CTA Impact Estimator from Auxiliary Observational Data

Jiaying Wang

April, 2023

1 Introduction

The study conducted by Pane et al. (2014) investigated the effectiveness of the ”Cognitive Tutor Algebra” (CTA) curriculum as an alternative teaching approach in 12 high schools and 32 middle schools in Texas. The study utilized a randomized controlled trials (RCT) to evaluate the effectiveness of the intervention. While RCTs are a reliable method for assessing the impact of interventions, it is also important to consider auxiliary observational data in order to supplement the analysis of the RCT results.

This report focuses on the auxiliary observational data, specifically on the schools that were not included in the RCT study and were randomized in Texas. These schools were not randomized into a treatment or control group, and did not receive the CTA curriculum treatment. The variable of interest for prediction is the percentage of students who passed the TAKS (Texas Assessment of Knowledge and Skills) Math exam for the 2007-2008 academic year.

The report will employ machine learning algorithms on the auxiliary schools' data to produce predictions that can be used as a covariate in order to improve the precision of outcomes in randomized controlled data. This report also explores various pre-processing techniques for the data, the selection of machine learning algorithms, and the impact of the test scores and pre-scores on the prediction.

Ultimately, the objective of this research is to determine whether the CTA curriculum has a positive effect on the mathematical performance of students.

2 Data Preparation

The entirety of the data was acquired through downloading from the website `https://rptsvr1.tea.texas.gov/perfreport/aeis/2008/DownloadData.html`

A first level of data processing was completed by Adam Sales and Charlotte Mann, which produced .Rdata files from the raw TEA data. Within the taksRem data provided, there are specific columns that follow a distinct naming pattern, such as “outmA08”. In this pattern, “out” signifies the percentage of students who passed the TAKS math exam, “m” designates the type of school data (with “m” representing middle school and “h” representing high school), “A” represents the subgroups (including All (“A”), White (“W”), Black (“B”), Hispanic (“H”), Male (“M”), Female (“F”), Economically Disadvantaged (“E”)), and “08” indicates the year in question.

The columns with the aforementioned pattern contains 7 subgroups and data for 2 years (2008 and 2009) in total, but we are solely interested in the ”A08” variables. We aim to analyze the data for middle and high schools separately. However, when we merged the taksRem dataset with the grdXwalk dataset, we noticed that certain schools include both elementary and secondary education (as indicated by the GRDTYPE variable, with ”S” representing Secondary, ”M” representing Middle, ”E” representing Elementary, and ”B” representing both Elementary and Secondary). To address this issue, we established the following classification rules: we treated ”M” and ”E” as middle schools, and ”S” and ”B” as high schools, unless their ”outh” variable was null. We also removed schools that did not fit into either the middle or high school category, resulting in the exclusion of 117 schools out of the initial 2985 schools.

Upon categorizing all schools into either middle or high school data, we proceeded to create dummy variables for the categorical variables. Additionally, we generated columns with a naming convention of ”exist34/45/...” to indicate whether each school existed between the two years under consideration. Specifically, a value of 1 in a given row of the ”exist34” column signified that the corresponding school did not exist in the data during the years 2003 and 2004, while a value of 0 would indicate that the school did exist during that time period. We end up with a middle school dataset comprising of 1394 schools and 5178 covariates, as well as a high school dataset with 1474 schools and 5179 covariates.

3 Missing Value Treatment

The covsRem_noscale dataset contains missing values, which we addressed using two methods: replacing the NAs with 0s or imputing using the missForest algorithm.

