Speed Dating Classification

Applied Machine Learning

May 5, 2022

Abstract

Our project seeks to provide a detailed analysis of "match" patterns for individuals who are seeking potential romantic partners during speed dating events. We will use data retrieved from a set of experimental speed dating events from 2002-2004 where participants would have a 4-minute "first date" with every other participant of the opposite sex. At the end of the meeting, they were asked to fill out questionnaires where they responded to a series of questions about their dating preferences and whether they would like to go on a date again, which represented as 0 or 1 binary value decision. When both participants on the same date give 1 as their decisions, there is a match. Our key objective was to build an accurate and robust machine learning model to predict the target value "match" based on the attributes and features. After performing exploratory data analysis, data pre-processing, model tuning and selection, we present our final model: Random Forest on SMOTE-resampled data. This model gives a test accuracy of 0.971, precision of 0.938, recall of 0.888 and F1-score of 0.912.

1 Data Exploration

1.1 Exploratory Data Analysis

Summary of the Speed Dating data:

- Shape of the dataset: 8378×123
- Number of Classes: 2
- Number of Missing Values: 18372
- Number of Numeric Features: 59
- Number of Categorical Features: 62

Our key findings include the following: there were a total of 8378 dates that occurred. Demographically, females were more inclined to say "NO" than males. The average age of participants was 26.82. Females between the ages of 40 and 50 were generally not interested in dating. Most people belonged to the Business/Finance/Economics field of study. The least number of people belonged to the Architecture field. 50% of participants did not receive calls from their matches, 27% received only 1 call. 62% of participants did not get any dates from the experiments. Those who go out frequently got more matches.

1.2 Data Pre-processing

We have an unbalanced dataset. We have 1113 positive instances where it was a "match" and 5353 negative instances where it is "not a match". So positive instances were just 20.79% of the whole dataset. Hence we implement several resampling techniques to help improve model performance on the minority class, which is predictions on if there is a "match". The first method we use is stratified random sampling where we just split the data in such a way that the proportions of observations with a successful "match" and those without a "match" are maintained in the training set and the test set. Another method is random undersampling and random oversampling where we pick samples of the majority class at random without replacement and pick samples of the minority class at random with replacement, respectively. The third method is synthetic minority oversampling ("SMOTE") where samples of the minority class are synthesized by picking a sample at random and getting the k-nearest neighbors from the feature space. Random oversampling, random undersampling and SMOTE allows each set to have observations of equal class size.

We had 62 categorical features to be pre-processed for model building later. By reviewing the unique values of these columns, we decided to use one-hot encoding and ordinal encoding as the feature engineering techniques. Although we recognize the fact that some algorithms can handle categorical features as-is, both methods of encoding help to convert text values and allow us to train other statistical models that require numeric inputs to and draw predictions to make inferences. Specifically, we apply one-hot encoding to variables of "gender", "race", "race_o" and "field", because these variables contain only a limited set of values that do not have an inherent order. The rest of the features had an ordinal relationship, for example, many features were answers to questions like - "How interested are you in the following activities, on a scale of 1-10?" or "We want to know what you look for in the opposite sex out of attractiveness, sincerity, intelligence, fun, ambitious" For some features the scale was 1-10 and for some 1-100. The values were not integers but a range like:

- '[0-15]', '[16-20]', '[21-100]'
- '[0-1]', '[2-5]', '[6-10]'
- '[0-5]', '[6-8]', '[9-10]'

So we used Ordinal Encoding for such categories. During our EDA, we observed that almost 87% of the instances were missing in the dataset. Based on Figure 1, we removed the top 2 features, `expected_num_interested_in_me` and expected_num_matches, this resulted in achieving missing instances percentage dropping to 32.5%. Then upon removing the additional 2 features `shared_interests_o` and `shared_interests_partner`, it came down to 22.5% and we decided to go with this. We removed these missing instances from the dataset in order to avoid unnecessary bias. (see Figure 1)

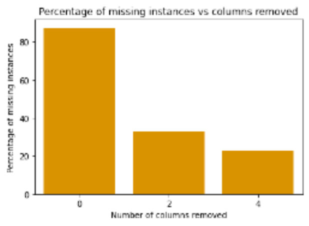


Figure 1: Missing data pattern

2 Model Building

2.1 Sampling Techniques Machine learning Models

We implemented four machine learning models in this project, namely Logistic Regression, Random Forest, Gradient Boosting and Support Vector Machine to compare the results and performances. For all of the models referred as default, we train them on the preprocessed data after stratified sampling during `train_test_split`. Along with these, we also implemented resampling techniques such as undersampling, oversampling and SMOTE to alleviate class imbalance issues.

