Machine Learning Report - Group 21

Feature Engineering

First, we performed Exploratory Data Analysis on the training data. We discovered that CurrentGameMode and CurrentTask both have 12.65% missing values, which is nearly the same proportion (12.66%) as the missing values in LevelProgressionAmount. By creating a heatmap of the features with missing values, we observed a consistent pattern: these three features were missing simultaneously. Additionally, the feature LastTaskCompleted has a higher missing value percentage of 46.57%, and the rows where these three features are missing also lack values in LastTaskCompleted. This indicates that these four features are missing not at random. Logically, we inferred that the simultaneous missing values suggest that the user hasn't started to play the game, and therefore, hasn't chosen a mode or task. Consequently, we replaced the missing values in the CurrentGameMode, CurrentTask, and LastTaskCompleted columns with 'None', and in the LevelProgressionAmount column with 0. Next, we addressed outliers. We observed that negative values in the CurrentSessionLength feature are abnormal, as all should be positive integers starting from 0. Since no similar outliers were present in the test data, we handled this issue solely in the training data. Additionally, we didn’t find any duplicates in the dataset.

Second, we performed feature engineering, including feature transformation, extraction, and selection. We converted QuestionTiming into boolean as it only has two unique values. We extracted five features: hour, day_of week, month, week_of_year, and is_weekend from the TimeUtc column. However, via the correlation matrix, we noticed that feature week_of_year has a strong relation with month feature, so we decided to drop it. After that, we standardized the numerical features by removing the mean and scaling to unit variance to ensure that numerical features are on a comparable scale which helps our gradient descent based model converge more quickly and efficiently. We compared one-hot encoding and label encoding and ultimately decided to one-hot encode all the categorical features. This way, each category is treated independently without implying any order, preserving the categorical nature of the data. we combine these preprocessing steps into pipelines to better process test data before predicting target values.

Model selection

Since our target value, well-being scores, are continuous numbers, we decided to focus on regression models. We tried Linear Regression, Stochastic Gradient Descent Regression, Extreme Gradient Boosting Regression, and Logistics Regression. During the data preprocessing stage, we used Linear Regression as our baseline model to test the effect of feature engineering. It didn’t perform well on the validation dataset until we standardized these numerical features. The processes of data preprocessing and training models are not independent of each other, as we found out that the performances of different models varied when we adopted and combined various methods of preprocessing. Ultimately, Linear Regression became our best model, achieving the lowest MAE of 96.48 on our validation data.

Hyperparameter Tuning and Performance Discussion

However, when we uploaded the predicted dataset to the Codalab, the MAE increased to 111.4422. It indicates a potential issue with overfitting. We then realized that we should use K-fold cross-validation instead of the simple train-test split method, which helps to get robust predictions. Finally, to search for a set of hyperparameters that result in the best performance for our two best-performing models, Linear Regression and SGD regression, we used grid search, which automatically tried all possible combinations of hyperparameters to find the best set. For Linear Regression, the best parameter was 'model__fit_intercept': False, and for SGD regression, the best parameters were 'model__alpha': 0.0001, 'model__eta0': 0.1, 'model__learning_rate': 'adaptive', 'model__max_iter': 1000.

Without time limitations, we would explore more sophisticated imputation methods for missing values. Additionally, we believe that Lasso and Ridge Regression are effective techniques for addressing overfitting in linear regression models. However, due to the approaching deadline, we do not have the time to delve into these assumptions.