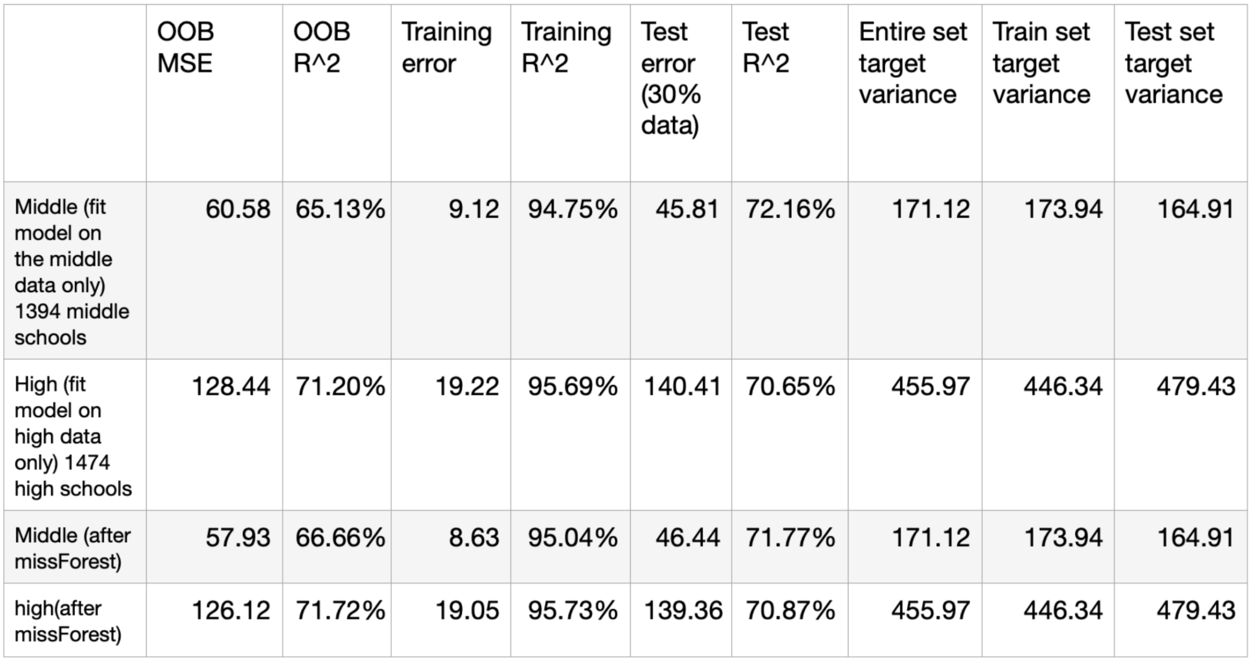
3.1 Replace with 0

To address missing values in each dataset, we replaced all missing values with 0 (since the covariates were already scaled, this is equivalent to mean imputation) and generated missing value indicator covariates for each column. Specifically, for every covariate in the dataset, we created a corresponding missing value indicator column with the naming pattern “mis” + “its original covariate name”. If a row’s value for a covariate was missing, its corresponding missing indicator column was assigned a value of 1 (indicating a missing value), while a value of 0 was assigned if the value was non-missing.

3.2 MissForest Algorithm

For each data set, we replaced every NA using the missForest algorithm in R so that we have a data set without any missing value. Next, we applied scaling to all variables except for the output variable we are interested in predicting. This step helps to ensure that all variables are on the same scale and have equal weight in the analysis. However, it is important to note that scaling may result in zero variance columns, which introduces NA values back to the scaled data set. We addressed this issue by replacing these NA values with 0 as the final step of the data preprocessing.

Then we trained a random forest model using these pre-treated data, and the following Figure 1 summarizes the model's performance.



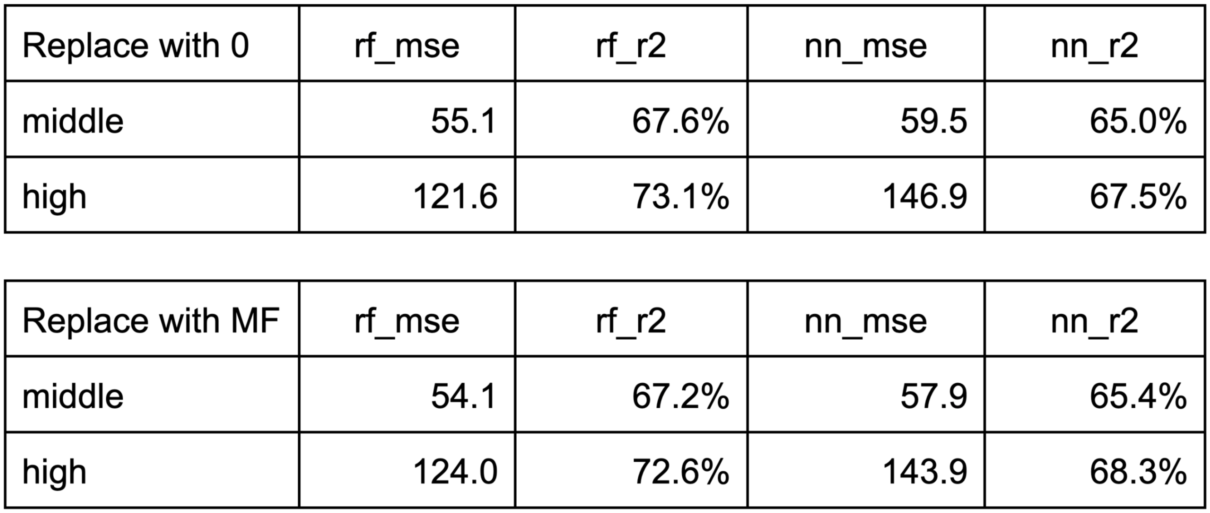
	OOB MSE	OOB R^2	Training error	Training R^2	Test error (30% data)	Test R^2	Entire set target variance	Train set target variance	Test set target variance
Middle (fit model on the middle data only) 1394 middle schools	60.58	65.13%	9.12	94.75%	45.81	72.16%	171.12	173.94	164.91
High (fit model on high data only) 1474 high schools	128.44	71.20%	19.22	95.69%	140.41	70.65%	455.97	446.34	479.43
Middle (after missForest)	57.93	66.66%	8.63	95.04%	46.44	71.77%	171.12	173.94	164.91
high(after missForest)	126.12	71.72%	19.05	95.73%	139.36	70.87%	455.97	446.34	479.43

Figure 1

The result suggests that replacing NA with 0 yields comparable results to imputing with missForest algorithm.

As missForest is a random forest imputation algorithm, using a random forest as the prediction model may have influenced the results. To test this, we introduce the neural network algorithm.