We first implemented Logistic Regression and observed that the default model fitted on data achieved around 85% test accuracy. Then we applied the sampling techniques and observed that using the undersampling technique with Logistic regression did not improve test accuracy in the case, however we see that oversampling and SMOTE performed slightly better and generated a higher test accuracy but lower precision than the default model.

For the random forest model, we used cross validation and performed grid search in efforts to fix overfitting and to find the best hyperparameters such as `max_depth`, `max_features`, and `min_sanple_split` etc. Then we used these hyperparameters and implemented the model where we observed a 85% test accuracy. However, we saw that the model was still overfitting on the training set. We then applied the sampling techniques with Random Forest model (undersampling, oversampling and SMOTE, etc), and we observed that undersampling causes random forest to make much more positive predictions but raises false positives as well, and the overall precision score decreased as a result. However, oversampling and SMOTE with random forest seem to do well and generated test accuracy of 97%.

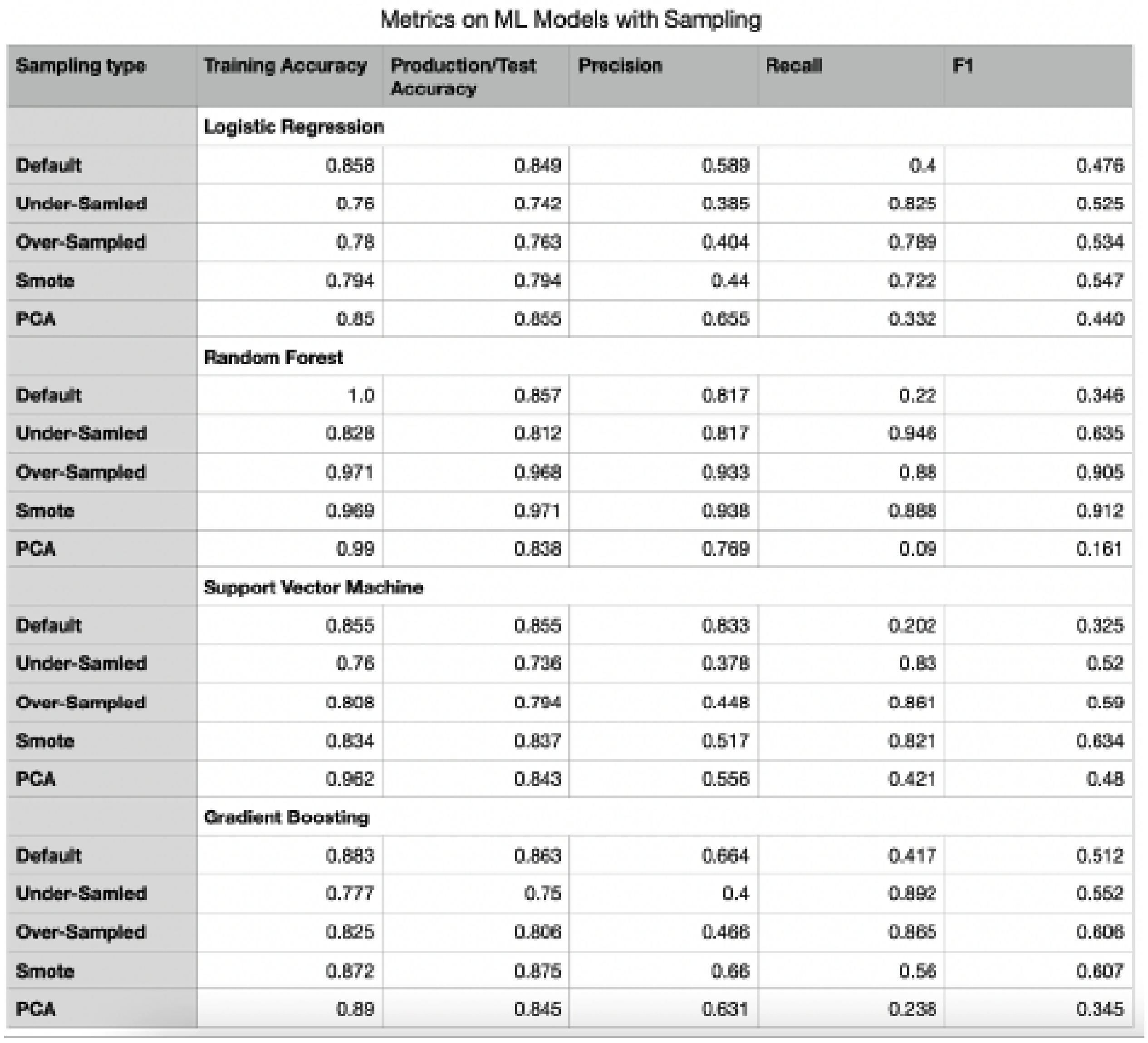
Support vector machine model is also implemented in this project because SVM tends to work well with classification tasks. We observed that the default SVM generated the same test accuracy as the default random forest model (85% test accuracy). Interestingly, we see that sampling techniques did not help improve the test accuracy in SVM.

