Projecting K-5 Enrollment in New York City Public Schools

Michael Fairley[1] and Deepti Mahajan[2]

***Abstract*— We sought to use machine learning algorithms to predict K-5 school enrollment counts across census tracts and across years within Community District 20 in New York City. We used census data and historical counts as our input, and output enrollment count predictions for grades, census tracts and future years.**

I. Introduction

In recent years, the city of New York has faced population growth unseen since the 1920s, about a 4.6% change since 2010 [1]. This poses a challenge for the NYC Department of Education (DOE), as they work to provide enough resources for every public school student. In city planning and school district rezoning, enrollment numbers are a vital statistic. Specifically, the number of students from each part of a district is needed for allocating resources. Thus, the NYC DOE is providing enrollment figures for Kindergarten through grade 5 from each census tract in Community School District 20 20 (which resides in Brooklyn) between the 2001-2002 and 2010-2011 school years as a public challenge to create a model that can predict enrollment figures for the years 2011-2012 through 2016-2017.

II. Related Work

In the past, the NYC DOE has used a "cohort-survival model" to project enrollment data. This model involves using enrollment data from past years, birth rate, population projections, and changes in housing in specific calculations [2]. However, this model does not provide the required precision, especially at the census tract level. Thus, alternative methods of modeling enrollment from each census tract were needed; our goal was to apply machine learning methods to improve these enrollment predictions.

The cohort-survival model is a simple projection model that computes the ratio of students that survive from one grade to the next averaged over a given number of years [3]. Equation (2) shows how this method can be used to make a projection (n is typically 5 years). For the Kindergarten grade, births from 5 years prior are used to compute a survival ratio. The method is most suitable when trends in school enrollments are relatively stable over time. This method fails when $e_{gt} = 0$, which often happens at the census tract level.

$$s_g = \frac{1}{n}\sum_{t=1}^{n}\frac{e_{gt}}{e_{g-1,t-1}} \tag{1}$$

$$e_{g,n+1} = s_g e_{gn} \tag{2}$$

[1]Department of Management Science and Engineering
[2]Department of Electrical Engineering

Census Tract	School Year	Grade	Student Count
80	20102011	4	38
80	20102011	5	31
81	20012002	K	4

TABLE I
Example enrollment data

III. Dataset and Features

A. Enrollment Data

The enrollment data was structured in the format in Table I.

B. 2010 Census Data

We obtained publicly available 2010 census data, specifically, *Table DP-1(Profile of General Population and Housing Characteristics: 2010)*. These data contained demographic information about each census tract within Kings County, New York, which contains School Community District 20. This includes information such as the total population, population by age, by race and the number of households by type. In total there were 184 variables available in this dataset, however we pre-processed the dataset by removing duplicated variables, variables having the same value for every census tract, near-zero variance variables, and linearly dependent variables. This reduced the number of variables to 118. We then centered and scaled all of the variables.

IV. Methods

A. Demographic Indicators

In order to determine if demographic indicators could predict school enrollment numbers, we took the enrollment data for 2010 and split up the census tracts into a training and test set (70%/30% split). We then used various regression models to predict enrollments based on the 2010 census data.

1) Random Forest: Random forest is an ensemble learning algorithm that combines the predictions of many decision trees. Decision trees is a supervised learning method that predicts the outcome variable by using simple decision rules on the predictors; Figure 1 shows an example of such decision rules. A decision tree on its own is prone to overfitting, but random forests can help to reduce this issue by combining many decision trees together [4]. The random forest method uses a training method called *bagging*, whereby the training set is sampled with replacement into B parts and a decision tree is trained on each one, and finally new predictions are made by taking the mean of the predictions from all of the generated decision trees. For each of these bags, a random subset of features is chosen to generate the decision tree [6].