To improve the accuracy of our results, we introduced 10-fold cross-validation to our testing process. In Figure 2 below, ”rf” represents the random forest model, where all covariates are used as predictors. ”Nn” represents the neural network model, where only the top 20 most important covariates, ranked by the random forest algorithm, are used as predictors. For detailed configuration of the neural network, see Appendix 1.



Replace with 0	rf_mse	rf_r2	nn_mse	nn_r2
middle	55.1	67.6%	59.5	65.0%
high	121.6	73.1%	146.9	67.5%

Replace with MF	rf_mse	rf_r2	nn_mse	nn_r2
middle	54.1	67.2%	57.9	65.4%
high	124.0	72.6%	143.9	68.3%

Figure 2

Replacing missing values with missForest (MF) results in slightly better performance in terms of MSE for both random forest and neural network models, except for high school in the random forest model. When R-squared values are taken into account, the results are mixed. Comparing middle rf_mse in two tables, even though MF generates a lower MSE, it also generates a lower R^2.

Overall, the difference in performance between the two methods is relatively small, with both methods achieving similar results. It seems that replacing missing values with 0 is a reasonable approach that can achieve good performance for both random forest and neural network models, especially for datasets with a large number of missing values since MF imputation is computationally expensive. Therefore, we decided to focus on using random forest for our further analysis and stopped using missForest.

4 Covariate Selection & Neural Network Tuning

We explored four methods for selecting covariates in our prediction model. Additionally, we attempted to fine-tune the neural network to determine if different configurations could potentially outperform the random forest algorithm.

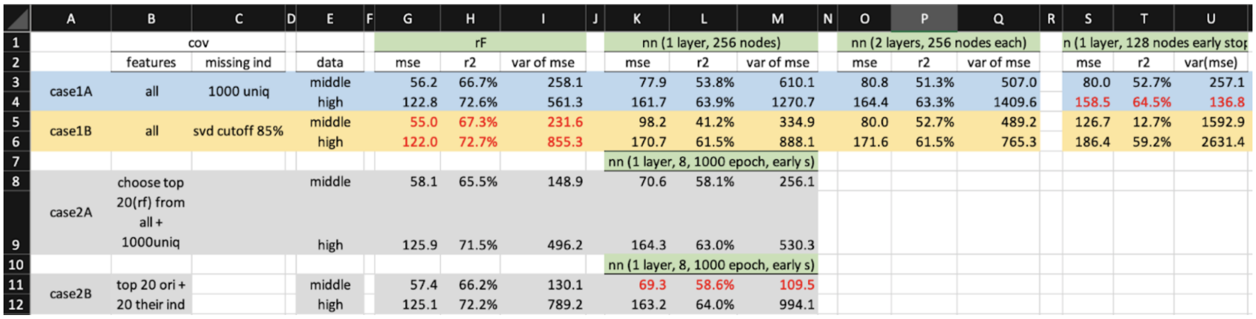
Case 1A: we created missing indicator columns for each original variable, but we only selected the non-repetitive ones as many missing indicator columns were identical. The selected missing indicator columns are represented by the "1000 unique missing ind" in the graph below. Therefore, this model includes all the original variables in the dataset along with these selected 1000 unique missing indicator columns.

Case 1B: we utilized Singular Value Decomposition (SVD) to decrease the number of missing indicator columns. The cutoff point was set to 85%, which means that we chose the variables that cumulatively account for 85% of the variance. This approach helped us to reduce the number of columns in the dataset, while still retaining a significant amount of information.

Case2A: we first ran a Random Forest on the original covariates and obtained a feature importance table. From this table, we selected the top 20 important variables. These 20 variables were then combined with the 1000 unique missing indicator variables created in Case 1A, and fed into the random forest or neural network model for prediction.

Case 2B: we first run Random Forest on the dataset that includes all 5000 original variables and 1000 unique missing indicator variables (as in Case 1A). Then, we obtain a table that ranks the feature importances of all 6000 variables. From this table, we select the top 20 covariates from the original dataset (i.e. excluding the missing indicator variables) and their corresponding 20 missing indicator variables. In total, we select 40 variables to be fed into the random forest/neural network model.

