Supplemental Material

Machine learning Pipelines Configuration

We evaluated the following Machine Learning Pipelines based on different ML classifiers. Extensive finetuning of the various parameters and experimentation was conducted to find the best steps of these pipelines as they appear below.

LightGBM

1. **Feature Scaling**: No scaling is applied to numerical features, since for the LightGBM classifier scaling has no effect.
2. **Handling Missing Values**: We encoded the missingness using the built-in LightGBM function.
3. **One Hot Encode**: We didn't perform one hot encoding of the categorical features since it has no effect for the LightGBM classifier.
4. **LightGBM Classifier**: We used the LightGBM classifier with the following hyperparameters: is_unbalance=True, subsample=0.33, subsample_freq=7, extra_trees=True, colsample_bytree=0.95, feature_fraction_bynode=1.0, num_leaves=480, reg_alpha=2.3e-6, reg_lambda=2.5e-7, learning_rate=5e-4, linear_lambda=1e-5, max_depth=33, min_child_samples=1, n_estimators=500, objective='cross_entropy'

Random Forest

1. **Feature Scaling**: No scaling is applied to numerical features, since for the Random Forest classifier scaling has no effect.
2. **Handling Missing Values**: We encoded the missingness by filling the missing values with the value -99.
3. **One Hot Encode**: We didn't perform one hot encoding of the categorical features since it has no effect for the Random Forest classifier.
4. **Random Forest Classifier**: We used the Random Forest classifier of the scikit-learn library with the following hyperparameters: class_weight='balanced', max_depth=645, max_leaf_nodes=830, min_impurity_decrease=2.6e-4, min_weight_fraction_leaf=1e-3, n_estimators=260, max_samples=0.73

Logistic Regression

1. **Feature Scaling**: Min-Max scaling is applied to numerical features, using the implementation of the scikit-learn library.
2. **Handling Missing Values**: We imputed the missing values using the miceforest implementation of the MICE imputer. The optimal parameters for the LightGBM regressor used by the imputer were found by running 50 iterations of the internal procedure of miceforest. We used mean matching with 10 matching candidates. We ran the MICE algorithm for 5 iterations.
3. **One Hot Encode**: We performed one hot encoding of the categorical features, using the implementation of the scikit-learn library.

4. **Logistic Regression**: We used the Logistic Regression classifier of the scikit-learn library with the following hyperparameters: max_iter=10000, solver='saga', class_weight='balanced', penalty='elasticnet', C=0.003, l1_ratio=0.04

SVM

1. **Feature Scaling**: Min-Max scaling is applied to numerical features, using the implementation of the scikit-learn library.
2. **Handling Missing Values**: We imputed the missing values using the miceforest implementation of the MICE imputer. The optimal parameters for the LightGBM regressor used by the imputer were found by running 50 iterations of the internal procedure of miceforest. We used mean matching with 10 matching candidates. We ran the MICE algorithm for 5 iterations.
3. **One Hot Encode**: We performed one hot encoding of the categorical features, using the implementation of the scikit-learn library.
4. **SVM**: We used the SVC classifier of the scikit-learn library with the following hyperparameters: class_weight='balanced', C=0.15, kernel='rbf', gamma=0.29, probability=True

Decision Tree

1. **Feature Scaling**: No scaling is applied to numerical features, since for the Decision Tree classifier scaling has no effect.
2. **Handling Missing Values**: We encoded the missingness by filling the missing values with the value -99.
3. **One Hot Encode**: We didn't perform one hot encoding of the categorical features since it has no effect for the Decision Tree classifier.
4. **Decision Tree Classifier**: We used the Decision Tree classifier of the scikit-learn library with the following hyperparameters: class_weight='balanced', max_depth=375, max_leaf_nodes=1240, min_impurity_decrease=2e-3, min_weight_fraction_leaf=0.0478

Fuzzy Decision Tree

1. **Feature Scaling**: No scaling is applied to numerical features, since for the Fuzzy Decision Tree classifier scaling has no effect.
2. **Handling Missing Values**: We encoded the missingness by filling the missing values with the value -99.
3. **One Hot Encode**: We didn't perform one hot encoding of the categorical features since it has no effect for the Fuzzy Decision Tree classifier.
4. **Fuzzy Decision Tree Classifier**: We used the Decision Tree classifier of the fuzzytree library with the following hyperparameters: fuzziness=0.97, max_depth=550, min_impurity_decrease=3e-8

Neural Network

1. **Feature Scaling**: Min-Max scaling is applied to numerical features, using the implementation of the scikit-learn library.

2. **Handling Missing Values**: We imputed the missing values using the miceforest implementation of the MICE imputer. The optimal parameters for the LightGBM regressor used by the imputer were found by running 50 iterations of the internal procedure of miceforest. We used mean matching with 10 matching candidates. We ran the MICE algorithm for 5 iterations.
3. **One Hot Encode**: We performed one hot encoding of the categorical features, using the implementation of the scikit-learn library.
4. **Neural Network**: We used the MLP Classifier of the scikit-learn library with the following hyperparameters: hidden_layer_sizes=(25, 10), activation='relu', max_iter=145, early_stopping=False