EECE 5644 Project Milestone Report

Group 7: Robert Ashby, Helena Calatrava, Jay Im, Kuo Yang

Introduction Our group has chosen to analyze a dataset on heterosexual 'speed dating' to hopefully be able to effectively use machine learning techniques to match people together based on a profile of data a person provides. The original dataset contains 8378 rows and 195 columns. Each row represents one date, while each column represents one feature containing information about all the dates (e.g., expectation, description of the two people involved, outcome, etc.).

Data Exploration and PCA An initial selection of important columns (which represent features) has been performed by domain knowledge, discarding irrelevant features such as those whose data was taken post matching. After this, the number of remaining columns is 58 out of the initial 195 columns. Please refer to **Figure 1** to see the correlation between these selected features. We have considered the rows (i.e., dates) to be *i.i.d.*, although we acknowledge that this might not be true given the existence of different rounds in the dataset. We plan to explore various techniques for inter-row dependence identification (rank minimization, graphical LASSO, etc.). There are datasets that we cannot explain, and PCA can explain the correlations between the variables by lowering each dimension. Therefore, we used PCA to determine the correlation of continuous variables except categorical variables and found that most of the continuous variables have a correlation. PCA cannot explain which specific continuous variable has meaning, however, it is hopefully meaning that the continuous variables we use are correlated with each other and have their own necessity.

Data Cleaning Our dataset had a fair amount of null values upon initial inspection. We need to develop a more robust method of handling features with large amounts of null values, so we decided to ignore for now any features that had more than 1000 null values. Of the list of features that remain, we decided to ignore any rows that have null values as another temporary measure until we figure out a way to deal with these features. After handling the missing values in our dataset, 8024 remain out of the initial 8378 rows.

Stemming and Encoding We need to convert categorical data into numeric data so that the model can process it. After data cleaning, feature "from" has 266 unique categories. However, the content is not written in a uniform format. Therefore, a portion of categories are repeated. We use stemming to help provide uniformity. The basic idea is: extract the stem, convert all letters to lowercase, and then filter out the content that does not include geographic information. Using this method, we reduced the 266 types of data to 228 types. Next, we encode these categorical data using ordinal encoding. Because ordinal encoding gives the categorical data a numerical meaning, linear classification models like logistic regression will not work well. However, decision-tree based classification models will not be affected by it. So we are expecting a better performance when using the random forest.

Data Split As a first classification attempt, we have used k-Nearest Neighbors (KNN), Logistic Regression (LR) and Random Forest (RF) to classify the remaining 8024 samples/rows considering the 58 features that have been selected. The data is split into the training and testing sets, with proportions of 80% and 20%, respectively. We use stratified sampling to make sure that the data is evenly split according to label data.

Target Label With the first experiment, we consider the column *match* as the label of our classification problem (i.e. y_i = row_i['match']), which is equal to 1 in case both A_i and B_i liked each other and equal to 0 otherwise. Classification results obtained with this approach were 82.20%, 83.49% and 83.25% for KNN, LR and RF. Although these results seem good, we realized that the number of matches predicted by the LR classifier (the one giving best performance) was 0. Data inspection suggested that this was due to a low percentage of matches in the dataset (16%), which gives an accuracy of 84% if always predicting a negative match. We consider this experiment to be a failure, but a relevant milestone of our project, as with it we concluded **that our data was imbalanced**. In order to solve this, we have two solutions in mind. The first one, is the use of tools to deal with imbalanced data, such as the Python imbalanced-learn package, while the second one consists of the work developed in experiments 2 and 3. With the second experiment, we consider the features *dec_o* (1 if person B_i likes person A_i after the date, 0 otherwise) and *dec* (1 if $person_i$ likes person B_i after the date, 0 otherwise) as labels. With y_i = row_i['dec_o'], classification results obtained were 66.35%, 61.51% and 66.25% for the KNN, LR and RF, while with y_i = row_i['dec'], classification results obtained were 65.10%, 62.26% and 67.69%. The percentage of positive *dec_o* and *dec* values in our data is 42.15% and 42.22% (see **Figure 2**), approximately, which means that data is not as imbalanced when considering these features and this explains why obtained accuracies are lower, given that the posed classification problem is now more challenging. Results obtained in this experiment suggest that the use of these features in the classification label brings diversity to the classification outcome, thus making the classification problem more complete. With the last experiment, we use a categorical label, with 0 (*dec_o* = 0, *dec = 0*), 1 (*dec_o* = 0, *dec = 1*), 2 (*dec_o* = 1, *dec = 0*) and 3 (*dec_o* = 1, *dec = 1*) as labels. The problem posed with this categorical label is more challenging because the data with negative match from experiment 1 has been separated into three categories (labels 0, 1 and 2 for this experiment). Obtained classification accuracies are lower, being 44.54%, 38.56% and 45.98% for the KNN, LR and RF (a confusion matrix of our results was created see **Figure 3**). We consider this experiment to be a success when it comes to the design of our classification label. It also poses a new question regarding how to change the parameters of the classifiers so that results are improved, given that the classification problem is more difficult after re-designing the label. This brings us to the next chapter on hyperparameter optimization.

Hyperparameter Optimization and Model Selection In order to find the best values for the hyperparameters of the KNN, LR and RF classifiers, and as part of the model selection stage, we have used Python's GridSearchCV to exhaustively search over specified parameter values using k-fold cross-validation. So far we have tried with k = 5. Results after hyperparameter optimization are 45.61%, 38.19% and 46.85% for the KNN, LR and RF classifiers. In all the experiments, better results are obtained with RF. However, all accuracy percentages are low (under 50%).