During our attempts to fine-tune the neural network model, we found that random forest consistently outperformed the neural network models after experimenting with various configurations such as adjusting the number of layers and nodes and introducing early stopping. As a result, we decided to use random forest as our main modeling approach moving forward (Figure 3). For further information, please refer to the excel document "Covariate Selection & Neural Network Tuning_table.xlsx" on Github. Line 4 and Line 11 are highlighted in red to indicate that these neural network tuning attempts produced the best result for high school and middle school, respectively, compared to all other neural network attempts.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1		cov					rF				nn (1 layer, 256 nodes)				nn (2 layers, 256 nodes each)				n (1 layer, 128 nodes early stop		
2		features	missing ind		data		mse	r2	var of mse		mse	r2	var of mse		mse	r2	var of mse		mse	r2	var(mse)
3	case1A	all	1000 uniq		middle		56.2	66.7%	258.1		77.9	53.8%	610.1		80.8	51.3%	507.0		80.0	52.7%	257.1
4					high		122.8	72.6%	561.3		161.7	63.9%	1270.7		164.4	63.3%	1409.6		158.5	64.5%	136.8
5	case1B	all	svd cutoff 85%		middle		55.0	67.3%	231.6		98.2	41.2%	334.9		80.0	52.7%	489.2		126.7	12.7%	1592.9
6					high		122.0	72.7%	855.3		170.7	61.5%	888.1		171.6	61.5%	765.3		186.4	59.2%	2631.4
7											nn (1 layer, 8, 1000 epoch, early s)										
8	case2A	choose top 20(rf) from all + 1000uniq			middle		58.1	65.5%	148.9		70.6	58.1%	256.1								
9					high		125.9	71.5%	496.2		164.3	63.0%	530.3								
10											nn (1 layer, 8, 1000 epoch, early s)										
11	case2B	top 20 ori + 20 their ind			middle		57.4	66.2%	130.1		69.3	58.6%	109.5								
12					high		125.1	72.2%	789.2		163.2	64.0%	994.1								

Figure 3: Covariate Selection & Neural Network Tuning Table.xlsx

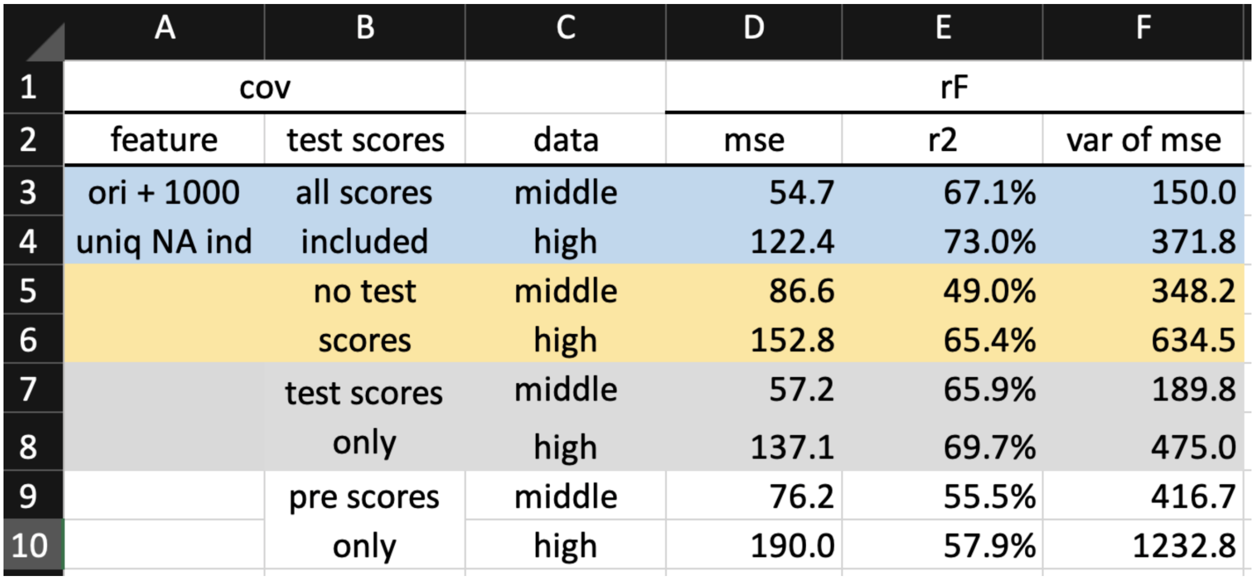
Based on Figure 3 above, the random forest model from case 1B achieved the best performance (in red), while the model from case 2B performed almost as well as the model from case 1A. This suggests that using singular vector decomposition to reduce the number of missing indicator columns can slightly improve the model's performance.

Moreover, we observed that selecting only the top 20 most important variables from the original dataset (case 2B) can still yield a relatively accurate prediction. Notably, most of these top 20 variables are test scores from previous years (see Appendix for more details). Therefore, the finding suggests that a relatively simple model can achieve a good prediction performance for this data set.

5 Variables Impact Exploration

We conducted a further analysis on the variables used in case1A, which included all the original covariates along with 1000 unique missing indicator covariates, resulting in a total of 6000 covariates. We were particularly interested in the impact of test scores from previous years, which were included in the original covariates, on our prediction.

We made three modifications to the 6000 covariates in the dataset (Figure 4: Case1A&Random Forest Tuning result.xlsx). In the yellow chunk, we excluded all test scores from previous years (e.g., science/reading/Spanish scores, etc) as well as any corresponding missing indicator columns. In the gray chunk, we included all test scores and their indicator columns but excluded all other factors such as Campus Financial Information, Demographics, etc. In the white chunk, the pre-scores refer to the TAKS math score in 2007. There are ten pre-scores in total in the original covariates, five "prem" (m refers to middle school) and five "preh" (h refers to high school). For the middle school dataset, we selected the "prem"s with their corresponding five missing indicators, and the same goes for the high school dataset.