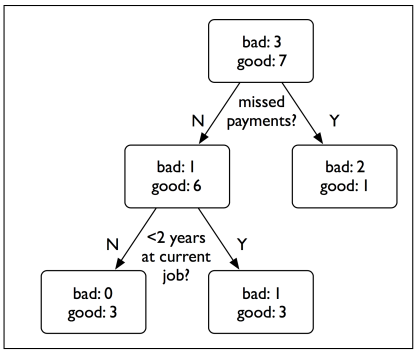

Fig. 1. Example decision tree [5]

An important parameter of the random forest is `mtry`, which is the number of features to randomly select at each split.

We used the `caret` package in R to train a random forest model for each grade, and used "out-of-bag" error estimation, which replaces the need for cross-validation. For each split, some of the training data is left out and an error estimate is obtained by running each case through the tree generated at the split [7].

The training and test errors for each grade are shown in Table II, and a plot of prediction vs. actual enrollment counts is shown in Figure 2

Grade	Train		Test		
	`mtry`	R^2	RMSE	R^2	RMSE
K	118	0.926	6.754	0.6986	9.3880
1	60	0.9119	7.6576	0.7486	10.5670
2	60	0.9229	7.3128	0.7759	7.1703
3	2	0.9189	7.1780	0.7207	10.6611
4	60	0.9231	5.9423	0.7723	8.8016
5	60	0.9319	5.7485	0.682	10.333

TABLE II
MODEL ERRORS

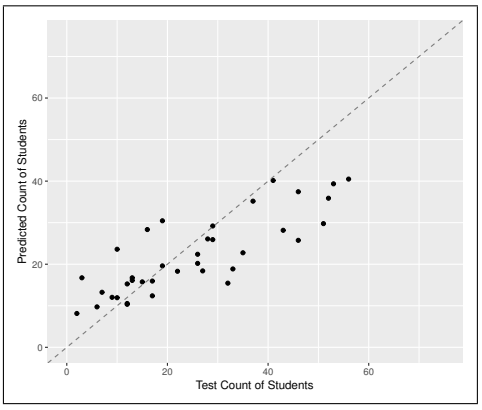

Fig. 2. Random Forest Kindergarten error on test set

2) Random Forest Variable Importance: In random forests, for each variable, the MSE is computed on the out-of-bag data for each tree both with and without the variable. The difference in these errors is then averaged and normalized by the standard error and provides the variable importance measure [8]. Figure 3 shows the top 5 variables for the Kindergarten random forest model, where Table III provides a description of each variable.

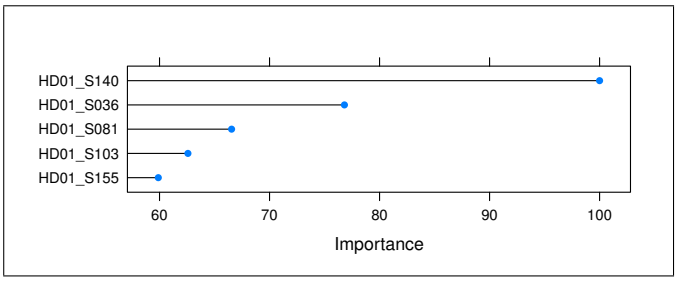

Fig. 3. Random forest variable importance for Kindergarten model

Census variable code	Description
HD01_S140	Number, RELATIONSHIP, Total.population, In.households, Nonrelatives, Under.18.years
HD01_S036	Number, SEX.AND.AGE, Male.population, 45.to.49.years
HD01_S081	Number, RACE, Total.population, One.Race, Asian
HD01_S103	Number, RACE...Race.alone.or.in.combination with one.or.more.other.races, 4, Asian
HD01_S155	Number, HOUSEHOLDS.BY.TYPE, Total.households, Family.households..families, 7, Male.householder, no.wife.present

TABLE III
MODEL ERRORS

3) Partial Least Squares: Partial least squares is a learning method that deals particularly well when there are many correlated features in a dataset, and a small number of observations. It can also perform multivariate regression (when there is more than one outcome variable), which is exactly the situation we have with multiple grades to predict. The method can explicitly account for the correlation between the outcome variables. As motivation for using this method, consider Figure 4, which shows that the grades are highly correlated with each other.

