Final Project

CSI-531 Data Mining

Caroline Bablin, Steven Caraballo, Oren Merberg

1 Introduction

While there is a large amount of Massive Open Online Courses, or MOOC's, there is also a high dropout rate. The structure and flexibility of these courses is appealing to many, yet so many students do not complete the course. Prediction of student dropout in MOOC's has been a widely studied problem, with the intention that being able to predict early on would allow for intervention with the student to prevent them from dropping out. While this is an important goal for many, analyzing what factors are closely related to student success could provide a deeper understanding of the leading causes for dropout. Our goal is to discover these factors and include them in various modeling techniques and see if this leads to better dropout prediction.

2 Regression

One method that was hoped to be implemented was a regression technique. Before implementing any sort of regression, it was necessary to inspect the data. The data used was from XuetangX, a major MOOC platform in China, as used in the KKD Cup 2015. The goal of using this data was to predict whether a student will drop out of a course within the next 10 days based on previous student behavior. The structure of the data, that is whether variables were numerical or categorical, and the distribution of the variables is a key first step before implementing regression. Starting with 140 variables, it was clear not all of them could be used in regression. In theory, there is nothing against using all of these variables, but in practice it would be both time consuming and even perhaps creating a model that overfits the data. Also, regression models with fewer variables are preferred over those with many.

Upon inspecting the available data, it was clear there were several categorical variables. Notably, the prediction label was a 0 or 1, denoting whether a student dropped out of the course or not, signaling that this was a categorical variable. Due to this, logistic regression would be used instead of linear or polynomial regression. There were also several variables that would not be useful in terms of analyzing, such as enrollment ID's. These kinds of variables serve as an identification for the student, and would not yield meaningful results, so they were removed. To work with the remaining categorical variables, we would need to treat them as factors, that is each category corresponds to an number, say 0, … , n. However, quickly glancing at some of the columns it appeared that some variables took the same value for every observation. To check this, boxplots were made of these variables. For many, it was true that each

observation took the same value as the boxplot was merely a line. For these variables that took the same value, they were removed since a constant term in each observation would not change the regression outcome. For other variables, boxplots showed the distribution of the data was in some cases very vast. It appeared for some variables there were many outliers that lied very far from the rest of the data. It raised the question whether some of these values were entered with an error or if they indeed were that value, as it almost seems improbable that some of the values entered were real. If these values would significantly impact the performance of the model, an option would be to remove these values.

After removing the aforementioned variables, there were still many variables left. One approach that could have been taken would be to create a model using all of the remaining variables, and then remove them one by one based on the significance of the variable in the model. Inspecting p-values for each variable could signal that a variable is significant or not, where those variables with high p-values being insignificant. However, with somewhere along the lines of 130 variables remaining, this would again be a tedious process. A model would have to be created and then the variable with the highest p-value would be removed, and this would be repeated until a model was reached with all p-values for variables being small or until other measures were reached. This process can be efficient with some cases, but here it would not be appropriate.

As a secondary approach, the data was even further inspected before making any models. First, the data was centered so each of the remaining variables had a mean of 0 and a standard deviation of 1. This makes it easier to compare distributions across the variables. Next, to determine which variables to use in the model, a correlation matrix was computed for all variables, including the labels. It was hoped that by inspecting the correlation between each of the variables and the label, it would give an idea of which variables were most highly correlated with the prediction label. While it is good to have variables that are correlated to the prediction label, it is also important to consider the correlation between the variables themselves. Variables that had a high correlation between themselves could impact the model, a phenomenon known as multicollinearity.

While the method above was considered, it was clear that there were still too many variables to pick apart one by one. As a sort of dimension reduction and feature engineering, the columns corresponding to the daily activity of students were split into four weeks. The columns corresponding to the daily activity of day 1 through day 7 were week 1, and so on. These columns were averaged across each student, so each student now had one value for their activity in each week of the course. It was

hypothesized that there may be a relationship between the student activity between any two of the weeks. The idea was that students who are very active in one week may be more likely to be as active in the following weeks. To test this hypothesis, scatter plots were created between weeks and the correlation between all weeks was also calculated. The scatter plots did not look very linear, instead there was a cloud of points, but with very high density around the origin. The correlation between weeks also showed that there was little correlation between weeks. This could signal that student activity may not be a significant variable to helping to predict student dropout. With these results, we turned to the other variables.