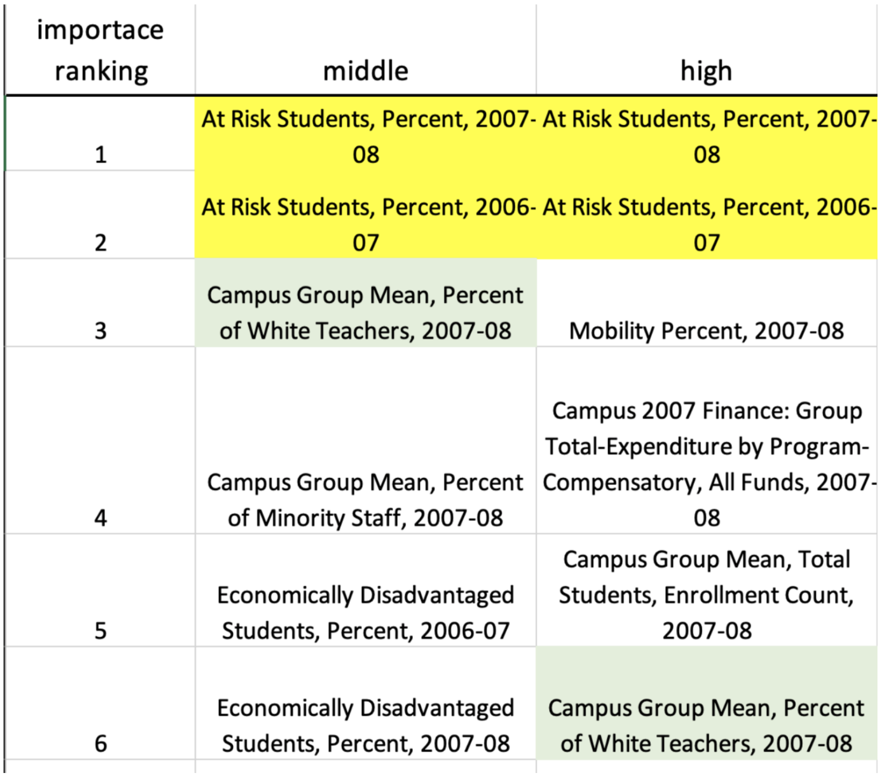


	A	B	C	D	E	F
1	cov			rF		
2	feature	test scores	data	mse	r2	var of mse
3	ori + 1000	all scores	middle	54.7	67.1%	150.0
4	uniq NA ind	included	high	122.4	73.0%	371.8
5		no test	middle	86.6	49.0%	348.2
6		scores	high	152.8	65.4%	634.5
7		test scores	middle	57.2	65.9%	189.8
8		only	high	137.1	69.7%	475.0
9		pre scores	middle	76.2	55.5%	416.7
10		only	high	190.0	57.9%	1232.8

Figure 4: Case1A & Random Forest Tuning Result.xlsx

It is observed that the test scores alone can provide a relatively good prediction for the prediction performance.

We conducted further analysis to determine which variables, aside from test scores, contribute the most to the prediction (Figure 5). We ranked the important variables in the yellow chunk, excluding all test scores from previous years and their corresponding missing indicator columns. Please refer to the excel document "Case1A_importance.xlsx" on Github for more details. Variables with the same color represent the identical variable in both the middle and high school datasets.



importace ranking	middle	high
1	At Risk Students, Percent, 2007-08	At Risk Students, Percent, 2007-08
2	At Risk Students, Percent, 2006-07	At Risk Students, Percent, 2006-07
3	Campus Group Mean, Percent of White Teachers, 2007-08	Mobility Percent, 2007-08
4	Campus Group Mean, Percent of Minority Staff, 2007-08	Campus 2007 Finance: Group Total-Expenditure by Program-Compensatory, All Funds, 2007-08
5	Economically Disadvantaged Students, Percent, 2006-07	Campus Group Mean, Total Students, Enrollment Count, 2007-08
6	Economically Disadvantaged Students, Percent, 2007-08	Campus Group Mean, Percent of White Teachers, 2007-08

Figure 5: Case1A_importance.xlsx

6 Random Forest Tuning

We next focused on tuning the hyperparameters of the random forest model in case1A. The hyperparameters we considered were ntree, mtry, nodesize, and maxnode.

ntree - the number of trees to grow. The larger the tree, the more computationally expensive it is to build models.

mtry - how many variables we should select at a node split. The default value is p/3 for regression and sqrt(p) for classification (p is the number of covariates). We should always try to avoid using smaller values of mtry to avoid overfitting.

nodesize - It refers to how many observations we want in the terminal nodes. This parameter is directly related to tree depth. The higher the number, the lower the tree depth. With lower tree depth, the tree might fail to recognize useful signals from the data.

maxnode - Maximum number of terminal nodes trees in the forest can have. If not given, trees are grown to the maximum possible (subject to limits by nodesize). If set larger than maximum possible, a warning is issued. Setting a larger value for maxnodes can lead to more complex trees that are able to capture intricate patterns in the data. However, it may also increase the risk of overfitting.

We tuned each hyperparameter individually by holding the values of the remaining hyperparameters fixed. To locate the optimal hyperparameters, we utilized 10-fold cross-validation and calculated the average mean squared error (MSE) of the random forest model across 10 runs for each hyperparameter value. The resulting plots for each hyperparameter can be found in the “code/Random Forest Tuning/tune result” folder on Github, including the graph below.

