Synchronous e-Learning Effects on Math Performance of High School Students

Tina M. Kirk
Department of Computer Science and Electrical Engineering
Marshall University
Huntington, WV
kirk41@marshall.edu

I. Introduction

Web-based learning is becoming more commonplace in education with advances in technology and the opportunities they can afford to individuals in remote places. In the United States, K-12 students typically attend school in person, where a teacher or group of teachers monitors their engagement in various classes. Students can take a virtual class here and there, but it is not the norm. In the past two years, amid the Covid pandemic when attending in person was deemed a public health risk, school districts without many web-based courses went through a sort of trial by fire with the sudden need for them in *every* class *every* day. In all the school districts that abruptly shut down in-person school, levels of readiness to switch to virtual learning varied. In general, there were already many challenges associated with e-learning, such as student motivation and engagement [1] being at or near the top. Adding a global pandemic to the mix made for an exceptionally difficult situation.

The focus of this project is the mathematics performance of students from a high school in a district which will be referred to as District X, USA. Shortly after the start of the pandemic in mid-semester, students and teachers were abruptly sent home with no uniform direction on how to proceed. For the next two and a half months, teachers struggled to find ways to record, write, send, and receive lessons and assignments. Everyone did the best they could, from district supervisors to school administration, to teachers, to students. Students and some teachers did not have their own devices, nor did everyone have reliable internet access. Teachers gave each other advice as to which software worked the best, but it seemed everyone was using a different platform. Students were confused from one class to the next with all the different styles. It was a period of great uncertainty, where from week to week, it was not known if there would be a return to in-person learning. It was decided that it would not be equitable for students to be held accountable for missed assignments in that period because many did not have the necessities for learning in that manner. The result for the 2019-2020 school year was that students effectively missed half of the content for their spring classes. Nevertheless, they went on to the next set of classes the following year. For example, a student in an Algebra 2 class, having learned half of the content, went on to Trig, and so forth. The ripple effect can still be observed.

For the 2020-2021 school year, District X was more prepared. Every student got a device and hotspots were given to anyone without adequate internet access. A single-sign-on system was implemented, meaning students did not have to memorize several different usernames and passwords, and one single platform was purchased for everyone to use. The fall of 2020 was largely remote, but at least there was a plan and students were held accountable. Live class meetings were regularly held, but there was still the challenge of student engagement and monitoring, and suddenly the opportunities and motives to cheat were dramatically increased. Although students were being taught and accountable for submitting work, there is just no way to know how many learned the concepts. Again, the ripple effect can still be observed.

The purpose of this project is to study the effects of this major interruption on student performance in mathematics and to create a predictive model for future student performance if such an event happens again. Supervised and unsupervised machine learning [2] models that can be used to predict a student's performance will be created and compared. The models that will be used include Random Forest Classifier, Decision Tree and Random Forest Regression, and K-Means Clustering, checking for distortion and silhouette coefficients. Using end-of-year testing data for current seniors in District X, including any available test scores in grades 9-12, their corresponding proficiency levels (such as l-Novice, 2-Below Mastery, …, 5-Distinguished), both pre-pandemic and during, current GPA, letter grade in the three core math classes of Algebra 1, Geometry, and Algebra 2, whether they took any of those classes in either spring or fall of 2020, when the majority of the e-learning took place. This study focuses on current seniors. Any identifying information for students, school, and school district will be removed.

For future work, the models can be tested for accuracy on current juniors who will be tested in May 2022, as they have experienced similar virtual learning obstacles as the seniors.

II. Related Works

One of the most significant challenges to e-learning systems is student motivation and engagement, according to Hussain, Zhu, Zhang, and Raza Abidi [1]. They used machine learning techniques to identify low-engagement students in a social science course at the Open University (OU) so they could assess the effect of engagement level on student performance. The input variables were highest education level, final results, score on the assessment, and number of clicks on virtual learning (VLE) activities. The output variable was the level of engagement on the activities. The authors applied several machine learning (ML) algorithms, trained the models, and then compared the accuracy and kappa values of the models. Their results found that J48, decision tree, JRIP, and gradient-boosted classifiers exhibited best in terms of accuracy, kappa value, and recall. They evaluated the techniques using cross-validation. A dashboard was developed for use by the professors at the university to aid in better intervention for students in advance of exams. They also studied the relationship between student engagement and final exam score. This work is limited in its scope to *asynchronous* e-learning where student engagement is harder to measure. In the type of courses studied here, called "massive open online courses [1]," students download lessons/videos and pace themselves. The most difficult challenge of these courses is dropout rate. This study differs from mine in that the dropout rate for high school students in *synchronous* e-learning classes is rare. Reviewing this work is beneficial to my study in that the authors used several types of ML techniques, some of which I will be using and can follow their examples by running their dataset through my algorithms for comparison. The data used in this study are publicly available at https://analyse.kmi.open.ac.uk/open_dataset) for research purposes.

