Arun Agarwal

Principles of Data Science, CIS 3715

April 11th, 2021

Progress Report II

Deep Analysis and Prediction of the Effects of Various Factors on the Performance of Students on Examinations

Project Summary/Overview:

The purpose of an examination is to understand the ability and learning of a student; thus, through careful analysis of patterns between parental level of education, gender, race, reading scores, writing scores, math scores, and other factors from the Royce Kimmons Kaggle dataset named “StudentPerformance.csv, one can understand which aspects have the largest impact on test outcomes as well as what are the best ways to improve student scores for the future. Furthermore, the improvement of test scores would benefit the economy at a national level by creating equality, leading to major GDP growth, and building a modern society. Correlated with improved test scores is a stronger education system, which for the learner, creates more employment opportunities, secures a higher income, develops problem-solving skills, provides a prosperous and happy life, and educates them to give back to the community.

Prediction of student academic performance in mathematics, reading, and writing based on various demographic and socioeconomic statistics can be performed through creation of various data science models such as linear regression, logistic regression, and k-NN. While the dataset includes this information for 1000 students, not all useful pieces of data, such as hours spent studying, are provided, so I can only perform analysis on the effects these features have on performance. A model with good RMSE, MSE and MSAE scores signifies the model predicts student’s performance well, making the data useful in identifying methods to improve student performance for the future. For each model, I will use different hyperparameters and preprocessing techniques to optimize performance. Various types of graphs and plots will also be utilized for exploratory data analysis. The early detection of students who are vulnerable to suffering academic failure (through use of these models) can subsequently be used to design new teaching/mentoring strategies for an overall strengthening education system and society.

Progress Report (What has been done):

The beginning steps of the proposed timeline in the project proposal were as follows. For week 1, I was to write the project proposal, import the dataset, import required libraries, complete data preprocessing (including cleaning the data, making graphs to understand the data, label encoding, and splitting the dataset), and exploratory data analysis. All of this and more was completed by the end of Week 1 for the project, taking a total of approximately 12 hours, which is 5 hours longer than the proposed time. I spent a larger amount of time for that week because both the project proposal and progress report I had to be completed for week 1. Furthermore, I did more than what was proposed for that week, as will be explained in detail later, and was very detailed in the writing of my notebook, so this added a significant amount to the time. Also, all the steps outlined in the previous Project Proposal I for Week II were completed (with added detail) as follows. Since in the previous week I completed label encoding and splitting the dataset into training and test sets, I started this week by starting and finishing the linear regression and ridge regression models and their analysis (written in the Jupyter Notebook). Model metrics were written, and scores for MSE, MAE, RMSE, and R-squared were coded and outputted in the notebook for both the training set and test set. Model performance was then analyzed by the creation of numerous graphs and plots of the outputted data. Next, I built a logistic regression model using k-fold cross validation. I tuned the hyperparameter for the regularization term (I kept the code in for all tested hyperparameters to demonstrate the varying performance with each one). I then evaluated the accuracy, precision, recall, and F1 scores of the classifications for each preprocessing technique for an understanding of performance. Finally, I created an extra linear regression model, called a linear least square regression model, to understand the strength of the relationship between various provided factors and exam scores. Residuals—the deviation of the fitted values from the actual values—needed to be created for this to determine if this model was good to predict the reading scores from the same individual’s writing scores. A coefficient of determination (R^2 value) and a plot of the two features graphed was also created and outputted. With completion of these models came all corresponding pattern and model performance analysis. All of this was completed by the end of Week 2 for the project, taking a total of approximately 14 hours, which is 2 hours longer than the proposed time. I spent a larger amount of time for that week because creating four models took longer than expected (especially the logistic regression model due to its stark differences and the outputting of all tested hyperparameters). Furthermore, I was very detailed in the writing of my notebook, so this added a significant amount to the time. In the project proposal, 10 steps were outlined in the procedure to reach full completion of the project, and the first three steps (each with multiple parts) were completed for week 1, steps four through seven (also containing multiple parts) were completed for week 2, and the remaining three steps were outlined out in the Jupyter Notebook. These last three steps (as well as any final touch-ups) are to be completed in the following two weeks, as will be discussed later.