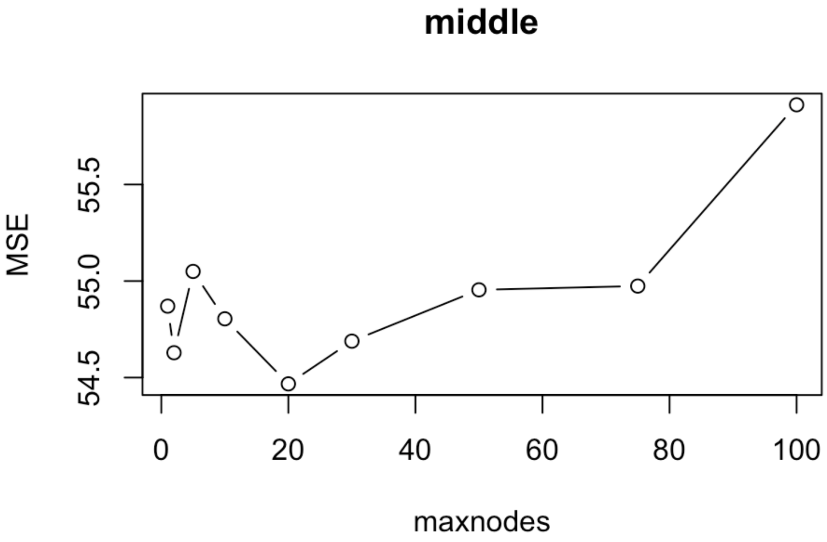
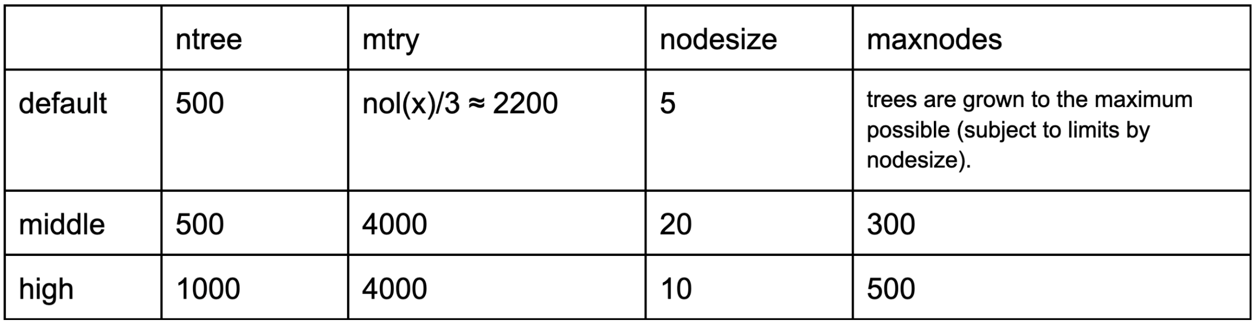


Figure 6

During our individual hyperparameter tuning process, we noticed that the average MSE did not exhibit significant changes as we modified the hyperparameter values (Figure 7). As a result, we decided not to attempt simultaneous tuning of multiple hyperparameters.

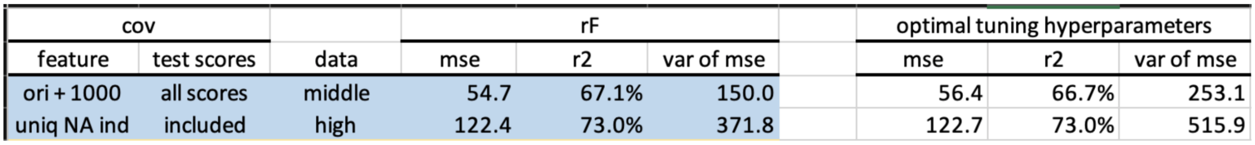


	ntree	mtry	nodesize	maxnodes
default	500	nol(x)/3 ≈ 2200	5	trees are grown to the maximum possible (subject to limits by nodesize).
middle	500	4000	20	300
high	1000	4000	10	500

Figure 7

After applying the optimal hyperparameter values obtained through tuning, we followed the same method as described in the blue section under Case1A and generated the table below (Figure 8).

The table indicates that the performance of the default random forest settings is comparable to that of the optimal hyperparameters, implying that using the default setting might be sufficient.



cov			rF				optimal tuning hyperparameters		
feature	test scores	data	mse	r2	var of mse		mse	r2	var of mse
ori + 1000	all scores	middle	54.7	67.1%	150.0		56.4	66.7%	253.1
uniq NA ind	included	high	122.4	73.0%	371.8		122.7	73.0%	515.9

Figure 8: Case1A&Random Forest Tuning result.xlsx

7 Conclusion

In conclusion, this report focused on auxiliary observational data from Texas schools that were not included in the randomized controlled trials (RCTs) of the CTA curriculum. We investigated various techniques for handling missing values and identified the Miss Forest algorithm as being computationally expensive with results comparable to simply replacing missing values with zeros. We also examined the choice of covariates and found that a subset of covariates was sufficient for producing a good result. While test scores were found to be the most important predictor, we also ranked the non-score predictors in terms of their impact on the outcome. Additionally, we utilized singular value decomposition (SVD) on missing indicator covariates and found it slightly improved the performance of the random forest prediction model. Although we attempted to tune neural networks, the random forest consistently outperformed all neural network models. Finally, we tuned the hyperparameters of the random forest and discovered that the default settings were effective.