The next set of variables that were inspected were browser problem and parallel enrollments. The intuition behind using these variables was that if a student encountered a browser problem, they would be less likely to continue with the course. Similarly, if a student is in multiple courses at once, it would be expected that their participation in another course may be less. Plotting these values against the true values, it appeared that for browser problem there were many more true labels of 0, meaning the student dropped out. This was especially true when there were higher values for browser problems. With parallel enrollments, the result was not as clear. Looking at the correlation with each of these variables with the true label showed that parallel enrollments was not highly correlated with the label while browser problems were slightly correlated.

With this data inspection done, a base model was created. To start, a model with the student activity in each of the 4 weeks were used as variables. Then models with browser problems and both browser problems and parallel enrollments were created. These three models were compared by inspecting the p-values of each of the variables in the model, the residual deviance, and the AIC. Concerning the p-values, those variables with small p-values are deemed significant in the model. Across each of the models, the p-values for each of the variables was nearly 0, meaning they are significant in the model. Due to this, other methods of comparison needed to be used to compare the models. The residual deviance, which gives an idea of how well the model performs. The smaller the deviance signals a better model, as this value compares the actual value of the prediction to the predicted value. In a perfect model, the deviance would be zero. For the first model with the variables of student activity in each week, the residual deviance is 60390. This is quite high but adding the variables of browser problem and parallel enrollments one by one decreases the deviance to 60031. While this value is still very high, the decrease in deviance shows that the model with the extra two variables is better. Comparing the AIC, which estimates prediction error, it is clear again that the model with the added variables is better as the AIC decreases from 60400 to 60045. While this is a good sign that the third model is better than the first,

these values for both residual deviance and AIC are still very high. With further work, it is hoped that a better model than the previous ones could be produced.

```
> summary(base_mod)

Call:
glm(formula = label ~ mean_week1 + mean_week2 + mean_week3 +
    mean_week4, family = "binomial")

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.0904   0.4886   0.4967   0.5416   6.1217

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  1.42940    0.01048  136.40   <2e-16 ***
mean_week1  -0.85630    0.02393  -35.79   <2e-16 ***
mean_week2  -0.92448    0.02618  -35.31   <2e-16 ***
mean_week3  -1.13962    0.02501  -45.58   <2e-16 ***
mean_week4  -0.64678    0.01807  -35.79   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 73952  on 72394  degrees of freedom
Residual deviance: 60390  on 72390  degrees of freedom
AIC: 60400

Number of Fisher Scoring iterations: 5
```

```
> summary(mod3)

Call:
glm(formula = label ~ mean_week1 + mean_week2 + mean_week3 +
    mean_week4 + browser_problem + parallel_enrollments, family = "binomial")

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.1135   0.4761   0.4981   0.5444   6.2357

Coefficients:
                     Estimate Std. Error z value Pr(>|z|)
(Intercept)          1.5579210  0.0127326  122.36   <2e-16 ***
mean_week1          -0.7228845  0.0256560  -28.18   <2e-16 ***
mean_week2          -0.7727289  0.0284997  -27.11   <2e-16 ***
mean_week3          -1.0498270  0.0260845  -40.25   <2e-16 ***
mean_week4          -0.6171404  0.0182274  -33.86   <2e-16 ***
browser_problem     -0.0087711  0.0006252  -14.03   <2e-16 ***
parallel_enrollments -0.1033165  0.0090237  -11.45   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 73952  on 72394  degrees of freedom
Residual deviance: 60031  on 72388  degrees of freedom
AIC: 60045

Number of Fisher Scoring iterations: 6
```