Going into detail of previously completed work, the first thing I did (after finishing the project proposal) was find and download the dataset. The dataset is provided by Royce Kimmons and can be found on the Kaggle website under the Student Performance on Examinations project. I did a preliminary exploration of the features and size of the dataset, and then moved on to setting up the coding environment. For consistency, I utilized the same program that is used in the class’s labs, Jupyter Notebook, accessed through the Anaconda Navigator. Since I created my own environment to work with, I needed to import multiple libraries to be able to perform all analysis that I plan to on this project at some point. To know which libraries I needed to import, I reviewed our class examples (Slides and Labs 3-8) and previous work with this dataset, available from the Kaggle dataset. I also revisited the similar projects listed in the Project Proposal document. I reviewed the code and setup of these notebooks not only to understand the necessary libraries to import but also to get an understanding of how I should go about working on the project and what I can include. Once I learned all the libraries necessary to complete my project, I started the import process in the Anaconda Navigator. I installed plotly, lightgbm, py-xgboost, and seaborn into the environment; I also needed to install bayes\_opt, but I could not see any option to install this library into my environment. Therefore, I may decide to not perform bayes optimization in my project. This might work out to my favor as we have never learned about the optimization project in class, so I am not sure if I would be able to write the necessary code for it. Other libraries that I would use for my project that are already imported include numpy, pandas, matplotlib, sklearn, and pylab.

With the environment set up, the Jupyter Notebook software could be opened, and creation of the notebook began. A folder was created for the project, called ‘Final Project 3715’, with the csv dataset uploaded into it. Also added into the folder was a new notebook, which when opened was named ‘Agarwal—001—Final Project Student Performance on Examinations’. Next, due to my lack of experience with the intricacies of Jupyter Notebook, numerous things had to be looked up. For example, I had to learn how to make different headers, how to write plain text, how to write italics, how to make bullet points, how to organize the code well, how to reorder cells, how to add numbered points, how to create links in the document (for a table of contents), and how to do numerous shortcuts (delete cells, add cells above, add cells below, move cells, etc.). I learned this information through simple searches and through the help page in the software itself. Next, I added a title to the beginning of the document, my name, and the class name. Next, I created a cell called ‘Table of Contents’. When examining other projects with the dataset, I found that using a Table of Contents in a Jupyter Notebook is a great way for one to organize their dataset. Thus, as the project goes on, I will continue to add links to various areas in my projects for ease of access and organization in the notebook. As of that point, there was only a link to the ‘Goals’ tab in the document. I made more links this week, as will be discussed later. Next, I added sections called ‘Introduction’ and ‘Goal’ which, as their title states, introduces the project (explain what will be done, why it will be done, how it will be done, etc.) and describes the end-goals. Another section, named “Questions to Try to Answer,” lists all of the essential questions I will try to answer through the analysis I will perform on this dataset (8 questions are listed).

Next, the coding part of the notebook began with importing the libraries stated earlier. The imports are broken down in the notebook based on what they will be used for. After this, I made a section for reading in the csv file, similar to how we do it with every lab. I also called the head() function to print the first few rows to get a superficial understanding of the dataset and its features. Furthermore, I added a target column to the data called ‘average score’, which is simply the average of the math, reading, and writing scores. This will be useful for creating models in which having one overarching target column will be helpful. After this, I listed out the targets of the data (the scores) and the features (gender, race/ethnicity, parental level of education, lunch, and test preparation course). I next printed out the size of the dataset; seeing that the dataset has only 1000 points, I chose not to remove any outliers in hopes of preserving/holding on to as much of the data as possible. I also used the describe method to understand basic information about the numerical features such as the count, mean, std, min, and max and analyzed the meaning of these results. Moving on, I checked for missing values using isnull().sum(), which demonstrated that there are no missing values and no filling in would need to be completed. I also used the info() function to demonstrate which features were categorical and which numerical (listed earlier).

