**Continuous Factory Process Data Prediction**

**Objective**

This project focuses on developing a predictive machine learning model to analyse and forecast continuous factory process data. The model aims to predict a process parameters based on historical sensor readings, ensuring optimized manufacturing performance and reducing operational inefficiencies.

Predicted Parameter:

* Stage1.Output.Measurement14.U.Actual

Predictor Parameters

* AmbientConditions.AmbientHumidity.U.Actual
* AmbientConditions.AmbientTemperature.U.Actual
* Machine1.RawMaterial.Property1', 'Machine1.RawMaterial.Property2
* Machine1.RawMaterial.Property3', 'Machine1.RawMaterial.Property4
* Machine1.RawMaterialFeederParameter.U.Actual
* Machine1.Zone1Temperature.C.Actual
* Machine1.Zone2Temperature.C.Actual', 'Machine1.MotorAmperage.U.Actual
* Machine1.MotorRPM.C.Actual', 'Machine1.MaterialPressure.U.Actual
* Machine1.MaterialTemperature.U.Actual
* Machine1.ExitZoneTemperature.C.Actual
* Machine2.RawMaterial.Property1', 'Machine2.RawMaterial.Property2
* Machine2.RawMaterial.Property3', 'Machine2.RawMaterial.Property4
* Machine2.RawMaterialFeederParameter.U.Actual
* Machine2.Zone1Temperature.C.Actual
* Machine2.Zone2Temperature.C.Actual', 'Machine2.MotorAmperage.U.Actual
* Machine2.MotorRPM.C.Actual', 'Machine2.MaterialPressure.U.Actual
* Machine2.MaterialTemperature.U.Actual
* Machine2.ExitZoneTemperature.C.Actual
* Machine3.RawMaterial.Property1', 'Machine3.RawMaterial.Property2
* Machine3.RawMaterial.Property3', 'Machine3.RawMaterial.Property4
* Machine3.RawMaterialFeederParameter.U.Actual
* Machine3.Zone1Temperature.C.Actual
* Machine3.Zone2Temperature.C.Actual', 'Machine3.MotorAmperage.U.Actual
* Machine3.MotorRPM.C.Actual', 'Machine3.MaterialPressure.U.Actual
* Machine3.MaterialTemperature.U.Actual
* Machine3.ExitZoneTemperature.C.Actual
* FirstStage.CombinerOperation.Temperature1.U.Actual
* FirstStage.CombinerOperation.Temperature2.U.Actual
* FirstStage.CombinerOperation.Temperature3.C.Actual']

**Key Goals:**

* Preprocess raw industrial data from a continuous factory process.
* Engineer relevant features to enhance model performance.
* Train and optimize machine learning models, primarily using Random-Forest Regressors.
* Evaluate model performance using Mean Absolute Error method.
* Deploy the trained model for real-time inference.

**Project Structure**

The project is structured into the following folders:

**Data**

Contains the datasets used for model training, testing, and evaluation:

* continuous\_factory\_process.csv - Raw dataset containing sensor readings from industrial machines.
* hyperparameter\_results.csv - Results of hyperparameter tuning experiments.
* notes\_on\_raw\_dataset.txt - Document detailing the dataset structure and processing steps.
* Processed\_Data.csv - Cleaned and pre-processed dataset ready for training.
* X\_train\_dataset.csv - Training features dataset.
* X\_test\_dataset.csv - Test features dataset.
* y\_train\_dataset.csv - Training target values.
* y\_test\_dataset.csv - Test target values.

**Models**

Stores machine learning models and scaling objects:

* Final\_Model.pkl - The trained final model used for predictions.
* RandomForest\_best.pkl - Optimized Random Forest model.
* X\_minmax\_scaler.pkl - MinMaxScaler object used to scale input features.
* y\_minmax\_scaler.pkl - MinMaxScaler object used to scale target values.