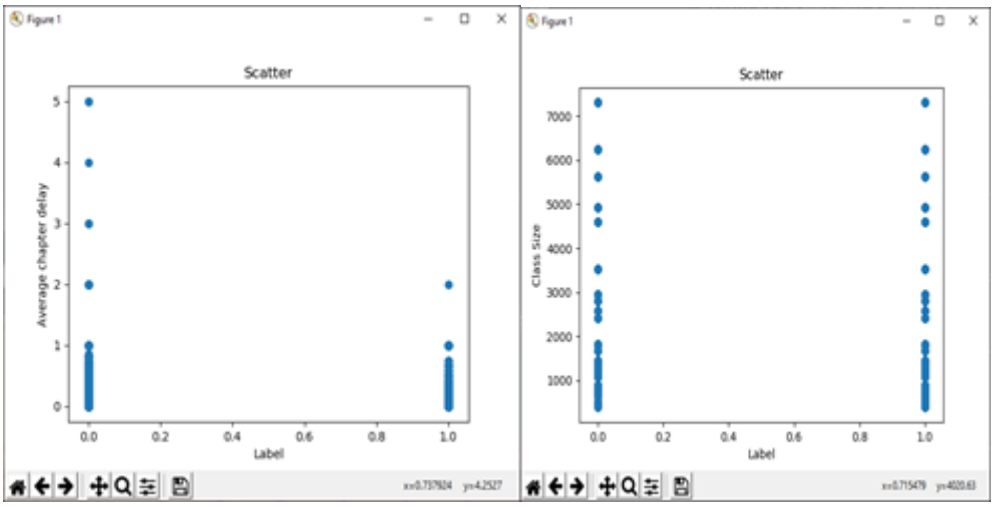
With future work, it would be worthwhile to contact the creators of the data and ask for a better description of some of the variables. Issues that came up along the way mainly had to do with the description of the data. There was no great explanation of what each of the variables were, so we used our best judgement given the column names. Another issue was figuring out what exactly the values in the columns corresponded to. It is possible some of the variables were indeed categorical, but just written as factors, that is each category corresponded to a specific integer. If this was the case, it would be necessary to include this information in the regression. By looking at boxplots, it was clear there were many outliers in variables. However, it was not clear if the outliers were real data or an error in entering the data, as many laid several standard deviations from the mean. These values simply did not seem plausible. Removing, or even replacing these values with another, say the mean, could alter the model and make it perform better. Lastly, including or excluding variables in new models may lead to new discoveries.

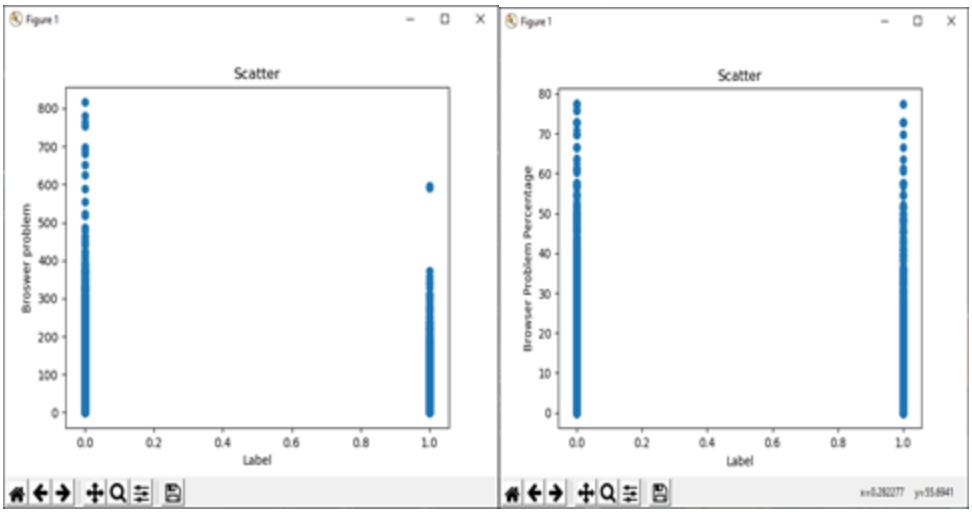
3 Clustering Algorithms

We were interested in using clustering algorithms as a method of detecting correlation throughout XuetangX. The original data set is composed of 141 features with over 72,000 entries. To help save time and memory, we had reduced the data down to 5 features to review the correlation and possible effect of each feature. The 5 features that were decided on were Label (Boolean related to student dropout), Average chapter Delays(The average delay a student shows in the chapters within the course) , Browser problem(The amount of connectivity issues) , Browser Problem Percentage(The percentage which a browser issue occurs for the given student) , Class Size (The amount of student enrolled in the course). These features were selected as a part of an exploratory analysis as we believed that these features would have a clean impact on a student's success or failure.

3.1 K-means Clustering

For this experiment, we decided to use K-mean clustering for exploratory analysis of the 5 features. Using the Matplotlib in Python, we were able to visual the data in a 2-dimensional scatter plot. Below are the 4 plots.





One apparent observation is that each scatterplot displays the data in the y axis in only 2 locations on the X axis which are 0 and 1. This is due to the Label data being a binary data type as it determines whether a student dropped out or not. The valuable information comes from the comparison between the values located on the 0 and 1 point on the x axis as the variation highlights areas of interest.

3.1.1 Objective Function

We decided to use an objective function to determine the number of clusters that is present in each feature comparison. We decided to experiment and use the number of within cluster Sum of Squares and the correlation between the number of clusters. The experiment resulted in the graph below.