Lastly, we implemented Gradient Boosting. Gradient Boosting supports sparse data and it provides lots of flexibility. We observed that default gradient boosting achieved slightly better test accuracy (86%) than default random forest and SVM models, and the SVM model does not seem to overfit. After applying sampling techniques on gradient boosting, we observed SMOTE technique slightly improved the accuracy of the model.

You will find plots of the roc curves, and precision-recall curves for each model in the **Appendix**.

2.2 Evaluation Metrics

In order to attest the performance of our models we used the following metrics: Training Accuracy, Production Accuracy, Precision Score, Recall Score and F1 Score. Indeed those metrics are relevant for our binary classification problem. Precision quantifies the number of correctly labeled positive class predictions divided by the number of all positively labeled class. Recall quantifies the number of correctly labeled positive class predictions made out of all actually positive examples in the dataset. F1 score provides a single score that balances both the concerns of precision and recall in one number. We have summarized all the above metrics in Figure 2:



Metrics on ML Models with Sampling

Sampling type	Training Accuracy	Production/Test Accuracy	Precision	Recall	F1
	Logistic Regression				
Default	0.858	0.849	0.589	0.4	0.476
Under-Samled	0.76	0.742	0.385	0.825	0.525
Over-Sampled	0.78	0.763	0.404	0.789	0.534
Smote	0.794	0.794	0.44	0.722	0.547
PCA	0.85	0.855	0.655	0.332	0.440
	Random Forest				
Default	1.0	0.857	0.817	0.22	0.346
Under-Samled	0.828	0.812	0.817	0.946	0.635
Over-Sampled	0.971	0.968	0.933	0.88	0.905
Smote	0.969	0.971	0.938	0.888	0.912
PCA	0.99	0.838	0.769	0.09	0.161
	Support Vector Machine				
Default	0.855	0.855	0.833	0.202	0.325
Under-Samled	0.76	0.736	0.378	0.83	0.52
Over-Sampled	0.808	0.794	0.448	0.861	0.59
Smote	0.834	0.837	0.517	0.821	0.634
PCA	0.962	0.843	0.556	0.421	0.48
	Gradient Boosting				
Default	0.883	0.863	0.664	0.417	0.512
Under-Samled	0.777	0.75	0.4	0.892	0.552
Over-Sampled	0.825	0.806	0.466	0.865	0.606
Smote	0.872	0.875	0.66	0.56	0.607
PCA	0.89	0.845	0.631	0.238	0.345

Figure 2: Evaluation Metrics

Figure 2: Evaluation Metrics

3 Model Interpretation & Feature Importance

One of the goals of our speed dating project is to understand what attributes or features contribute to a final match. Hence, we chose to implement interpretable models and plotted their feature importance (top 20 features). For example, the feature importance graph of the random forest model on SMOTE resampled data (See Figure 3) shows that `funny_o` is the most important feature in the model followed by `like`, `d_shared_interests_o` and `d_shared_partner`. Similarly, the feature importance graph of the gradient boosting model on SMOTE resampled data (See Figure 4) also shows the same three features as the most important top 4 features, but with `like` being the first most important feature followed by `d_shared_interests_o`, `d_shared_interests_partner` and `funny_o`. Overall, we observed consistency in the top 20 features across all models.