For future work, it would be worth exploring the use of other models besides the random forest to potentially achieve better results. missForest and SVD could also be applied to the covariates to increase accuracy. Overall, we hope that this report will provide a promising approach for predicting outcomes of the RCT data with greater accuracy.

8 Citation

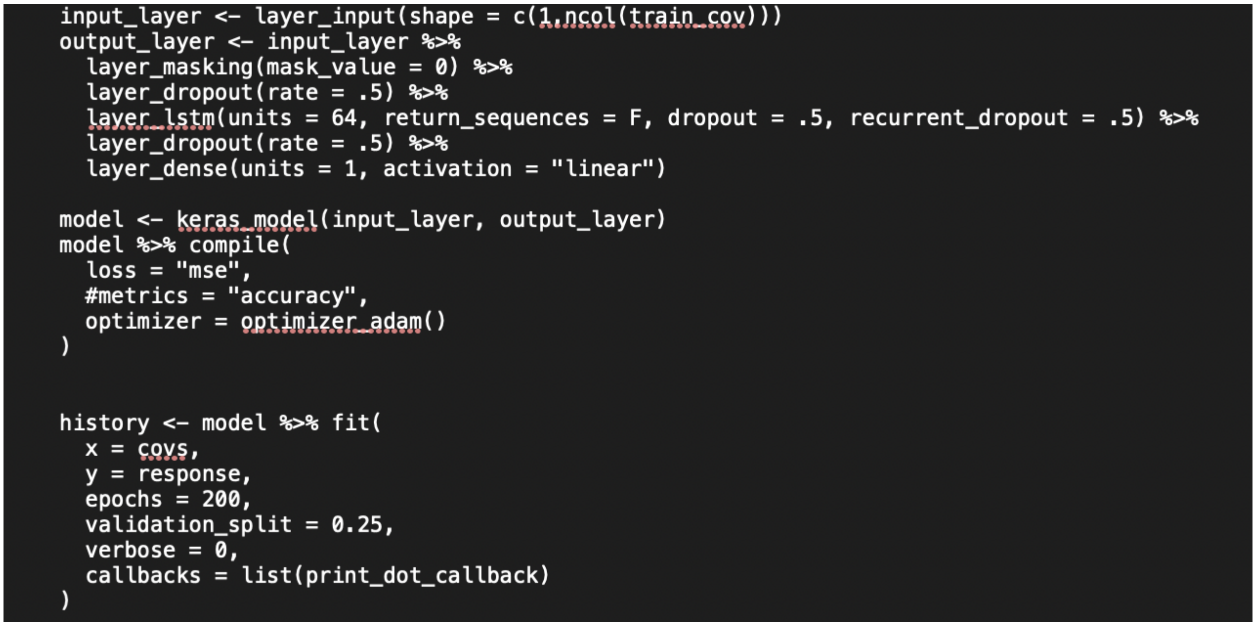
Pane, J. F., Griffin, B. A., McCaffrey, D. F., & Karam, R. (2014). Effectiveness of Cognitive Tutor Algebra I at scale. Educational Evaluation and Policy Analysis, 36(2), 127–144.

Saraswat, Manish. “Practical Tutorial on Random Forest and Parameter Tuning in R.” Hackerearth, n.d., https://www.hackerearth.com/practice/machine-learning/machine-learning-algorithms/tutorial-random-forest-parameter-tuning-r/tutorial/

Boehmke, Bradley and Brandon Greenwell. ”Hyperparameters”. Hands-On Machine Learning with R. 1 Feb. 2020, https://bradleyboehmke.github.io/HOML/random-forest.html#hyperparameters

9 Appendix

1. Neural Network setting for Figure 2 (this setting is random; copied from sample Neural Network Tutorial code)



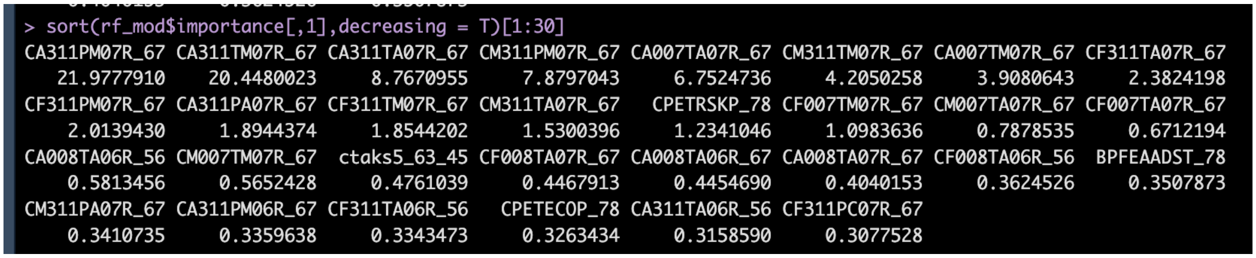
```
input_layer <- layer_input(shape = c(1,ncol(train_cov)))
output_layer <- input_layer %>%
  layer_masking(mask_value = 0) %>%
  layer_dropout(rate = .5) %>%
  layer_lstm(units = 64, return_sequences = F, dropout = .5, recurrent_dropout = .5) %>%
  layer_dropout(rate = .5) %>%
  layer_dense(units = 1, activation = "linear")

model <- keras_model(input_layer, output_layer)
model %>% compile(
  loss = "mse",
  #metrics = "accuracy",
  optimizer = optimizer_adam()
)


history <- model %>% fit(
  x = covs,
  y = response,
  epochs = 200,
  validation_split = 0.25,
  verbose = 0,
  callbacks = list(print_dot_callback)
)
```

