Meta Smart Factory – Trial Task Brief

Subject: Feature Engineering, Predictive Modeling, and Optimization on Industrial Data

Dear [Candidate Name],

Thank you once again for applying to Meta Smart Factory via LinkedIn.

We would like to invite you to complete a short technical task using a real-time, labeled dataset collected from an actual industrial process. The dataset will be provided in Excel format. Your objective is to extract actionable insights and simulate how input parameters influence output performance—similar to the real-world process optimization projects we deliver to manufacturers.

Please note: Python is mandatory for this task. Excel is provided solely as the data source.

Task Scope

Deliverables: A concise presentation (maximum 5 slides per section) and an accompanying Excel summary file.

---

1. Input–Output Correlation

Objective: Identify dependencies and relationships among inputs and outputs to guide future modeling decisions.

Recommended Techniques:

1.1 Spearman Rank Correlation

- Measures monotonic relationships between variables.
- Applicable even when the relationship is not strictly linear.

Use Cases:

- Input–Output: Analyze how sensor readings (e.g., temperature) influence outcomes (e.g., product yield).
- Input–Input: Detect multicollinearity among input features.

- Output–Output: Reveal dependencies among performance metrics (e.g., yield vs. energy usage).

1.2 Mutual Information (MI)

- Captures the shared information between two variables.
- Effective in identifying non-linear dependencies.

Use Cases:

- Input–Output: Select the most informative variables for model building.
- Input–Input: Detect redundant or overlapping sensor signals.
- Output–Output: Prioritize dependent KPIs.

Why This Matters:
Understanding these correlations enables dimensionality reduction, avoids redundant inputs, prevents overfitting, and improves model clarity and efficiency.

---

2. Feature Importance

Objective: Determine which input variables most significantly impact the output variable.

Recommended Approaches:

2.1 XGBoost Feature Importance

- A tree-based machine learning model that ranks features based on:
    - **Gain**: Improvement in accuracy from a feature
    - **Cover**: Number of samples affected
    - **Frequency**: How often the feature is used

2.2 SHAP (SHapley Additive Explanations)

- Model-agnostic method based on cooperative game theory.

- Quantifies the contribution of each input to individual predictions.
- Supports both global and local interpretability.

Use Case Example:
Determine whether temperature, pressure, humidity, or dosing time has the greatest impact on product quality.

---

3. Output Prediction & Parameter Optimization

Objective: Predict output performance (e.g., quality, yield, energy) based on process inputs, and determine optimal input values to improve those outputs.

Recommended Algorithms:

3.1 SVR (Support Vector Regression – RBF Kernel)

- Effective for smooth, non-linear relationships between inputs and outputs.
- Suitable for small to medium datasets.

3.2 Random Forest Regressor

- A non-linear ensemble method that captures complex relationships and feature interactions.

3.3 DVM (Dynamic Vector Machines)

- Suitable for time-varying systems with process drift.
- Learns and adapts as new data is introduced.

Industrial Scenario:
Predict oil yield using temperature, pressure, moisture, dosing speed, and concentration. Then, identify optimal input values to maximize the predicted yield.

---

4. (Optional) Anomaly Detection

Objective: Identify abnormal patterns or behaviors in sensor data that may indicate system faults or process instability.

Recommended Approaches:

4.1 LSTM Autoencoder

- Designed for time-series anomaly detection.
- Learns to reconstruct “normal” sensor behavior and identifies sequences with high reconstruction error as anomalies.

Best For:

- Streaming sensor data (e.g., temperature, vibration)
- Early fault detection or process deviation tracking

Use Case:
Detect abnormal fluctuations in temperature or vibration patterns that could signal mechanical wear or instability.

4.2 Isolation Forest

- Unsupervised model that isolates outliers using random trees.
- Flags data points that are easy to isolate (i.e., anomalies).

Best For:

- Static, high-dimensional snapshots of process data
- Outlier detection across multiple variables

Use Case:
Flag batches with unusually high energy consumption or sensor values that fall outside expected operating ranges.

Why Anomaly Detection Matters in Manufacturing:

Application	Value Gained
Fault Detection	Early identification of failures or degradations
Process Safety	Alert on unsafe or unstable operating conditions
Quality Assurance	Prevent defective outputs by monitoring anomalies

---

Optional Output Submission (for Anomaly Detection):

If this section is implemented, please include:

- Visualizations of detected anomalies
- Explanation of thresholds or detection logic
- Comparison of normal vs. abnormal samples
- Summary of detection accuracy or confidence level (if applicable)

Tools & Submission Guidelines

- You may use any Python library (e.g., `scikit-learn`, `pandas`, `xgboost`, `shap`, `keras`) to perform the task.
- Deliverables include:
    - A presentation (maximum 5 slides per section)
    - An Excel summary file with results, visualizations, and notes
- Task Duration: **1 week**
- Supporting walkthrough video: Loom Link

If you are ready to proceed, please confirm. Upon your confirmation, we will send you the Excel dataset and further instructions.

We look forward to hearing from you and wish you the best of luck.

Kind regards,
Meta Smart Factory – Technical Team