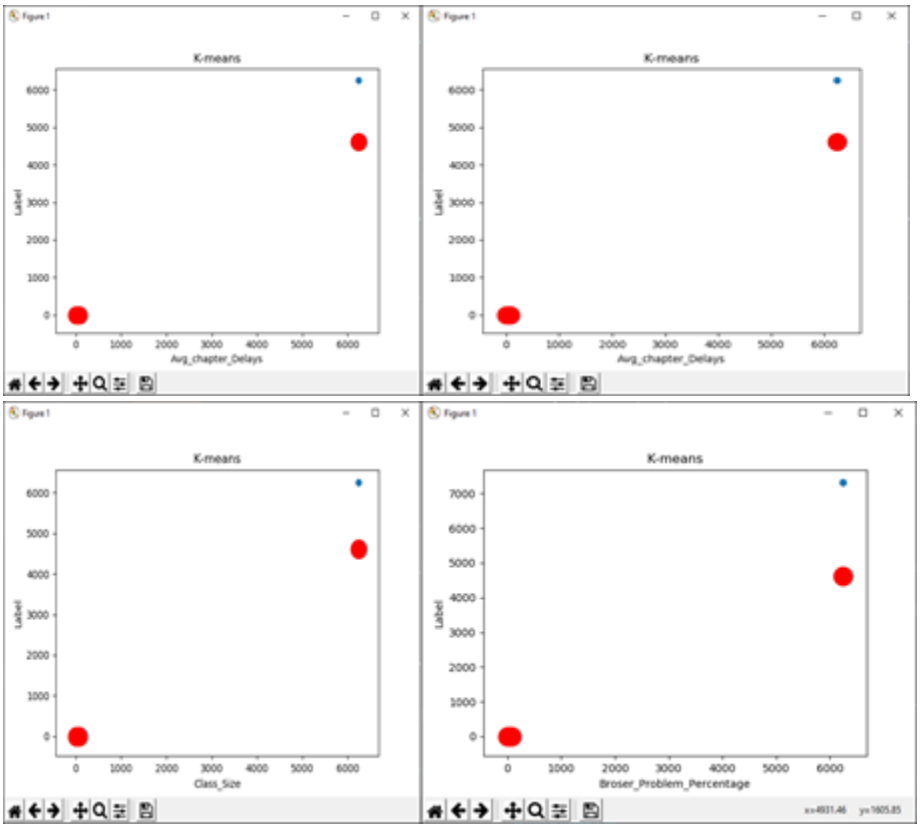
The graph visualizes the relation the sum of squared distances for all points within a cluster and its center point. We can see the largest bend occurs at 2 clusters. This makes sense as the data is being processed has an X axis which only contains binary data. We were working on implementing a multi-level K-means Objective Function which would isolate each of the binary datasets and run sum of squared distances for all points within a cluster dependent on the student either dropping out or not. This would allow for an in-depth comparison of each selected feature in relation to a student dropping out. With that determination, we could explore the effect of these features in additional MOOC data sets which contain the same information.

3.1.2 K-mean Results

After we reviewed that the data was being split into 2 clusters based on the Binary data, we had decided that Implementing 4 clusters would allow for insight into each binary classification as for each cluster specified in the objective function would be split into 2

clusters. The goal was to extract metrics based on the difference in each cluster center for students that dropped out and students who did not.

The graphs below display the data points as blue and the red point determine the center of the cluster. It became apparent that there were issues in the data as it seems the clustering method was not as robust as assumed.