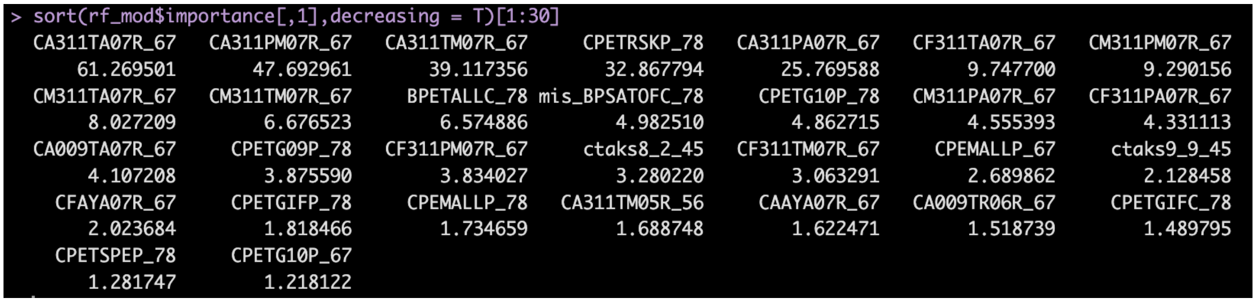
Figure 9

2. Case 2A top 20 important variables (from most important to less important). Majority of them are test scores for various kinds of exams. To see the specific meaning of these variables, go to Master Reference for Data Elements used in the AEIS (`https://rptsvr1.tea.texas.gov/perfreport/aeis/2008/xplore/aeisref.html`).



```
> sort(rf_mod$importance[,1],decreasing = T)[1:30]
CA311PM07R_67 CA311TM07R_67 CA311TA07R_67 CM311PM07R_67 CA007TA07R_67 CM311TM07R_67 CA007TM07R_67 CF311TA07R_67
   21.9777910    20.4480023     8.7670955     7.8797043     6.7524736     4.2050258     3.9080643     2.3824198
CF311PM07R_67 CA311PA07R_67 CF311TM07R_67 CM311TA07R_67   CPETRSKP_78 CF007TM07R_67 CM007TA07R_67 CF007TA07R_67
    2.0139430     1.8944374     1.8544202     1.5300396     1.2341046     1.0983636     0.7878535     0.6712194
CA008TA06R_56 CM007TM07R_67  ctaks5_63_45 CF008TA07R_67 CA008TA06R_67 CA008TA07R_67 CF008TA06R_56  BPFEAADST_78
    0.5813456     0.5652428     0.4761039     0.4467913     0.4454690     0.4040153     0.3624526     0.3507873
CM311PA07R_67 CA311PM06R_67 CF311TA06R_56   CPETECOP_78 CA311TA06R_56 CF311PC07R_67
    0.3410735     0.3359638     0.3343473     0.3263434     0.3158590     0.3077528
```

Figure 10: Middle School Importance Ranking



```
> sort(rf_mod$importance[,1],decreasing = T)[1:30]
  CA311TA07R_67   CA311PM07R_67   CA311TM07R_67     CPETRSKP_78   CA311PA07R_67   CF311TA07R_67   CM311PM07R_67 
      61.269501       47.692961       39.117356       32.867794       25.769588        9.747700        9.290156 
  CM311TA07R_67   CM311TM07R_67     BPETALLC_78 mis_BPSATOFC_78     CPETG10P_78   CM311PA07R_67   CF311PA07R_67 
       8.027209        6.676523        6.574886        4.982510        4.862715        4.555393        4.331113 
  CA009TA07R_67     CPETG09P_78   CF311PM07R_67     ctaks8_2_45   CF311TM07R_67     CPEMALLP_67     ctaks9_9_45 
       4.107208        3.875590        3.834027        3.280220        3.063291        2.689862        2.128458 
    CFAYA07R_67     CPETGIFP_78     CPEMALLP_78   CA311TM05R_56     CAAYA07R_67   CA009TR06R_67     CPETGIFC_78 
       2.023684        1.818466        1.734659        1.688748        1.622471        1.518739        1.489795 
    CPETSPEP_78     CPETG10P_67 
       1.281747        1.218122 
```

Figure 11: High School Importance Ranking