In the Machine Learning in E-Learning thesis [2] written by Samirah Siddiq, the author proposes a model to predict a learner's final grade in a course at earlier stages in the course, which would then aid the instructors in intervening when students that are struggling. In the setup for the study, many types of e-learning and course styles are described. Challenges of e-learning are many. One is the challenge with delivery of the content. In a synchronous method, the teaching and learning is happening at the same time, live and in real time. Some of the problems there are slow internet connections, learning curve with software, and lack of hands-on opportunities. In asynchronous style, students must pace themselves. Keeping them motivated and feeling included is a challenge. Students do not get immediate feedback from a teacher. Another challenge the author describes is "personalization," or trying to tailor the course to students' needs and learning styles. She believes this is done through two steps: Classification and Recommendation. First classify students based on similarities, then recommend the next courses for them to take. She says another part of the recommendation step is to identify students who need help. She goes on to mention 5 categories of machine learning: supervised, unsupervised, semi-supervised, reinforcement, and deep learning. The first two are described in the paper. "Supervised learning involves having a dataset with the correct output that is used to 'train' the system…On the contrary, unsupervised learning process is finding such relations between the features in the dataset without knowing the right results during training." In the latter, an algorithm tries to cluster points based on their statistical properties. Experimental results were given but there were no explicit conclusions or explanations of the results in context.

In Using Data Mining to Predict Secondary School Student Performance [3], Cortez and Silva focused on two questions: Is it possible to predict student performance? And What are the factors that affect student achievement? Modeling student performance can help to better understand and improve it. They mention several other studies that have addressed the issue of student performance, including one that, using the data mining method of association rules, proposed a solution that outperformed the traditional way of assigning students to remedial classes. The aim of this study and others like it is to predict student achievement, if possible, and to identify variables that may affect success or failure, so that some action may be taken to intervene while the student in the school system. To address the prediction of final grades in Math and Portuguese (language) classes, using past grades and demographic data, the authors used the data in 3 different ways: binary (pass/fail), 5-level classification (A-F), and regression (scores 1-20). They then employed 4 different data mining methods: Decision Trees (DT), Random Forests (RF), Neural Networks (NN), and Support Vector Machines (SVM). They concluded that "student achievement is highly affected by previous performances [3]." They found that only a small portion of the many input variables seemed to be relevant, so at the time of the publication indicated that they intend to explore automatic feature selection methods, which they expected to benefit the NN and SVM algorithms "which are more sensitive to irrelevant inputs." This study data is publicly available at https://www.kaggle.com/larsen0966/student-performance-data-set. Since this study is very similar to what I hope to accomplish, I will be able to run their data through my algorithms to compare to their results. This will give more confidence that my methods are sound.

Iatrellis, Savvas, Fitsilis, and Gerogiannis employed a two-phase machine learning approach [4], using both supervised and unsupervised techniques for predicting outcomes for students in higher education. With the K-Means algorithm, their experiments discovered three clusters of students. These clusters were used to train models to address the performance of each cluster individually. They looked at time to degree completion and enrollment in the offered programs. They claimed their developed models can predict with relatively high accuracy. They also describe how the "clustering-aided [4]" approach could be useful for analyzing learning in higher education.

The authors felt that their work had limitations in that the clusters of students were formed with data alone, without perspective of faculty, and that such input may affect the groupings.

In a work presented as a "student marks and prediction system using supervised learning techniques [5]," Yousafzai, Hayat, and Afzal analyzed the quality of education in 7 regions of Pakistan for intermediate and secondary schools. Data with 29 attributes was collected from a federal education board in Islamabad, Pakistan, preprocessed, and used to train both a decision tree classifier and a regression model. They used the classification model to predict the grade and the regression model to predict the marks. They concluded that the decision tree classifier and regression had "impressive results," as their classification accuracy was 96.64%. In the paper, they go into detail about the steps taken in each of decision tree and K-Nearest Neighbor algorithms. This will help in understanding if these are used in my study.

Since data sizes have grown faster than processor speed, the capabilities of machine learning methods is more limited by the computing time rather than the sample size [6]. Bottou says that "unlikely optimization algorithms such as stochastic gradient descent show amazing performance for large-scale problems [6]." The paper explains the terms gradient descent and stochastic gradient descent as well as gives examples of the latter.

Logistic regression is a statistical model that can be used to predict binary classes. A. Majumder, S. Gupta, D. Singh, and S. Majumder [7] proposed a method of identifying whether a patient is at risk for Covid-19 based on many preconditions. In their paper they detailed the steps and much of the code they used to perform logistic regression. Their proposed model is claimed to have 92% accuracy. Although my study is not about Covid 19, I will be able to run the data they used through my logistic regression model to check that the results are comparable to theirs before using it for the student performance data. The source of their dataset is https://www.kaggle.com/tanmoyx/covid19-patient-precondition-dataset?select=covid.csv.

A Support Vector Machine (SVM), based on finding a hyperplane that divides a dataset into two classes, is a machine learning algorithm that can be used for classification and regression purposes. Naicker, Adeliyi, and Wing compared the performance of support vector machines with some classical machine algorithms, concluding that they were far superior in predicting student outcomes [8]. Educational Data Mining (EDM) provides educational institutions with the tools to improve pass rates, curb dropout rates, and increase retention rates [8]. Using a linear support vector machine (LSVM), this paper attempted to answer the question, "What are the strong impacting factors for school-based learners' performance in reading, writing, and mathematics [8]?" The linear SVM classifier was "benchmarked with 10 other algorithms such as coarse decision tree, medium decision tree, fine decision tree, logistic regression, Gaussian Naïve Bayes, Kernel Naïve Bayes, quadratic SVM, cubic SVM, fine Gaussian SVM, and medium Gaussian SVM [8]." The data used is publicly available at https://www.kaggle.com/spscientist/students-performance-in-exams/activity.

