Report MLOps

DSBA CLI Tool Documentation

This document explains how to use the DSBA CLI for training models, making predictions, and more.

Available Commands

1. List Available Models

```
./src/cli/dsba_cli list
```

Displays the list of trained and available models.

---

2. Preprocess and Split Dataset

```
./src/cli/dsba_cli preprocess --csv <file.csv> --target <target_column_name> --model_id <model_prefix>
```

Function: Preprocesses data, then splits into train/test (80/20 by default) and saves files in `models/<model_prefix>/`.

Generated files:

- `train.csv` - Training data
- `test.csv` - Test data
- `useful_columns.txt` - List of columns usable for predictions

---

3. Train Model with 5 Classifiers

```
./src/cli/dsba_cli train --csv <file.csv> --target <target_column_name> --model_id <model_prefix> [--algorithm <algo>]
```

Trained classifiers:

- XGBoost
- RandomForest
- LogisticRegression
- SVM
- DecisionTree

Options:

- `-algorithm`: Specify a particular algorithm or "all" (default: all)

The best model is automatically selected based on F1 score.

---

4. Make Predictions with Specific Model

```
./src/cli/dsba_cli predict_with_model --model <model_name> --input <input_file.csv> --output <output_file.csv>
```

Uses the specified model to predict the target on an input CSV file, then writes results to an output CSV file.

---

5. Display Available Evaluation Metrics

```
./src/cli/dsba_cli metrics
```

Displays available metrics for evaluating models with their definitions:

- **Accuracy**: Percentage of correct predictions
- **Precision**: Percentage of true positives among positive predictions
- **Recall**: Percentage of true positives detected
- **F1-Score**: Harmonic mean of precision and recall

---

6. Build Docker Image

```
./src/cli/dsba_cli build_image
```

Builds the FastAPI Docker image for deployment with the specified platform (linux/amd64).

---

7. Run Docker Container

```
./src/cli/dsba_cli run_container
```

Runs the FastAPI container with the following configuration:

- Port mapping: 8000:8000
- Container name: mlops-container
- Volume mount for models directory
- Environment variables loaded from .env file

Complete Usage Example

```
# 1. Preprocess the data
./src/cli/dsba_cli preprocess --csv data/titanic.csv --target Survived --model_id titanic_v1

# 2. Train the model with optimization
./src/cli/dsba_cli train --csv data/titanic.csv --target Survived --model_id titanic_v1 --algorithm all

# 3. List available models
./src/cli/dsba_cli list

# 4. Make a prediction
./src/cli/dsba_cli predict_with_model --model titanic_v1_xgboost --input data/test.csv --output results/predictions.csv
```

```
# 5. Check available metrics
./src/cli/dsba_cli metrics

# 6. Build Docker image for deployment
./src/cli/dsba_cli build_image

# 7. Run the API container
./src/cli/dsba_cli run_container
```

Prerequisites

- Python 3.x installed
- File marked as executable: `chmod +x src/cli/dsba_cli`
- Environment variables configured (notably `DSBA_MODELS_ROOT_PATH`)
- Dependencies installed according to project requirements.txt
- Docker installed (for image building and container running)
- `.env` file configured for Docker container environment

Generated File Structure

```
models/
└── <model_id>/
    ├── train.csv          # Training data
    ├── test.csv           # Test data
    ├── useful_columns.txt  # Columns for predictions
    └── <model_id>_<algo>.pkl  # Trained models
```

Docker Commands

Build Image

```
./src/cli/dsba_cli build_image
```

Builds a Docker image tagged as `fastapi-app` with linux/amd64 platform compatibility.

Run Container

```
./src/cli/dsba_cli run_container
```

Runs the container in detached mode with:

- Port 8000 exposed
- Models directory mounted as volume
- Environment variables from `.env` file

Documentation automatically generated - Updated according to project evolution.