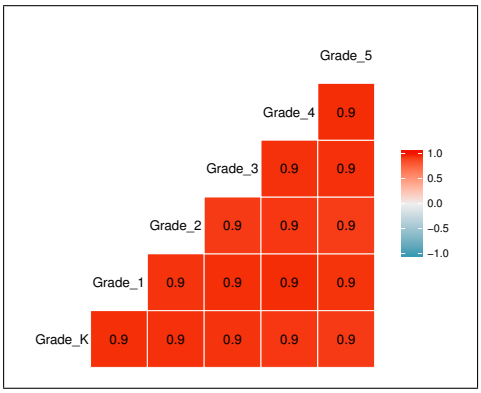

Fig. 4. 2010 enrollment count correlation between grades

In a normal multiple linear regression model, we model

the outcome variables as so:

$$Y = XB + \varepsilon \tag{3}$$
$$B = (X^\top X)^1 X^\top Y \tag{4}$$

where ε is a Gaussian error term with constant variance [9]. The problem with this approach is that $X^\top X$ is often singular because there are more rows than columns or because of collinearities [10]. Indeed, we faced this problem when attempting linear regression and had to remove highly correlated features before the model could solve. Partial least squares deals with this by dimension reduction, by decomposing X into orthogonal scores T and loadings P [10].

$$X = TP$$

To accomplish this, we obtain components iteratively, first we take the singular value decomposition of $S = X^\top Y$ and use the first left and right singular vectors, w, q as weight vectors for X and Y respectively to obtain scores t and u.

$$t = Xw = Eq$$
$$u = Yq = Fq$$

where E and F are initialized to X and Y. We normalize the t scores for X: $t = \frac{t}{\sqrt{t^\top t}}$. We obtain the loadings by

$$p = E^\top t$$
$$q = F^\top t$$

Then we deflate E and F.

$$E_{n+1} = E_n - tp^\top$$
$$F_{n+1} = F_n - tq^\top$$

We repeat this procedure starting from the singular value decomposition of $E_{n+1}^\top F_{n+1}$, and save the vectors w, t, p, q in the matrices W, T, P, Q in each iteration. We then represent the weights as:

$$R = W(P^\top W)^{-1}$$

We then calculate regression coefficients that can be used in Equation (3).

$$B = R(T^\top T)^{-1} T^\top Y = RT^\top Y = RQ^\top$$

The number of components to use (ie. the number of iterations to use in the above method) is a parameter of the model and can be optimized by cross-validation.

We use the R `pls` package to perform partial least squares regression with Y being a matrix, with each column representing a grade (K-5) and each row representing a census tract. We used leave-one-out cross validation to determine the number of components. Figure 5 shows that 3 is the optimal number of components. Figure 6 shows the predictions on the test set for each grade, and Table IV shows the associated errors.

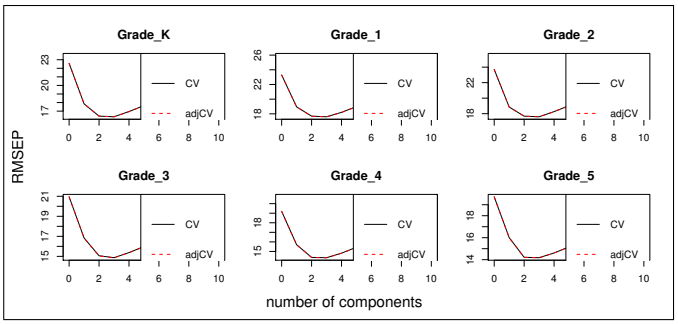

Fig. 5. Leave one out cross validation to optimize number of components

Grade	Train RMSE	Test RMSE
K	13.77	8.755
1	14.97	10.99
2	14.93	8.453
3	12.52	10.75
4	12.22	9.652
5	12.24	10.81

TABLE IV
MODEL ERRORS

B. Enrollment Projections

1) Linear Regression: Unfortunately, we could not incorporate the relevant demographic data into a predictive model, as data at a census tract level was not available for years between the 2000 and the 2010 Census. Thus, we decided to try predicting future enrollment numbers using only the provided historical enrollment data. As a preliminary model, we used linear regression. We trained the model with 2001-2009 data as the input and 2010 enrollment as the output, using 70% of the census tracts for training and 30% for testing (randomly chosen).