Higher education added technology to traditional teaching methods years ago. Machine learning has opened the door to have a chance at predicting student performance. Sekeroglu, Dimililer, and Tuncal considered two datasets "for the prediction and classification of student performance respectively using five machine learning algorithms [9]," to "evaluate the efficiency of raw data without any data selection or preparation algorithm [9]." They performed eighteen experiments and said that "preliminary results suggest that performances of students might be predictable and classification…can be increased by applying pre-processing to the raw data before implementing machine learning algorithms [9]." They went on to discuss relatively recent developments in artificial intelligence (AI) and special education that will "enable collaborative educational environments [9]" and facilitate the academic needs of students with special needs and their families [9]. The authors used three machine learning algorithms to predict student performance: Backpropogation (BP), Support Vector Regression (SVR) and Long-Short Term Memory (LSTM). For the classification experiments, they used BP, SVR ("SVM in classification [9]"), and Gradient Boosting Classifier (GBC). Results are given for the prediction and classification experiments, as well as explanations of each of the methods. This study found significant results and it was concluded that educational data can be predicted by machine learning algorithms but results might be improved by other kinds of data selection and algorithms that were not used within.

"Classification methods are typically strong in modeling interactions [10]." Soufi, Amin, and Awan present the basic classification techniques. Their study is a "comprehensive review of different classification techniques in machine learning [10]." They describe Decision Tree Induction, Bayesian Networks, K-Nearest Neighbors, and Support Vector Machines. The work is useful for both seasoned and newcomers in the field.

"The K-Means algorithm is generally the most known and used clustering method [11]." It is technically a supervised machine learning tool, but Sinaga and Yang set out to construct an *un*supervised k-means model that was free of initializations and parameter selections but could also find an optimal number of clusters. They called it "U-k-means [11]." They analyzed its computational complexity and compared it to existing methods. This is a very interesting idea and useful to study because it takes an existing method and stretches its boundaries. They claim their results demonstrate superiority of the U-k-means algorithm.

III. PROPOSED METHOD AND EVALUATION

Data collected from 146 seniors in a District X high school included their test scores from 2018 8th grade,

fall 2019 10th grade, and spring 2021 11th grade. The tests were in the ranges as described in the tables below, including proficiency levels. Because the tests have different score ranges for the proficiency levels, scores were standardized before use in the model.
For all three tests, the proficiency levels are 1-Novice, 2-Below Mastery, 3-Mastery, 4-Above Mastery, 5-Distinguished. With test scores, proficiency levels were listed, but the breakdown of intervals with levels was not given. There are gaps in the intervals because these are based on the data and the full intervals are not known. For example, a student scoring a 528 might be a proficiency level 1 or 2.

2018 8th grade test

Score Start	Score End	Proficiency Level
380	527	1
529	586	2
587	616	3
618	830	4
?	?	5

2019 PSAT 10th grade

Score Start	Score End	Proficiency Level
280	450	1
460	470	2
480	500	3
510	520	4
530	640	5

2021 SAT 11th grade

Start	End	Proficiency Level
320	410	1
420	510	2
520	580	3
590	740	4
?	?	5

The attributes available in the dataset were ID (student ID number), Gender (M or F), 2018 Test Score (number from 380-630), standardized 2018 Test Score, Prof 2018 (2018 test proficiency level), 2019 Test Score (number from 280-640), starndardized 2019 Test Score, Prof 2019 (2019 test proficiency level, 2021 Test Score (number from 320-740), standardized 2021 Test Score, Prof 2021 (2021 test proficiency), GPA, grades from Alg 1, Geo, and Alg 2, Math class grade average, and Math class grade average (Adjusted for Pandemic), which is explained below.

Spring of 2020 classes at this school were interrupted by an abrupt halt to normal instruction. Students were sent home and teachers scrambled to figure out lesson plans and means of getting work to students. Roughly halfway through the semester, students were allowed to stop working and keep the grades they had, or they could work to improve them. Because of this, many students received grades that did not reflect the amount of learning they received. For students taking a core math class in Spring semester 2020, I deducted 1 point before calculating the average. Grades were recorded as A-4, B-3, C-2, D-1, and F-0. If a student had a core math class in Fall of 2020, where much of it was remote, however they all had new devices and hotspots available for use, they did more to earn their grades, but rampant cheating was suspected. Many students admitted it was easier to cheat. For this I will deduct ½ point because they were held accountable for the entire semester of content, but still had possible grade inflation. This with the total of their math class grade points, will give a math class grade average that is weighted for the challenges of the pandemic. The -1 and -0.5 will reflect loss of instruction and/or possibly inflated grades due to circumstances.

