The MLOPS pipeline I have decided to describe in this document is based upon the DSBA platform repository. This pipeline is designed for use with classification models, hence the metrics used are ones like f1 scores. The features are as described below. Not all features are implemented in depth. This is merely a description of the full pipeline as well as my own additions to it.

**Interfaces**

**Command Line Interface with following features:**

Allows viewing all models in the registry. Also allows for making predictions using a specified model with specified dataset

**Web API**

List models and make predictions via HTTP requests. Processes and returns structured JSON data.

**Notebook Interface:**

Primary development environment. Jupyter notebook for model training, experimentation, visualizations and selection of a final model.

**Dockerization:**

The API service can be packaged in a Docker container which allows for configuring model paths and other settings via environment variables.

**Environment Management**

Allows configuring model storage locations via DSBA\_MODELS\_ROOT\_PATH

Platform and user independent path handling

Test individual components in isolation using unit tests.

**Data Ingestion and Exporting:**

Loading CSV files from local paths or URLs. Query data from PostgreSQL databases with configurable parameters

Also allows writing dataframes to CSV files easily.

**Data Preprocessing**

Feature Engineering: Can encode non-numeric columns.

Data Splitting: Split datasets into training and testing sets with user defined ratios

**My addition**: Simple code to impute missing values in dataframes so many classifier models work

Feature-Target Separation: Split data frames into features and target variables

**Model Training**

Presumably the user will be experimenting with various models and different hyperparameters. This will be done through the user’s own code within the notebook interface and not be part of the reusable pipeline packaging code.

**Model Evaluation**

Compute metrics like accuracy, precision, recall, F1 score, and generate a confusion matrix

Visualization: Plot confusion matrices and classification metrics for notebook interface.

**Model Registry**

Storage & Retrieval: Save and load models with associated metadata like creation date, algorithm details, parameters, and performance metrics.

Listing: List all available trained models

**Model Prediction**

Allows for full dataset inference as well as single record inference on selected data.

**My Improvement to the Pipeline: Model Selection**

Organize models as candidates alongside their evaluation results. Compare models across multiple metrics the user may select. Allows for automatically selecting models based on configurable criteria (a specific metric such as f1 score). Allows for saving performance metrics within model metadata, which was originally unimplemented in the base repository.