2) Neural Network: In an effort to improve upon the linear regression predictions, we implemented a feed-forward neural network to capture any non-linear trends that may have existed in the data and could not be modeled by a linear regression. A feed-forward neural network is a model consisting of multiple neurons or units. Specifically, we implemented a multilayer perceptron (MLP), in which the neurons are arranged in layers, with an input layer, one or more hidden layers, and an output layer.

Each neuron takes in a certain number of inputs, each multiplied by some weight w_{ij} and sums them. The net input is put through some nonlinear, differentiable activation function to yield the neuron output. We can express these transformations with the following equations, where a_j is the net input, ϕ is the activation function, and o_j is the neuron output:

$$a_j = \sum_{i=1}^{n} w_{ij}^{(1)} x_i + 1$$
$$o_j = \phi(a_j)$$

When training a neural network, the weights w_{ij} are opti-

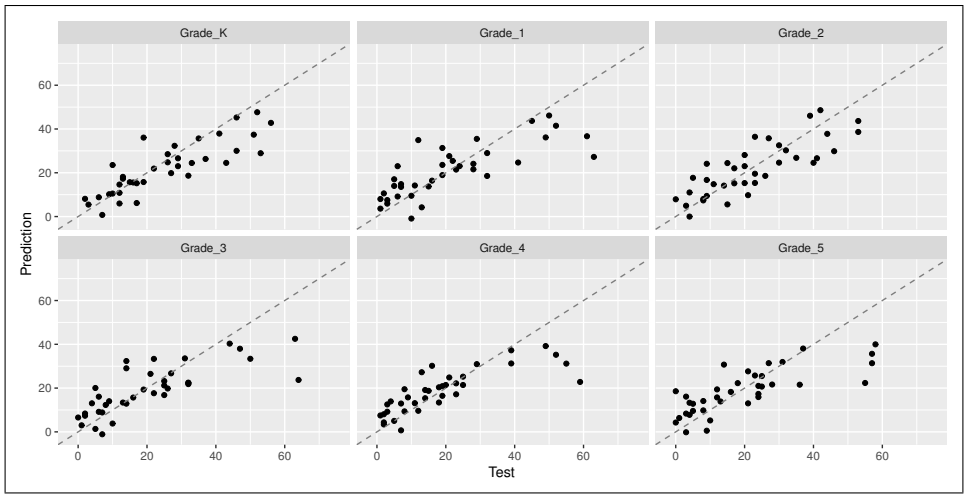

Fig. 6. Partial least squares error on test set

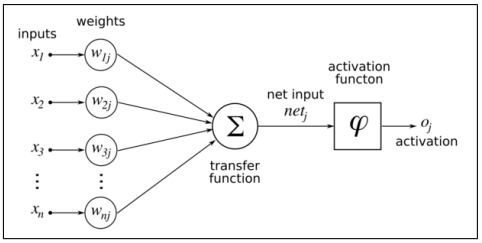

Fig. 7. Neuron Model

mized by finding the error gradient and iteratively adjusting the weights. This algorithm is called error backpropagation. Our neural network had an input layer with the number of input neurons equal to dimension of the input data. We tested different numbers of hidden layer nodes to optimize the MLP. As we were applying a neural network to a regression problem, we chose to use an output layer of one neuron with no activation function. We also chose to use a rectifier activation function for the input and hidden layers.

To train the network, the time-series data was divided into census tract examples consisting of 1 to N years enrollment for a particular grade as the input, and the $(N+1)^{th}$ year's enrollment as the output. We used the prediction of the final year (2010-2011) as our test set, as the remaining 9 prior years of data to be used for training. The network was trained with specified parameters: the grade level of interest, the year to use as test data, the number of years of historical data to use for prediction, and the number of years ahead to predict. Each census tract acted as one example for training the network, and Thus, the number of available training examples depended on the specified parameters, as we only had 10 years of data to use.