All students in the study took the 3 core math classes before their 2021 SAT test. To find the attributes that were the best in predicting the 2021 SAT proficiency level, a Random Forest Classifier was used first with all the data. Here are the results:

Accuracy: 0.67

	feature	importance
2	2018 Test Score	0.138661
3	standardized 2018 Test Score	0.135584
5	2019 Test Score	0.119198
8	GPA	0.105431
6	standardized 2019 Test Score	0.103127
0	ID	0.099585
13	Math class grade average (Adjusted for Pandemic)	0.081452
7	Prof 2019	0.064092
4	Prof 2018	0.040864
12	Math class grade average	0.039815
11	Alg 2	0.026542
1	Gender	0.019277
10	Geo	0.018391
9	Alg 1	0.007983

From this list of feature importances and some assumptions, I removed features that were either confounding or were thought to logically be useless at predicting academic performance. Removed attributes were:

- gender--while gender could be a factor in performance, it was near the bottom of the list,
- ID—student ID number is arbitrary and certainly would not be a good indicator,
- 2018 and 2019 test scores—although standardized test scores did not necessarily have greater feature importances, it will be better to use the standardized scores since the actual score ranges are so different in the two tests,
- 2021 Test Score and standardized 2021 Test Score had to be removed because they shouldn't be used to predict their own performance,

- Alg 1, Geo, and Alg 2 grades were removed because the class grade average that was adjusted for the pandemic had a greater feature importance and measures the same thing, and
- Math class grade average was removed because it is not used.

A second Random Forest Classifier was run with these results:
Accuracy: 0.70

	feature	importance
2	standardized 2019 Test Score	0.239000
0	standardized 2018 Test Score	0.228710
4	GPA	0.217022
5	Math class grade average (Adjusted for Pandemic)	0.141841
3	Prof 2019	0.090399
1	Prof 2018	0.083028

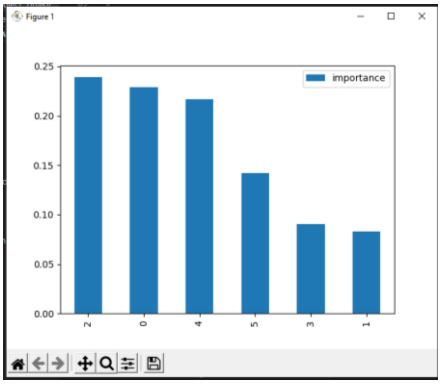

Fig. 1: Feature importances generated by the Random Forest Classifier using the 6 attributes

Decision Tree Regressors for the top two feature importances are shown in the figures below. In Fig. 2, r^2 = 0.5466 and in Fig. 3, r^2 = 0.5247.

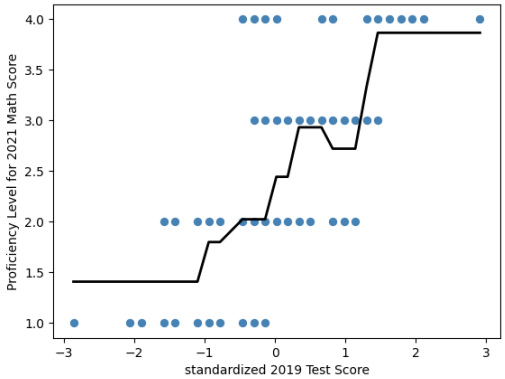

Fig. 2: 2021 Proficiency vs. standardized 2019 Test

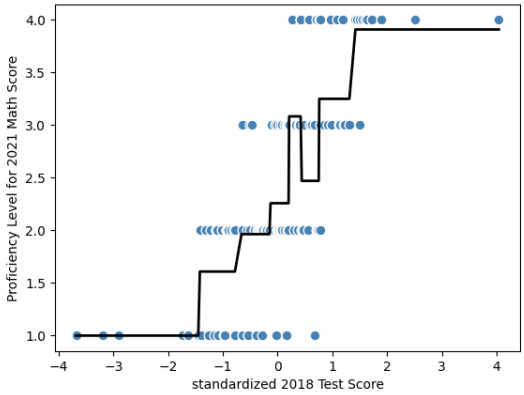

Fig. 3: 2021 Proficiency vs. standardized 2018 Test

Random Forest Regression with 2021 Proficiency vs. standardized 2019 Test Score had a mean squared error of 0.064 for the training set, 0.375 for the test set, and r-squared of 0.934 for the training set, and 0.440 for the test set. The residual plot is shown in Fig. 4.

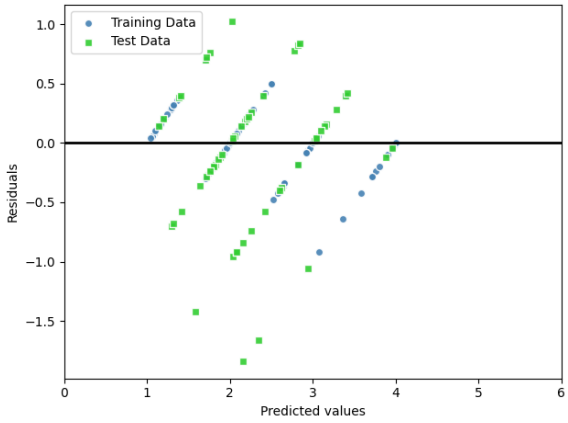

Fig. 4: Residual plot of predicted 2021 proficiency with standardized 2019 Test Score as explanatory variable

The k-means clustering algorithm with 150 samples shows 4 distinct clusters. Fig. 5 shows the clusters and Fig. 6 shows the elbow at 4, meaning 4 clusters is where the distortion starts to increase the most rapidly [12] and is the optimal number of clusters for this data. The distortion at the elbow is 451.62, which is a stark contrast to over 10,000 at just 3 clusters. Fig. 7 shows the silhouette coefficient of each cluster.