**Source Code**

Contains Jupyter notebooks covering the entire pipeline:

* load\_data.ipynb - Loads raw industrial data and performs initial exploration.
* pre\_data\_processing.ipynb - Handles data cleaning and missing value imputation.
* feature\_engineering.ipynb - Extracts and creates new features for improved model performance.
* model\_training.ipynb - Trains various machine learning models, including Random Forest.
* model\_tuning.ipynb - Fine-tunes hyperparameters to optimize model accuracy.
* model\_evaluation.ipynb - Evaluates model performance using metric MAE.
* model\_inference.ipynb - Deploys the final model for making predictions on unseen data.

**Setup Instructions**

1. Ensure that you have Python installed, then install dependencies using the provided requirements.txt:

pip install -r requirements.txt

1. Activate virtual environment, the project uses a virtual environment called myenv. To activate it run the following:

python3 -m venv myenv

source myenv/bin/activate # for macOS/Linux

myenv/Scripts/activate # for Windows

1. Run jupyter notebook:

jupyter notebook

**How to Use the Model**

1. Load the model, scalers, and selected features:

import joblib

best\_model = joblib.load("Models/Final\_Model.pkl")

scaler\_y = joblib.load("Models/y\_minmax\_scaler.pkl")

scaler\_X = joblib.load("Models/X\_minmax\_scaler.pkl")

with open("../Data/selected\_features.txt", "r") as f:

    selected\_features = [line.strip() for line in f]

1. Load your new data and perform feature engineering

import pandas as pd

import numpy as np

from sklearn.preprocessing import MinMaxScaler

# Load a new data sample

X\_new = pd.read\_csv("*YOUR NEW X DATA*.csv").sample(n=1)

# Perform feature engineering on X data

# Ambient temperature feature interaction

X\_new['Ambient\_Temp\_Humidity\_Interaction'] = X\_new['AmbientConditions.AmbientHumidity.U.Actual'] \* X\_new['AmbientConditions.AmbientTemperature.U.Actual']

# Motor/RPM feature interaction

X\_new['Machine1\_RPM\_Amperage\_Interaction'] = X\_new['Machine1.MotorAmperage.U.Actual'] \* X\_new['Machine1.MotorRPM.C.Actual']

X\_new['Machine2\_RPM\_Amperage\_Interaction'] = X\_new['Machine2.MotorAmperage.U.Actual'] \* X\_new['Machine2.MotorRPM.C.Actual']

X\_new['Machine3\_RPM\_Amperage\_Interaction'] = X\_new['Machine3.MotorAmperage.U.Actual'] \* X\_new['Machine3.MotorRPM.C.Actual']

# Material temperature and pressure feature interaction

X\_new['Machine1\_Temp\_Press\_Interaction'] = X\_new['Machine1.MaterialPressure.U.Actual'] \* X\_new['Machine1.MaterialTemperature.U.Actual']

X\_new['Machine2\_Temp\_Press\_Interaction'] = X\_new['Machine2.MaterialPressure.U.Actual'] \* X\_new['Machine2.MaterialTemperature.U.Actual']

X\_new['Machine3\_Temp\_Press\_Interaction'] = X\_new['Machine3.MaterialPressure.U.Actual'] \* X\_new['Machine3.MaterialTemperature.U.Actual']

# drop columns not needed to run model

X\_new = X\_new[selected\_features]

# apply MinMaxScaler to X data

X\_new\_scaled = scaler\_X.transform(X\_new)

1. Run the model and predict ‘*Stage1.Output.Measurement14.U.Actual’* values

y\_pred\_scaled = best\_model.predict(X\_new\_scaled)

# Convert prediction back to original scale

y\_pred = scaler\_y.inverse\_transform(y\_pred\_scaled.reshape(-1, 1))

print("Predicted Output:", y\_pred[0][0])

**Model Performance & Evaluation**

The model was evaluated using Mean Absolute Error (MAE) and achieved an average MAE of approximately 12.4%

Further evaluations can be conducted using model\_evaluation.ipynb, where different model performances are compared.

**Contributors**

* Project Owner: Mo Somji
* Developers: Mo Somji
* Date: 02/03/2025

If you have suggestions or improvements, feel free to contribute!