3) Recurrent Neural Network: We were also interested in the results that a recurrent neural network (RNN) would yield. RNNs take in an input value at one point in time, but retain information from previous time inputs. As shown in the figure, an intermediate value of the neuron is saved and used in calculating the output of the next time step. Thus, RNNs have some semblance of memory. However, since previous time steps data goes through multiple rounds of multiplication by a sigmoidal function before the current time step, the error gradient either vanishes or explodes. The gradient is needed for error minimization, which is important for training the network, so as a result it is difficult to data from multiple prior time steps to impact the current network output. There is a type of RNN that circumvents the vanishing gradient problem, called a Long Short-Term Memory Network (LSTM). LSTMs consist of memory blocks, which contain information from previous time steps in a gated cell-like structure. This cell has 3 nonlinear gates, which determine if the current input should be added to the stored state, the stored state should be erased, or the stored state should be used in the current time steps output, respectively. The inputs to the gates include the current input and past information, and the gates make the determination based on the weights of its inputs, which are learned parameters. The advantage of using an LSTM lies in its ability to learn long-term dependencies in the data–in our case, there may have been some sort of dependency of current enrollment numbers on the enrollment many years before. As with the (nonrecurrent) neural network, the time series data was divided into N-length windows of consecutive years data as the inputs, and the $(N+1)^{th}$ year enrollment as the output. The data was scaled to a range of [0,1] and used to train an LSTM with 4 blocks for 100 epochs, using a mean squared error loss function.

V. Results

The MLP was tested with different neuron topologies. Figure 8 shows the RMSE of the test set with varying numbers of hidden layer neurons. The configuration with 4 neurons in the hidden layer yielded the lowest error (RMSE=5.43).

Another parameter that was optimized in the MLP was the input window size, or the number of previous years data used as a model input. Figure 9 shows the effect of window

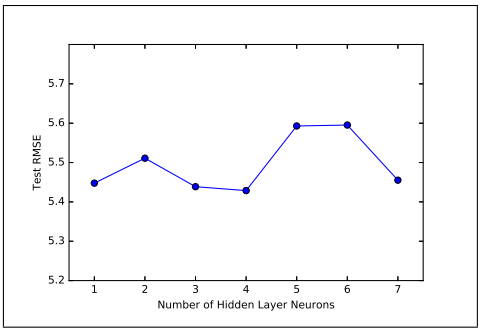

Fig. 8. Test RMSE vs. Number of Neurons in MLP Hidden Layer

size on the test data RMSE. A window size of 8 for the one-year prediction was optimal, with an RMSE of 5.43.

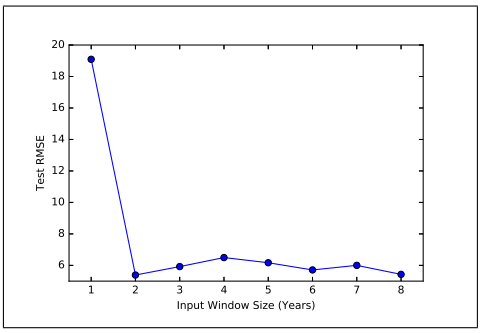

Fig. 9. Test RMSE vs. Input Window Size in Years in the MLP

The LSTM was also optimized by testing the input window size. An example of the results in Figure 10 shows that a window size of 8 for one-year predictions was optimal for the LSTM as well as the MLP.

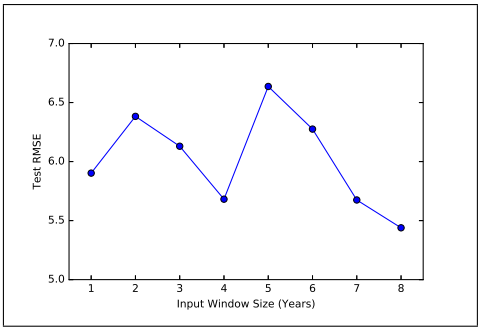

Fig. 10. Test RMSE vs. Input Window Size in Years in the LSTM

Once the optimal parameters were found, the three methods were compared. Table V contains the results for one-year predictions of 2010 5^{th} grade data.