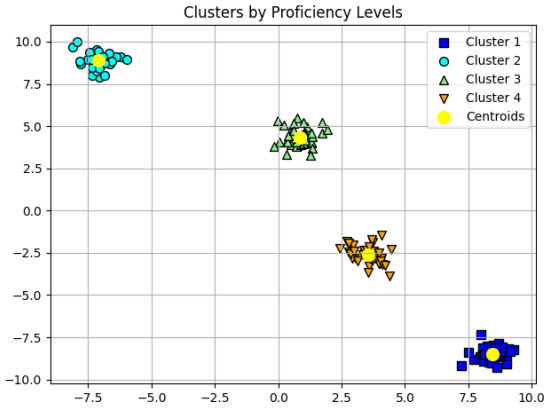

Fig. 5: K-Means algorithm generated 4 clusters with cetnroids shown.

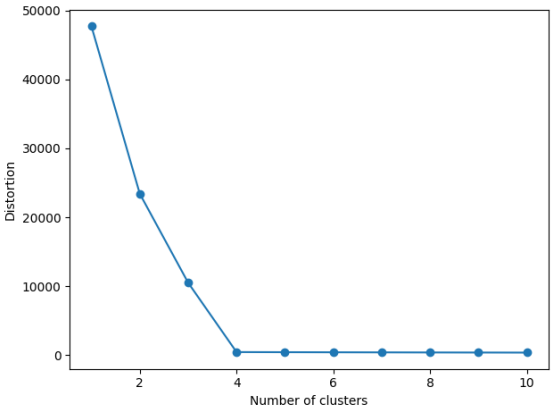

Fig. 6: Optimal number of clusters is 4, at the elbow

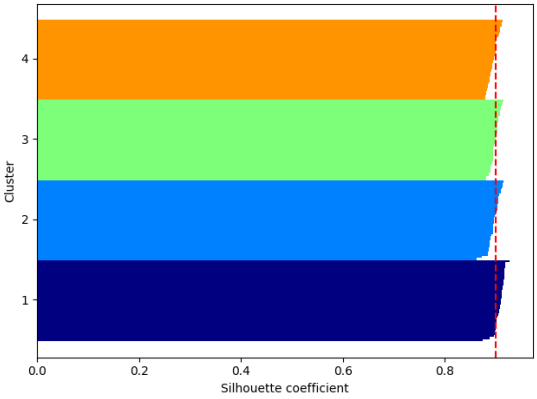

Fig. 7: This model shows an example of good clustering since the silhouette coefficients are not close to zero [12].

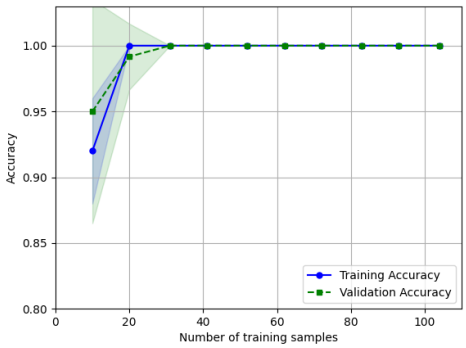

Fig. 8: Learning curve comparing training and validation accuracies.

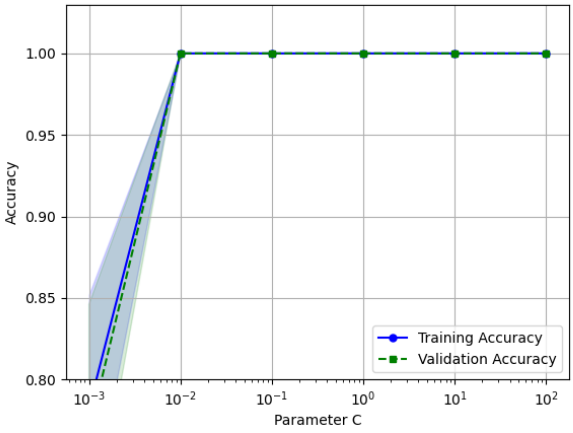

Fig. 9: Validation curve

In Fig. 8 above, we see a learning curve that shows it takes about 30 training samples for our model to perform quite well. Fig. 9 shows the use of the inverse regularization parameter C of the Logistic Regression Classifier. For C values of less than 0.01, the model has a greater risk of underfitting. The results of stratified k-fold cross validation with 10 folds is below.

