Data Scientist Hiring – Code Challenge

Purpose:

The goal of this code challenge is to assess the capabilities of a potential new hire in solving a data science problem. This includes evaluating their coding skills, problem-solving abilities, model development proficiency, and understanding of MLOps principles. By completing this challenge, you will demonstrate your ability to handle real-world data science problems, develop effective models, and communicate your findings clearly.

Problem Statement:

Your task is to analyze the data provided in the CSV files and train (using training data) a classification model of your choice. The final trained model should be capable of accurately classifying unknown data as either "perfect" or "imperfect" using the data the evaluation data. It is crucial to ensure that the model does not overfit or underfit the provided data. Please share all relevant model artifacts, including but not limited to recall and precision values. Additionally, highlight any data challenges encountered during the process.

Data Description:

The attached dataset originates from a time series molecular biology experiment. The first column represents the time points, while the remaining columns contain the observations from the experiment. The labels for the experimental data are provided in the first row. The data is categorized into two distinct groups, with one category labeled as "perfect" and the other is “imperfect” indicating if the experiment was conducted under optimal conditions or not.

Additional Note:

- There are two files (training data :“training_data_external” and evaluation data: “evaluation_data_external”).
- The data is small and imbalanced.
- The data might be converted to string values during transfer.

Detailed Instructions:

1. Data Understanding: Begin by thoroughly exploring the dataset to understand its structure, distribution, and any inherent challenges. Pay attention to missing values, outliers, and any patterns that may influence model performance.

2. Model Selection: Choose an appropriate classification model. Justify your choice based on the nature of the data and the problem requirements. Consider models that are robust to overfitting and underfitting.
3. Model Training: Train the selected model using the provided data. Implement techniques to prevent overfitting, such as cross-validation, regularization, and hyperparameter tuning.
4. Model Evaluation: Evaluate the model's performance using relevant metrics, including recall, precision, F1-score, and accuracy. Ensure that the model generalizes well to unseen data.
5. Artifact Sharing: Provide all model artifacts, including code, trained model files, evaluation metrics, and any visualizations that support your findings.
6. Data Challenges: Document any challenges encountered during the data analysis and model training process. Discuss how these challenges were addressed and their impact on the model's performance.

Evaluation Criteria:

The detailed evaluation criteria are provided below.

Capability	Score	Evaluation Criteria
Problem Understanding	15	• Clear articulation of data relationships. • Justification for selected methods and features.
Data Analysis and Preprocessing	20	• Rigorous exploratory data analysis with meaningful insights. • Appropriate handling of missing values and outliers. • Logical data transformations and normalization techniques.
Modeling and Algorithm Selection	25	• Sound reasoning behind model selection (logistic regression, random forest, XGBoost, etc.). • Effective feature selection methodology and validation strategy (cross-validation, train-test splits).
Model Evaluation and Interpretation	20	• Appropriate use of evaluation metrics. • Clear interpretation and justification of results. • Demonstration of understanding false positives/negatives implications in clinical settings.
Coding and Technical Skills	10	• Clean, readable, well-documented code. • Efficient coding practices and proper use of libraries (pandas, scikit-learn, PyTorch/TensorFlow).

MLOps and Deployment Understanding	10	• Demonstration of reproducibility (e.g., through GitHub, Docker, environment files). • Knowledge of deployment concepts (model persistence, versioning, scalability considerations). • Model maintenance

Coding Language: Python/C/C++

Expected Deliverables: Zip file containing the following

- Solution: Explanation of the solution clearly and concisely.
- README file to explain on how to execute the code
- Code: Well-documented code for data preprocessing, model training, and evaluation. The code should be executable (jupyter notebook)
- Model Files: Trained model files and any necessary scripts to reproduce the results. Evaluation Metrics: Detailed metrics that demonstrate the model's performance.
- Documentation: A comprehensive report highlighting the data challenges, model selection rationale, and steps taken to ensure robust model performance.
- Any addition input files to run the software and the output files(s)

Suggestions/Guidelines:

- Do not share any proprietary code or knowledge belongs to other organizations.

Deadline: 48 Hrs.