With preliminary data examination completed, I started the exploratory data analysis/data visualization section of the project. First, I visualized all numerical features with histogram plots to see their distributions and analyzed the results. I decided 50 bins was a decent amount to display the data well. I then created a pair plot as another way to visualize the numerical features. Next, I used the corr() function of Pandas and the heatmap plot to show the correlation between different numerical features. The high correlation between the reading and writing scores was analyzed in detail. I also visualized the numerical features with density plots. After data visualization of the numerical features, I examined the categorical features using a variety of graphs. By coding a method to create pie charts, I was able to make a pie chart for each categorical feature at the same time. These pie charts were able to demonstrate a lot of proportions as well as pictorially demonstrate the unique values in each feature. Further categorical feature analysis was performed by creating bar graphs to compare each feature’s unique values to the average score for those values. The information collected from these bar graphs was then analyzed. Next, I explicitly coded a pass/fail mark, where 50% or lower is fail and standard letter grades are coded for the rest of the percentages. Then, I was able to see, for each student, which student is passing, and which is failing for each examination score. These new features were added to a copy of the dataframe, named df\_extended because I will only need these columns a few times. Using the previously created pie chart method, I was then able to create pie charts for the newly added categorical features. Two more columns were added to the extended dataframe, including ‘Total Marks’ and ‘Percentage’ to later find the percentage of marks. With the created grading scale, I plotted the number of students who obtained each grade to get a general sense of how well student do on examinations. I also made a count plot for parental level of education broken down by student grade. Finally, away from the grading scale, I performed analysis based on gender by developing plot density graphs for the three scores based on gender. The results of the gender differences were then analyzed.

Moving on, as done in labs, I converted the categorical features to numerical features using Label Encoding. This had to be done for a total of five columns, and the info method was called on the dataset to confirm that the whole dataset is now numerical values. Next, I split the dataset with 80% of the data being in the training set, and the final 20% in the test set to start the linear regression process for week 2. For organization purposes, I also outlined some of the other things I would do in the notebook including training the linear regression model, evaluating the linear regression model, using ridge regression model to do prediction, building a support vector machine (SVM), building a logistic regression model, building a linear least squares regression model, performing principal component analysis (PCA), and doing k-NN classification, much of which has now been completed. Sections for the Conclusion and References were also included at the end of the notebook.

Now, looking to the beginning of Week 2, I had to begin by re-running every cell created in the notebook in Week 1 (a total of 63 cells), which took an excessive amount of time since the exploratory data analysis plots take a lot of time to compile. I am not sure why the cells did not save as being ran in the previous week. Once all the cells were re-run, I picked up where I left off. Since in the previous week I completed label encoding and splitting the dataset into training and test sets, I started this week by starting and finishing the linear regression and ridge regression models and their analysis (written in the Jupyter Notebook). To train the linear regression model to do prediction, I used min𝑤 1/𝑛‖𝑦−𝑋𝐰‖22 (apologies for poor formatting, it is hard to write this in Word) and outputted the learned model parameter w to see how the learned model fit the training set. I of course printed out the MAE, MSE, and RMSE scores for the training set and analyzed the meaning of their values. I then evaluated the linear regression model to see how well it generalized on the testing set. I performed a procedure very similar to that completed in Lab Assignment 3 and made sure to print out the prediction scores (MAE, MSE, and RMSE) for the testing set as well for analysis. To better understand the results, I coded and outputted a bar graph of the ground truth and prediction. From the plot, we could then see the learned model did well to generalize on the testing set. I used the results from this model to see what questions (of the eight listed at the beginning of the document) could be answered (partially or completely) from the model. Similar to linear regression, I then began and completed the ridge regression procedure. For ridge regression, I used min𝑤 1/𝑛‖𝑦−𝑋𝑤‖22+𝜆‖𝑤‖22 (again, sorry for poor formatting) instead of the formula stated for linear regression. I also used different lambda values to see their affect on the performance of the model on the testing set. All lambdas tested were included in separate cells in the notebook to clearly demonstrate the difference in performance (but this also ended up taking more time). Similar to the linear regression model, the prediction scores for the ridge regression model on the training set (and later the test set) were outputted and analyzed. One could see from the scores that the prediction values for the training set for MAE, MSE, and RMSE were almost identical for the values for Linear Regression, as we expect (since the same training and test sets are used). I also created a bar graph for ridge regression to pictorially see how well the learned model did to generalize on the test set (and it did do well). To compare the performance of the ridge regression and linear regression models, I used a percent difference calculations (Percent Difference = (|Value 1 – Value 2|)/([Value 1 + Value 2]/2) \* 100%). One could see that, when testing different lambda values, the performance of the ridge regression model on the testing set worsened as the lambda value increased. This is because if one's lambda value is too high, the model will be simple, but they run the risk of underfitting the data (it did not learn enough about the training data to make useful predictions). Lambda values becoming incrementally smaller than 0.1 were also tested. Once the lambda values became extremely small, the model became more complex and ran the risk of overfitting the data. Analysis was then tied to the performance of students on examinations. Other plots were also created to analyze different parts of the outputted data.