Test Accuracy 100%
Fold: 1, Class dist.: [26 26 26 26], Acc: 1.000
Fold: 2, Class dist.: [26 26 26 26], Acc: 1.000
Fold: 3, Class dist.: [26 26 26 26], Acc: 1.000
Fold: 4, Class dist.: [26 26 26 26], Acc: 1.000
Fold: 5, Class dist.: [26 26 26 26], Acc: 1.000
Fold: 6, Class dist.: [26 26 26 26], Acc: 1.000
Fold: 7, Class dist.: [26 26 26 27], Acc: 1.000
Fold: 8, Class dist.: [26 26 27 26], Acc: 1.000
Fold: 9, Class dist.: [26 27 26 26], Acc: 1.000
Fold: 10, Class dist.: [27 26 26 26], Acc: 1.000

CV accuracy: 1.000 +/- 0.000

IV. CONCLUSIONS

The goal of this research was to use pre-pandemic standardized test scores and proficiency levels, math grades, and grade point average of high school seniors to predict post-pandemic standardized test proficiency levels and to determine if the pandemic had an effect on learning outcomes. The correlation between proficiency levels from 2018 to 2019 was about 0.620 and between 2019 and 2021 was higher, at 0.688. Looking at the correlation of the best predictor of target proficiency, the 2019 standardized test score, and the target variable, the correlation was even higher at 0.739. A 95% confidence interval from a hypothesis test for slope showed that we can be 95% confident that the true slope of the regression line, modeling target proficiency predicted by 2019 standardized test score, is between 0.58 and 0.78. The p-value was extremely low so we can conclude that there is in fact a nonzero correlation between the two variables. A paired t-test for means of the 2019 and 2021 standardized test scores showed that there was no significant evidence of a difference between the scores. All of this demonstrates that the pandemic seemed to have little effect on the math proficiency of this group of high school seniors.

We can, however, see from figures 1-7 that there is great predictive power of past performance for future performance. Fig. 1 shows that the two prior tests and GPA were the greatest predictors for the 2021 test proficiency level. Those variables along with math class grades and prior test proficiency levels helped the random forest classifier to have its greatest accuracy of 70%. Figures 2-4 show a positive correlation between each of 2018 and 2019 vs. the 2021 proficiency levels. The residual plot is a bit weird to look at because of the discrete target variables, but nevertheless shows no distinct pattern, showing the linear model is a good fit. Figures 5-7 show 4 distinct clusters of data, that 4 clusters is the optimal number, and that the clustering was good because the shapes in the graph of the silhouette coefficients are of similar length and width, and are not close to zero. Figures 8 and 9 show that 30 samples or greater is optimum, and that if a C parameter is used, it is best to use values greater than 0.01.

V. FUTURE WORKS

A new crop of juniors just took their end-of-year exams in District X. Scores will be available in the near future and it will be very interesting to study relationships in their data as well. As newer students move up, as all of their learning will be post-pandemic, non-virtual, it will then be interesting to compare with this year's seniors and juniors to see if the pandemic may have actually had an effect that was not discovered.