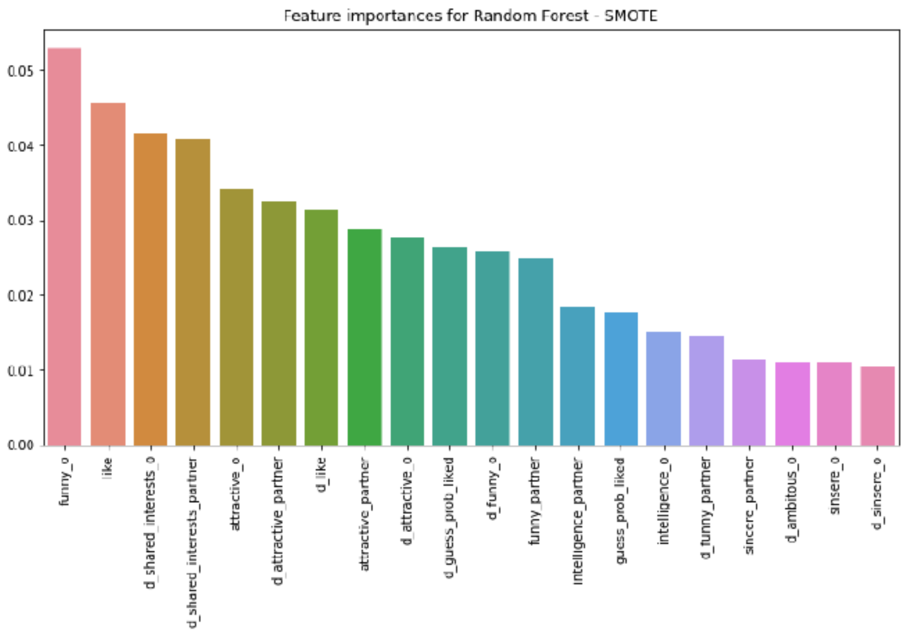


Figure 3: Feature Importances of Random Forest - SMOTE

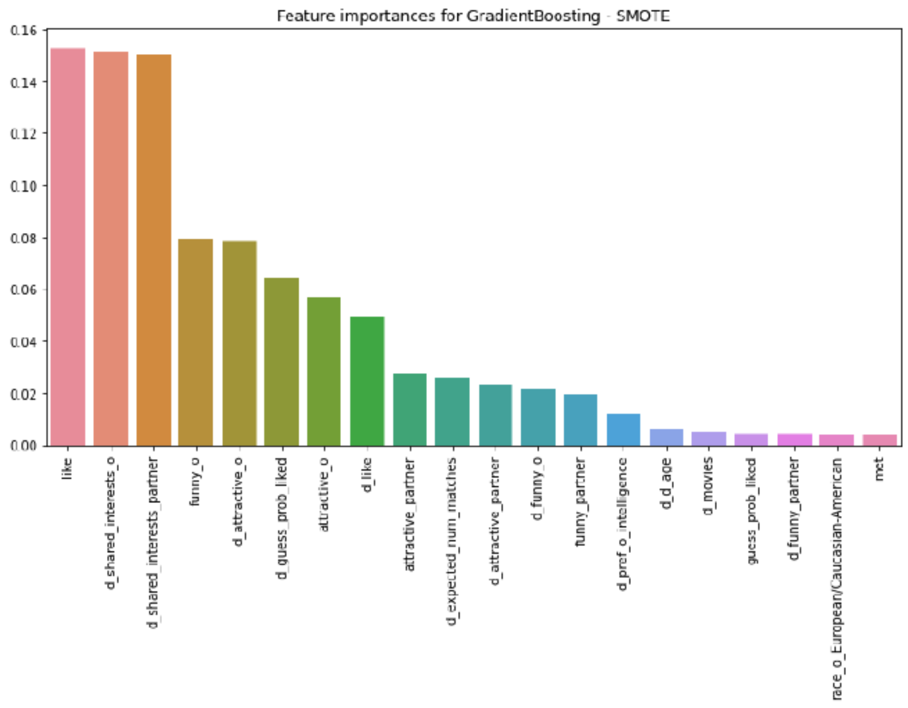


Figure 4: Feature Importances of Gradient Boosting - SMOTE

4 Conclusions

After some data exploration and pre-processing, our study aimed at building a binary classifier that would predict accurately the target value "match" according to the other available features of the dataset. After trial and error, we found that fitting random forests both on SMOTE and oversampled data perform significantly better than other models. And our final chosen model is Random Forest with SMOTE, which attains 0.97 accuracy, 0.94 precision, 0.89 Recall and 0.91

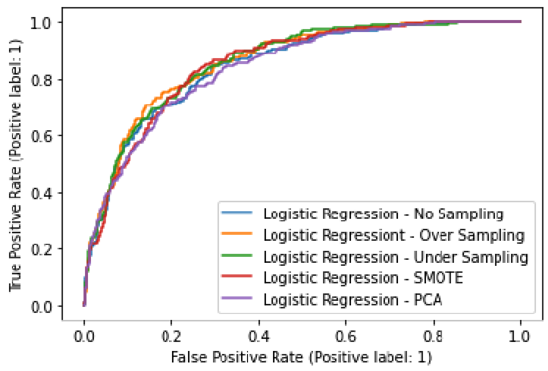
F1-score. If given more time, we could continue with Random Forest and use hyperparameter tuning to try to get better performance. Also, we find that models after applying Principal Components Analysis can give just as good performance in all of the evaluation metrics as the original data, but we successfully reduced the dimensionality to 