Feature Importance Analysis With L1-regularization we are able to check whether correlations are positive or negative, while with RF feature importance we can measure the magnitude of these correlations. So far we see that the most important features are *int_corr* (i.e., correlation between interests of the participants), *order* (i.e., how the dates are distributed in time) and *age_o* (i.e., age of partner).

Proposed Final Report Scope of Work Our results were not as accurate as we would have hoped, so we propose to do two things moving forward. The first is to improve the accuracy of our classifiers by further data processing (under/over sampling, etc.), exploring other models (SVM, unsupervised methods, etc.), and further tuning the models we have already created. We are also willing to split the dataset into three parts (i.e., training, testing and validation) in order to make sure our models are able to generalize, avoiding overfitting. We also propose to look into recommendation engines and matching algorithms to provide an ideal profile that a given person would match with, and use it to find possible matches for that person.

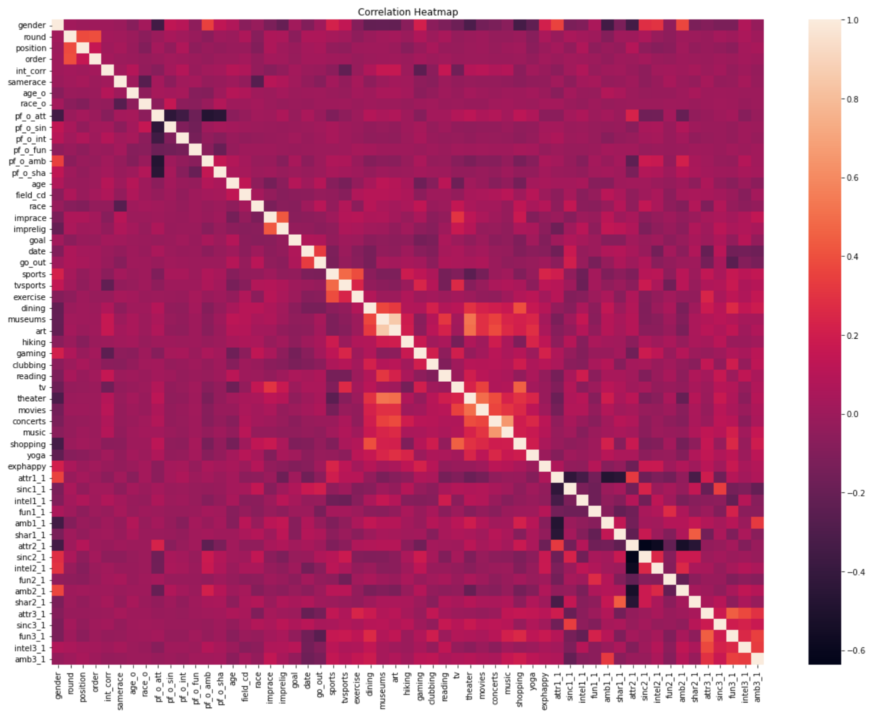

Figure 1. Correlation matrix between the 58 columns intuitively selected in the data exploration stage. High correlation values (see square in the middle) denote that there is a correlation between hobbies of the participants.

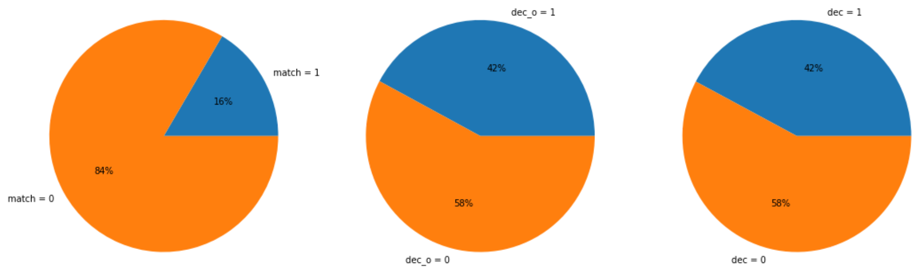

Figure 2. Percentage of features *match*, *dec_o*, *dec*, used for data labeling.

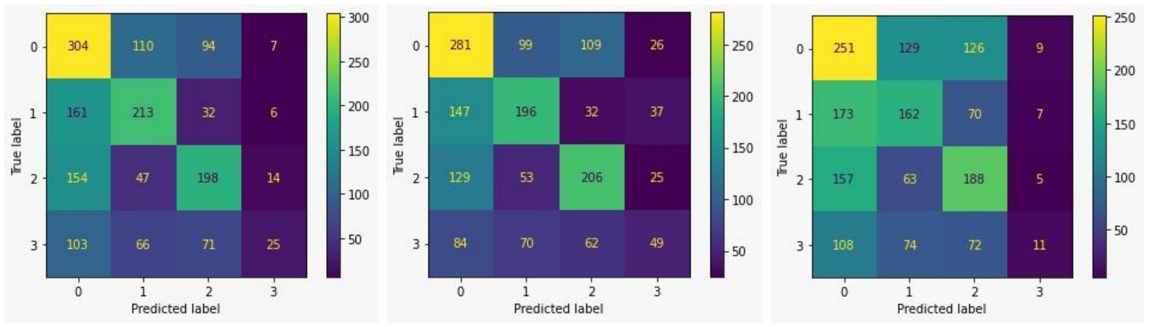

Figure 3. Confusion matrices of Random Forest (left), Logistic Regression (middle), KNN (right).