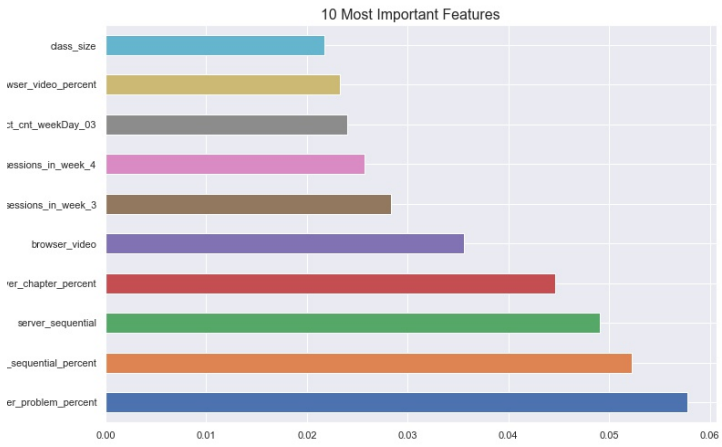


4 Decision Tree Classification

4.1 Dimension Reduction

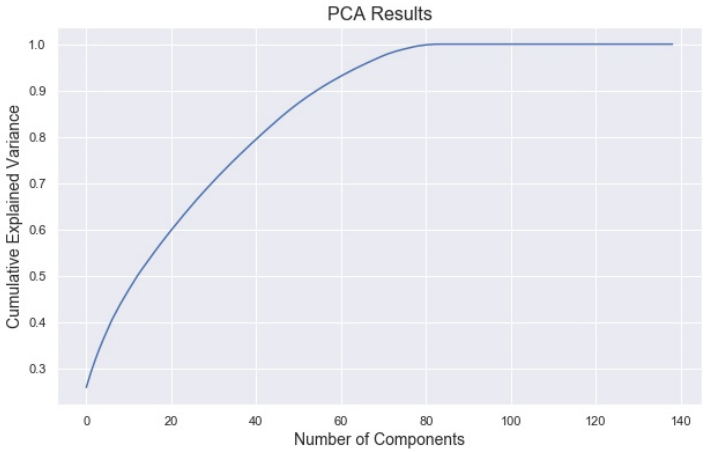
4.1.1 Feature Exploration

We would also like to explore decision trees as a classification method for predicting students that drop out versus students that finish the entire course. The dataset contains 141 features and 72,000 observations. The feature space is probably too large to collect insightful information, so we look at this first. We first analyze the feature importance using Extra Tree Classifier from sklearn's ensemble module. This algorithm implements random decision tree estimators on sub-samples of the dataset and detects which features have the greatest effect on the predictor variable (1). Figure 1 shows the results from this method. It is clear that not many of the features play an important role with the "most important feature" only getting a value of 0.06 or 6% out of the 141 features. This is evidence that some form of dimension reduction is necessary to get rid of extraneous variables. Reducing dimensions can help determine features that provide no information for prediction and can also isolate collinear features, which will in turn prevent overfitting. With this in mind, we turn to principal component analysis to reduce dimensions.



4.1.2 Principal Component Analysis

Principal component analysis (PCA) is one of the most common methods of dimension reduction. The aim of PCA is to reduce as many variables as possible while retaining the most information. In this way, the data is much easier to understand which is especially useful for our dataset that contained numerous features that were difficult to interpret.



To transform the data to principal components, we first want to know the optimal number of components to keep. This is often done by comparing the cumulative explained variance to the number of principal components. The general rule is to keep the explained variance around 95%. The graph in Figure 2 shows the curve for our dataset. We can see that anything after 80 components keeps almost 100% of the explained variance. To keep 95% explained variance, we would need about 60 to 65 components, so we choose to keep 60 components for this analysis.

4.2 Classifier Evaluation

4.2.1 Hyperparameter Tuning

With the data reduced, we can now implement sklearn's decision tree classification method. First, it is a good idea to tune the parameter to find the optimal setup. We consider the maximum number of leaf nodes and the cutoff criterion. It is best practice to withhold an additional training subset for cross validation, so we implement 10-fold cross validation to train the classifiers. To optimize model settings, we use the information gain (entropy) and Gini index criteria along with maximum leaf node values of 2, 5, 10, 20, and 30. The results are shown in

Figure 3. We can clearly see that the Gini index provides the best results. The optimal number of leaf nodes looks to be either 10 or 20, so we use 20 for our model. Anything above that would likely lead to overfitting.

4.2.2 Results and Future Work

We finally run the decision tree model using Gini index criterion, a maximum of 20 leaf nodes, and with 10-fold cross validation. To evaluate the model, we show the confusion matrix and calculate the average precision score, recall score and F score. The data is shown on the following page.

Confusion Matrix	
2668	2310
964	17949

Average Precision	0.88117	**Recall Score**	0.86296	**F Score**	0.85460

The accuracy measures show promising results, however some aspects of our data are in need of a deeper analysis. There may be unnoticed signs of overfitting. Furthermore, the high number of false negatives from the confusion matrix are a sign that the class weights are unbalanced. This is in fact the case, with the majority class taking up almost 80% of the data.

In the future, we would like to implement a method to deal with this class imbalance. Methods like resampling, synthetic sampling, and bootstrapping do an excellent job at avoiding the downfalls of class imbalance.

References

1) "3.2.4.3.3. Sklearn.ensemble.ExtraTreesClassifier." Scikit, scikit-learn.org/stable/modules/generated/sklearn.ensemble.ExtraTreesClassifier.html.

2) Maklin, Cory. "K-Means Clustering Python Example." *Medium*, Towards Data Science, 21 July 2019, towardsdatascience.com/machine-learning-algorithms-part-9-k-means-example-in-python-f2ad05ed5203.