50. And we could have experimented more innovative ways of dimensionality reduction and the possibility of further correcting model overfit is boundless. Our main struggle was to understand the meaning of some features, implementing effective ways of encoding the categorical variable in the dataset, handling missing data and selecting useful covariates for model building. We acknowledge that our study has some limitations, and the dataset may not be representative of current times because the survey was focused on US college students 20 years ago. Extrapolating dating patterns from this dataset may cause us to form false beliefs about the current state of romantic relationships.

5 References

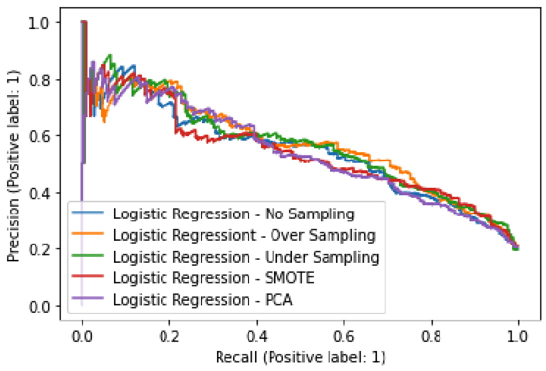
Vanschoren, J. (n.d.). Speeddating. OpenML. Retrieved February 21, 2022, from https://www.openml.org/d/40536

Iyengar, S., Kamenica, E., Simonson, I. (n.d.). Gender Differences in Mate Selection: Evidence from a Speed Dating Experiment. Retrieved February 21, 2022, from https://www8.gsb.columbia.edu/researcharchive/articles/867