Figure 11 provides a visual representation of the performance of the MLP on the test data, with each dot representing one census tract.

It was also important to compare the performance of the models trained to predict five years ahead. Table VI contains the results of making five-year predictions of 5^{th} grade enrollment in 2010.

Method	Train RMSE	Test RMSE
Linear Regression	4.54	6.55
Multilayer Perceptron	4.50	5.43
LSTM	4.46	5.44

TABLE V
ONE YEAR PREDICTION ERRORS

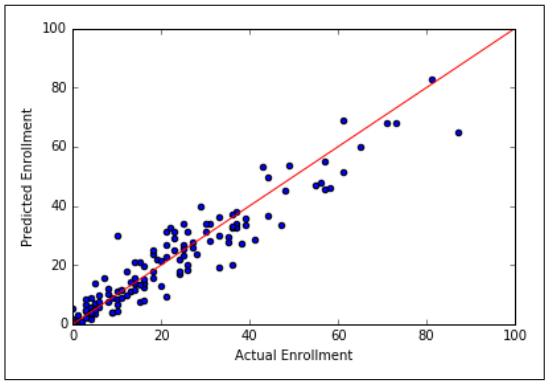

Fig. 11. MLP One Year Predicted vs. Actual Enrollment

VI. Conclusion

We showed that demographic data can be used to predict enrollment numbers with random forests or partial least squares; however these methods still have high test error, which could be improved by incorporating other features and using historical enrollment counts.

Out of the three methods used to predict enrollment numbers from only historical data (Linear Regression, Multilayer Perceptron, and the LSTM RNN), the MLP and LSTM had very similar results, and were consistently better than Linear Regression when applied to test data (predicting 2010 enrollment). The similar performance of the LSTM and MLP may be due to the fact that the LSTM was trained on multiple examples of the same set of time intervals. Since LSTMs (and RNNs in general) store information from previous inputs, much of the stored input may not have been unnecessary and could have even reduced the overall performance. In the future, we would like to explore using machine learning alongside the pre-existing cohort-survival model.

Method	Train RMSE	Test RMSE
Linear Regression	5.96	7.42
Multilayer Perceptron	4.97	6.41
LSTM	4.93	6.35

TABLE VI
FIVE YEAR PREDICTION ERRORS

REFERENCES

[1] "New York City Population Projections by Age/Sex and Borough, 2010-2040," New York City Department of City Planning, Tech. Rep., 2010.

[2] *NYC DOE Call for Innovation*, 2016. [Online]. Available: http://www.nyc.gov/html/cfi/html/DOE/index.html.

[3] S. F. LLC, "The Cohort Survival Ratio Method," Tech. Rep.

[4] Tin Kam Ho, "Random decision forests," *Proceedings of 3rd International Conference on Document Analysis and Recognition*, vol. 1, pp. 278–282, 1995, ISSN: 0818671289. DOI: 10.1109/ICDAR.1995.598994. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=598994.

[5] Michael S. Lewicki, *Artificial Intelligence : Introduction to Learning & Decision Trees*, 2007. [Online]. Available: http://www.cs.cmu.edu/afs/cs/academic/class/15381-s07/www/slides/.

[6] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001, ISSN: 08856125. DOI: 10.1023/A:1010933404324.

[7] *Random forests - classification description*. [Online]. Available: https://www.stat.berkeley.edu/%7B%5C~%7B%7D%7Dbreiman/RandomForests/cc%7B%5C_%7Dhome.htm%7B%5C#%7Dooberr.

[8] *The caret Package*. [Online]. Available: https://topepo.github.io/caret/variable-importance.html.

[9] A. Ng, *CS 229 Lecture Notes Supervised learning*.

[10] B. Mevik and R. Wehrens, "The pls Package: Principle Component and Partial Least Squares Regression in R," *Journal of Statistical Software*, vol. 18, no. 2, pp. 1–24, 2007, ISSN: 15487660. DOI: 10.1002/wics.10. [Online]. Available: http://www.jstatsoft.org/v18/i02/paper.