Next, I built a logistic regression model using k-fold cross validation similar to the procedure followed in Lab 5. While I used the same training and test sets created for the linear and ridge regression models, I did need to use a different formula to do classification () and needed to learn a different model parameter w. I tuned the hyperparameter for the regularization term (I kept the code in for all tested hyperparameters to demonstrate the varying performance with each one). With different hyperparameters, I got different model parameters w, resulting in different prediction performance. Thus, I used 5-fold cross-validation to select the best hyperparameter. Thus, I got the number of samples in the training and validation sets, shuffled the index of samples in the training validation set, split the index of the training validation set into five folds, made a list for the hyperparameters I would use, created training and validation sets for each regularization coefficient, built and trained the model for each coefficient, and outputted the accuracy score for each regularization coefficient (to make my selection for the best hyperparameter). From there, I retrained the model with the best hyperparameter on the training validation set. I then evaluated the accuracy, precision, recall, and F1 scores of the classifications for each preprocessing technique for an understanding of performance. I also used a bar plot to visualize the elements of the learned model parameter vector w. The coefficient values that were higher in the plot were able to fit the logistic regression model better then the others. Similar to the linear and ridge regression models, I used the results from the logistic model to see what questions (of the eight listed at the beginning of the document) could be answered (partially or completely).

Finally, I created an extra linear regression model, called a linear least square regression model, to understand the strength of the relationship between the reading and writing scores. It should be noted that this does not measure the slope. I first got the intercept and slope of the line using linear least squares to predict the reading scores. Residuals—the deviation of the fitted values from the actual values—needed to be created for this to determine if this model was good to predict the reading scores from the same individual’s writing scores. The actual equation of the line that I used in regression was ys = intercept + slope xs + residuals. To determine if it was good to predict the reading scores from the writing scores or without it, I printed the RMSE scores. I performed analysis on the outputted RMSE scores and concluded that it would be better off to use the writing scores to predict the reading scores than to predict the reading scores without the writing scores. A coefficient of determination (R^2 value) and a plot of the two features graphed was also created and outputted. The value of R^2 was more than 0.90, suggesting that the line is a statistically significant fit for the data. One can then use this line and its equation to predict the reading scores. Thus, I can answer one of my eight questions listed at the beginning of the Jupyter Notebook from the use of this model. The line was also plotted in a neater format for displaying purposes. With completion of all four of these models came all corresponding pattern and model performance analysis. I also tuned cells created in Week 1 with more text analysis from the exploratory data analysis and updated the Table of Contents with more links to sections of the document (specifically, to each of the models).

Progress Report (What has not been done):

As outlined in the previous paragraph, there are still multiple things to be done in the next two weeks. Through the creation of the above four models, I have decided that I would like to try to build a support vector machine (SVM) to see if I can create one and to see if I can learn any more information about the data than has already been learned from the exploratory data analysis and models. Next, I will conduct Principal Component Analysis (most likely the sklearn Neighborhood Component Analysis) to prepare for k-NN on the preprocessed dataset using the most effective preprocessing technique. Then, I will of course perform k-NN classification using sklearn.neighbors and plot the results. After all these models and plots have been made, I will compare their results and make general conclusions about which factors are interrelated to each other and to performance on examinations (written in the conclusion section of the notebook and in the final report). I will also add any final touch-ups to the coding section of the notebook (improve upon spacing, comments, descriptions, and layout of code and notebook). Throughout the process, I will also need to complete a lightning talk and submit a final report. Everything listed here should take approximately twelve to fifteen hours.

Progress Report (What will be done in the following week):

As outlined in the project proposal, there are multiple things that need to be completed in the following week. First, I will start, write, and finish the support vector machine (if I find that I can successfully do it), Principal Component Analysis, and k-NN classification as well as all corresponding components. These components are listed in the previous section. With completion of these models comes all corresponding pattern and model performance analysis. I will also need to complete my lightning talk preparation for my actual lightning talk sometime between April 20th and April 22nd. This should take approximately eight to nine hours to complete.