Appendix

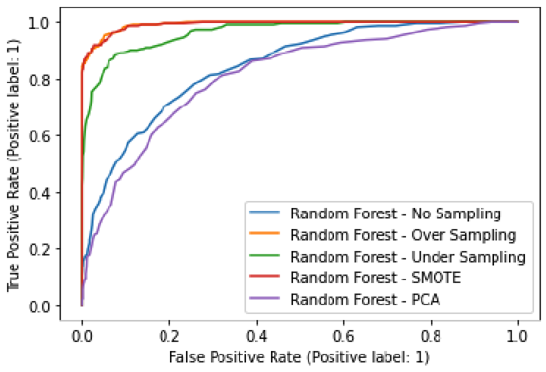


(a) Roc Curve

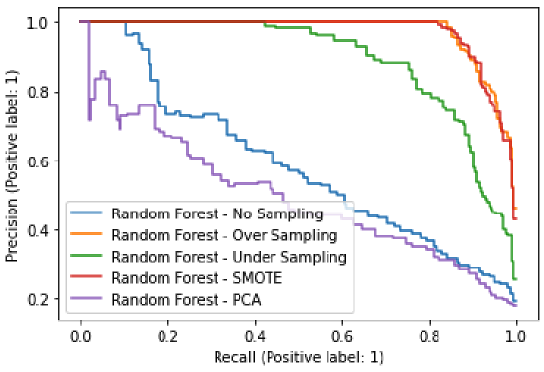


(b) Precision-recall Curve

Figure 5: Logistic Regression

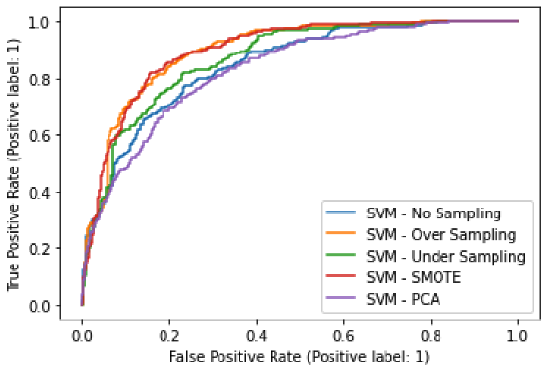


(a) Roc Curve

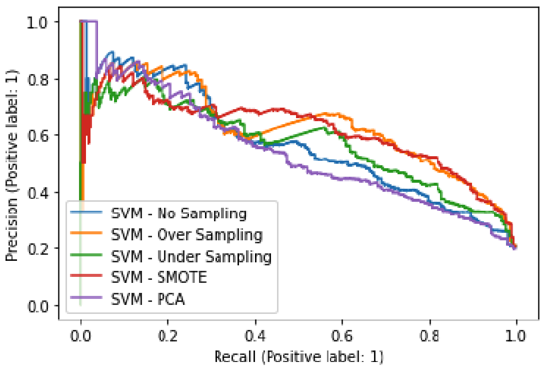


(b) Precision-recall Curve

Figure 6: Random Forest

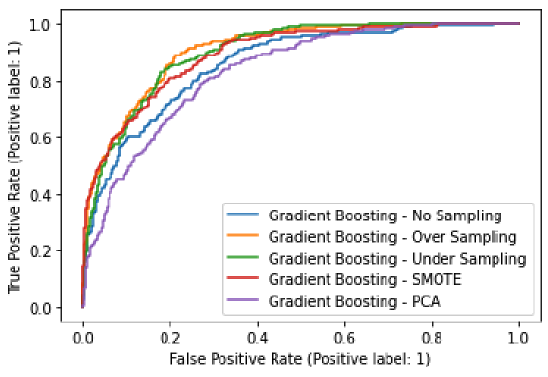


(a) Roc Curve

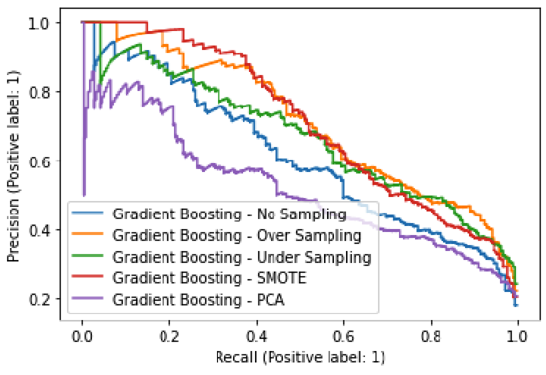


(b) Precision-recall Curve

Figure 7: Support Vector Machine



(a) Roc Curve



(b) Precision-recall Curve

Figure 8: Gradient Boosting Trees