, A & Sc, B & E Charles-Owaba, O. (2000). Predicting Students Academic Performance using Artificial Neural Network: A Case Study of an Engineering Course. The Pacific Journal of Science and Technology. 9.

Prechelt, L. (1994). A study of experimental evaluations of neural network learning algorithm: current research practice,. *Fakultat fur Informatik, Universitat Karlsruhe*.

Rahman, R. M. (2017, April). An intelligent system to predict academic performance based on different factors during adolescence. *Journal of Information and Telecommunication , 1*(2), 155-175. doi:https://doi.org/10.1080/24751839.2017.1323488

Sexton, R. S., & Dorsey, R. E. (2000). Reliable classification using neural networks: a genetic algorithm and backpropagation comparison. *Decision Support Systems, 30*, 11-22.

Sexton, R. S., & Gupta, J. D. (1999). Comparing backpropagation with a genetic algorithm for neural network training. *OMEGA The International Journal of Management Science, 27*, 679-684.

Sexton, R. S., & Gupta, J. D. (2000). Comparative evaluation of genetic algorithm and backpropagation for training neural networks. *Information Sciences, 129*, 45-59.

Sexton, R. S., Dorsey, R. E., & Sikander, N. A. (2002). Simultaneous optimization of neural network function and architecture algorithm. *Decision Support Systems*, 1034.

Sexton, R. S., Johnson, J. D., & Dorsey, R. E. (1995). Toward a global optimum for neural networks: a comparison of the genetic algorithm. *Decision Support Systems, 22*, 171-185.

Sexton, R. S., Sriram, R. S., & Etheridge, H. (2003). Improving decision effectiveness of artificial neural networks: a modified genetic algorithm approach. *Decision Sciences, 3*, 421-442.

Sexton, R. S., Tunc , E. A., & JND, G. (2000). Selecting scheduling heuristics using neural networks. *INFORMS Journal on Computing, 12*(2), 150-162.