ML Final Report

GitHub link: https://github.com/kesw/Tabular-Platground-Series-AUG_2022

Reference links:

(1) https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html

(2) https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.interpolate.html

Brief Introduction:

Using Multi-layer Perceptron classifier from sklearn as the model, containing the predict_proba() function to obtain the failure probability.

Methodology:

Data pre-process:

The data obtained had flaws for me to move to the training session, so some changes were made:

Missing of data in the 'loading' and 'measurements' column:

(1) use pandas Dataframe's "interpolate()" function to fill up the missing data, "limit_direction='both'" to also fill up the top and bottom sides.

Change type for the 'attribute_0' and 'attribute_1' columns from 'object' to 'int':

(1) First, only keep the 'numeric' part of the data and change it to int by "pandas.to_numeric".

(2) For example: "measurement_3" to "3".

Change type for the 'product_code' column:

(1) Use LabelEncoder() from sklearn.preprocessing to encode string to integer.

Balance the data:

(1) The data is quite unbalance so SMOTE is used to oversample the data.

Model architecture:

Multi-layer Perceptron classifier(MLPClassifier) from sklearn.

The reason for using a classifier is that the 'failure' column in the train_data has only 2 labels (0, 1), so a classifier would work well, and since sklearn provide the "predict_proba" function, probability is easily achieve.

Hyperparameter:

(1) Hidden_layer_surface: (100, 100, 100) # 3 layers with 100 neurons each.

(2) Batch_size = 64

(3) Learning_rate_init= 0.001

(4) Early_stopping=True

(5) Solver="sgd"

(6) Learning_rate="adaptive"

(7) Random_state=42

Notes: saving the model with "joblib"

109550193_submission.csv

Complete (after deadline) · now

0.58941   0.58545