VI. CODE EXAMPLES

The following code was adapted from the textbook, Python Machine Learning-Third Edition [12]. Many imports were needed but not shown here. Comments separate the sections to show examples of each algorithm and evaluation. Some code is not shown.

```
# Split the data into training and testing sets
XTrain, XTest, yTrain, yTest =
train_test_split(X,y,stratify=y, test_size=.2,
random_state=1)
print(XTrain.shape, XTest.shape)
# Fit the model to random forest classifier
# ****************************************
rfc.fit(XTrain, yTrain)
yPred = rfc.predict(XTest)
# Calculate the model accuracy
print('Accuracy:
%.2f%%'%(100*accuracy_score(yTest,yPred)))
# Visualize important features
feature_importances_df =
pd.DataFrame({'feature':list(X.columns),'importan
ce':rfc.feature_importances_}).sort_values('impor
tance',ascending=False)
print(feature_importances_df)
# Bar chart showing feature importances
df = pd.DataFrame(feature_importances_df)
df.plot(kind='bar')
plt.show()
# end random forest classifer
*******************************

# Decision Tree Regressor for standardized 2019
Test Score to predict Prof 2021
df = pd.DataFrame(X)
X1 = df[['standardized 2019 Test Score']].values
tree.fit(X1,y)
sort_idx = X1.flatten().argsort()
lin_regplot(X1[sort_idx],y[sort_idx],tree)
regr = LinearRegression()
reg = regr.fit(X1,y)
y_lin_fit = regr.predict(X1)
linear_r2 = r2_score(y,y_lin_fit)
print('r-squared standardized 2019 Test Score vs.
Prof 2021: %.3f' %linear_r2)
plt.xlabel('standardized 2019 Test Score')
plt.ylabel('Proficiency Level for 2021 Math
Score')
plt.show()
```

```
# Random Forest Regression
XTrain, XTest, yTrain, yTest =
train_test_split(X,y,test_size=0.4,random_state=1
)
forest =
RandomForestRegressor(n_estimators=50,criterion='
squared_error',random_state=1,n_jobs=-1)
forest.fit(XTrain,yTrain)
y_train_pred = forest.predict(XTrain)
y_test_pred = forest.predict(XTest)
print('Mean Squared Error training: %.3f, test:
%.3f' %(mean_squared_error(yTrain,y_train_pred),
mean_squared_error(yTest, y_test_pred)))
print('R-squared training: %.3f, test: %.3f
'%(r2_score(yTrain,y_train_pred),
r2_score(yTest,y_test_pred)))
# predictions will be continuous numbers from 1
to 5, but I want to convert that back to a whole
number since
#   all the known final test proficiency levels
were kept as whole numbers 1-5.
plt.scatter(y_train_pred,y_train_pred-
yTrain,c='steelblue',edgecolor='white',marker='o'
,s=35,alpha=0.9,label='Training Data')
plt.scatter(y_test_pred,y_test_pred-
yTest,c='limegreen',edgecolor='white',marker='s',
s=35,alpha=0.9,label='Test Data')
plt.xlabel('Predicted values')
plt.ylabel('Residuals')
plt.legend(loc='upper left')
plt.hlines(y=0,xmin=0,xmax=6,lw=2,color='black')
plt.xlim([0,6])
plt.tight_layout()
plt.show()

# Unsupervised K-Means Clustering (start over
with raw data) and
#   drop any evidence of target variable, like
any 2021 data
X = studentDataset.drop('Prof 2021', axis=1)
X = X.drop('2021 Test Score',axis=1)
X = X.drop('standardized 2021 Test Score',axis=1)
X = X.drop('ID',axis=1)
X = X.drop('Gender',axis=1)
X, y =
make_blobs(n_samples=145,n_features=13,centers=4,
cluster_std=0.5,shuffle=True,random_state=0)

# KMeans clustering Results
km =
KMeans(n_clusters=4,init='random',n_init=10,max_i
ter=300,tol=1e-4,random_state=0)
y_km = km.fit_predict(X)
plt.scatter(X[y_km == 0,0],X[y_km ==
0,1],s=50,c='blue',marker='s',edgecolor='black',l
abel='Cluster 1')
plt.scatter(X[y_km == 1,0],X[y_km ==
1,1],s=50,c='cyan',marker='o',edgecolor='black',l
abel='Cluster 2')
plt.scatter(X[y_km == 2,0],X[y_km ==
2,1],s=50,c='lightgreen',marker='^',edgecolor='bl
ack',label='Cluster 3')
plt.scatter(X[y_km == 3,0],X[y_km ==
3,1],s=50,c='orange',marker='v',edgecolor='black'
,label='Cluster 4')
plt.scatter(km.cluster_centers_[:,0],km.cluster_c
enters_[:,1],s=100, c='yellow',label='Centroids')
plt.title('Clusters by Proficiency Levels')
plt.legend(scatterpoints=1)
plt.grid()
plt.tight_layout()
plt.show()

print('Distortion: %.2f'%km.inertia_)
# identify the number of clusters k where the
distortion begins to increase more rapidly,
#   producing an elbow chart
distortions=[]
for i in range(1,11):
    km = KMeans(n_clusters=i,init='k-
means++',n_init=10,max_iter=300,random_state=0)
    km.fit(X)
    distortions.append(km.inertia_)
plt.plot(range(1,11),distortions,marker='o')
plt.xlabel('Number of clusters')
plt.ylabel('Distortion')
plt.tight_layout()
plt.show()

# Create a plot of the silhouette coefficients
for a k-means clustering with k=4.
```

```
km = KMeans(n_clusters=4,init='k-
means++',n_init=10,max_iter=10,tol=1e-
04,random_state=0)
y_km = km.fit_predict(X)
cluster_labels = np.unique(y_km)
n_clusters = cluster_labels.shape[0]
silhouette_vals =
silhouette_samples(X,y_km,metric='euclidean')
y_ax_lower, y_ax_upper = 0,0
yticks = []
for i, c in enumerate(cluster_labels):
    c_silhouette_vals = silhouette_vals[y_km ==
c]
    c_silhouette_vals.sort()
    y_ax_upper += len(c_silhouette_vals)
    color = cm.jet(float(i)/n_clusters)
    plt.barh(range(y_ax_lower,y_ax_upper),c_silho
uette_vals,height=1.0,edgecolor='none',color=colo
r)
    yticks.append((y_ax_lower+y_ax_upper)/2.)
    y_ax_lower += len(c_silhouette_vals)
silhouette_avg = np.mean(silhouette_vals)
plt.axvline(silhouette_avg,color='red',linestyle=
'--')
plt.yticks(yticks,cluster_labels + 1)
plt.ylabel('Cluster')
plt.xlabel('Silhouette coefficient')
plt.tight_layout()
plt.show()

# Stratified KFold Cross Validation with 10 folds
from sklearn.model_selection import
StratifiedKFold
from sklearn.pipeline import make_pipeline
from sklearn.decomposition import PCA
XTrain,XTest,yTrain,yTest=train_test_split(X,y,te
st_size=0.2,stratify=y,random_state=1)
print('Data Shape: XTrain, XTest, yTrain, yTest:
',XTrain.shape,XTest.shape,yTrain.shape,yTest.sha
pe)
pipe_lr =
make_pipeline(StandardScaler(),PCA(n_components=2
),LogisticRegression(random_state=1,solver='lbfgs
'))
pipe_lr.fit(XTrain,yTrain)
yPred = pipe_lr.predict(XTest)
print('Test Accuracy %.3f'
%pipe_lr.score(XTest,yTest))
kfold =
StratifiedKFold(n_splits=10).split(XTrain,yTrain)
scores = []
for k, (train,test) in enumerate(kfold):
    pipe_lr.fit(XTrain[train],yTrain[train])
    score =
pipe_lr.score(XTrain[test],yTrain[test])
    scores.append(score)
    print('Fold: %2d, Class dist.: %s, Acc: %.3f'
%(k+1,np.bincount(yTrain[train]),score))
print('\nCV accuracy: %.3f +/- %.3f'
%(np.mean(scores),np.std(scores)))

# Cross-validation accuracy
from sklearn.model_selection import
cross_val_score
scores = cross_val_score(estimator=pipe_lr,
X=XTrain, y=yTrain, cv=10, n_jobs=1)
print('CV accuracy scores: %s' %scores)
print('CV accuracy: %.3f +/- %.3f'
%(np.mean(scores),np.std(scores)))
# Learning Curve
from sklearn.model_selection import
learning_curve, validation_curve
train_sizes, train_scores, test_scores =
learning_curve(estimator=pipe_lr,X=XTrain,y=yTrai
n,train_sizes=np.linspace(0.1,1.0,10),cv=10,n_job
s=1)
train_mean = np.mean(train_scores, axis=1)
train_std = np.std(train_scores, axis=1)
test_mean = np.mean(test_scores, axis=1)
test_std = np.std(test_scores, axis=1)
plt.plot(train_sizes,train_mean,color='blue',mark
er='o',markersize=5,label='Training Accuracy')
plt.fill_between(train_sizes,train_mean+train_std
,train_mean-train_std,alpha=0.15,color='blue')
plt.plot(train_sizes,test_mean,color='green',line
style='--
',marker='s',markersize=5,label='Validation
Accuracy')
plt.fill_between(train_sizes,test_mean+test_std,t
est_mean-test_std,alpha=0.15,color='green')
plt.grid()
plt.xlabel('Number of training samples')
```

```
plt.ylabel('Accuracy')
plt.legend(loc='lower right')
plt.xlim([0,110])
plt.ylim([0.8,1.03])
plt.show()
# Validation Curve
param_range = [0.001, 0.01, 0.1, 1.0, 10.0,
100.0]
train_scores, test_scores =
validation_curve(estimator=pipe_lr,X=XTrain,y=yTr
ain,param_name='logisticregression__C',param_rang
e=param_range,cv=10)
train_mean=np.mean(train_scores, axis=1)
train_std = np.std(train_scores, axis=1)
test_mean = np.mean(test_scores, axis=1)
test_std = np.std(test_scores, axis=1)
plt.plot(param_range, train_mean, color='blue',
marker='o', markersize=5, label='Training
Accuracy')
plt.fill_between(param_range,
train_mean+train_std,train_mean-
train_std,alpha=0.15,color='blue')
plt.plot(param_range, test_mean, color='green',
linestyle='--', marker='s', markersize=5,
label='Validation Accuracy')
plt.fill_between(param_range, test_mean+test_std,
test_mean-test_std, alpha=0.15, color='green')
plt.grid()
plt.xscale('log')
plt.legend(loc='lower right')
plt.xlabel('Parameter C')
plt.ylabel('Accuracy')
plt.ylim(0.8,1.03)
plt.show()
```

REFERENCES

[1] M. Hussain, W. Zhu, W. Zang, and S.M.R. Abidi, "Student Engagment Predictions in an e-Learning System and Their Impact on Student Course Assessment Scores," Hindawi Computational Intelligence and Neuroscience, October 2, 2018.

[2] S. Siddiq. (2017) Machine Learning in E-Learning. DOI 10.13140/RG.2.2.36213.37605.

[3] P. Cortez, A. Silva, "Using Data Mining to Predict Secondary School Student Performance," (2008).

[4] O. Iatrellis, I. K. Savvas, P. Fitsilis, and V. C. Gerogiannis, "A two-phase machine learning approach for predicting student outcomes," Education and Information Technologies, June 25, 2020.

[5] B. Yousafzai, M. Hayat, S. Afzal, "Application of machine learning and data mining in predicting the performance of intermediate and secondary education level student," Education and Information Technologies, April 29, 2020.

[6] L. Bottou, "Large-Scale Machine Learning with Stochastic Gradient Descent," (2010)

[7] A. Majumder, S. Gupta, D. Singh, S. Majumder, "An Intelligent System for Prediction of Covid-19 Case Using Machine Learning Framework-Logistic Regression," *J. Phys.: Conf. Ser*. 1797, October, 2020.

[8] N. Naicker, T. Adeliyi, J. Wing, "Linear Support Vector Machines for Prediction of Student Performance in School-Based Education", *Mathematical Problems in Engineering*, vol. 2020, Article ID 4761468, 7 pages, 2020. https://doi.org/10.1155/2020/4761468

[9] B. Sekeroglu, K. Dimililer, and K. Tuncal. 2019. Student Performance Prediction and Classification Using Machine Learning Algorithms. In Proceedings of the 2019 8th International Conference on Educational and Information Technology (ICEIT 2019). Association for Computing Machinery, New York, NY, USA, 7–11. DOI:https://doi.org/10.1145/3318396.3318419

[10] A. Soofi, A. Amin, A. Awan. "Classification techniques in machine learning: applications and issues." *Journal of Basic and Applied Sciences* 13 (2017): 459-465.

[11] K. P. Sinaga, M. Yang, "Unsupervised K-Means Clustering Algorithm," in IEEE Access, vol. 8, pp. 80716-80727, 2020, doi: 10.1109/ACCESS.2020.2988796.

[12] S. Raschka, V. Mirjalili. *Python Machine Learning-Third Edition: Machine Learning and Deep Learning with Python, scikit-learn, and TensorFlow2.* PACKT Publishing Limited, 2019.