**S1-24\_AIMLCZG523 – MLOps - Assignment 1**

**Group No. 120**

|  |  |  |
| --- | --- | --- |
| ***Name*** | ***StudentID*** | ***Contribution*** |
| **JOSHI NIRANJAN SURYAKANT** | **2023AC05011** | **100%** |
| **PRATEEK RALHAN** | **2023AC05673** | **100%** |
| **KESHARKAR SURAJ SANJAY** | **2023AD05004** | **100%** |
| **SAURABH SUNIT JOTSHI** | **2023AC05565** | **100%** |

**MLOps Architecture Summary - California Housing Price Prediction Pipeline**

***Project Overview***

**End-to-end** MLOps pipeline for California Housing Price Prediction using **Linear Regression and Decision Tree models, implementing industry best practices for model development, deployment, and monitoring**.

***Architecture Components***

1. Data Management & Versioning
   1. *Dataset:* California Housing dataset (housing.csv) with 8 features (MedInc, HouseAge, AveRooms, AveBedrms, Population, AveOccup, Latitude, Longitude)
   2. *Data Versioning:* DVC (Data Version Control) for tracking dataset changes
   3. *Data Pipeline:* Automated data loading via `src/utils.py` using scikit-learn's fetch\_california\_housing
   4. *Storage:* Local data directory with DVC tracking files
2. Model Development & Training
   1. *Algorithms*: Multiple model comparison (Linear Regression, Decision Tree, Ridge Regression)
   2. *Configuration*: YAML-based config (`src/config.yaml`) with hyperparameters (alpha, test\_size, random\_state, max\_depth)
   3. *Training Pipeline*: `src/train.py` with automated model selection based on MSE
   4. *Model Storage*: Joblib serialization to `models/ridge\_model.pkl`
   5. *Code Quality*: Pre-commit hooks (Black, Flake8, Prettier) for code formatting and linting
3. Experiment Tracking & Model Registry
   1. *Platform*: MLflow for experiment tracking and model versioning
   2. *Tracking*: Automatic logging of parameters, metrics (MSE), and model artifacts
   3. *Model Registry*: Registered models with versioning (`BestHousingModel`)
   4. *Storage*: SQLite backend (`mlflow\_data/mlflow.db`) and local artifact storage (`mlruns/`)
   5. *UI*: MLflow tracking server on port 5555
4. API development & Deployment
   1. *Framework*: FastAPI with automatic OpenAPI documentation
   2. *Endpoints*:
      1. `/predict` - Housing price predictions
      2. `/metrics` - Prometheus metrics exposure
      3. `/retrain` - Model retraining trigger
   3. *Input Validation*: Pydantic models for request validation
   4. *Containerization*: Docker with multi-stage build process
   5. *Docker* *Hub*: Published image (`niranjanjoshi14/housing-api:latest`)
5. Monitoring & Observability
   1. *Metrics Collection*: Prometheus client integration with custom metrics

- `prediction\_requests\_total` - Request counter

- `prediction\_latency\_seconds` - Response time histogram

* 1. *Database Logging:* SQLite database (`prediction\_logs.db`) for prediction history
  2. *Monitoring Stack:*

- Prometheus (port 9090) for metrics scraping

- Grafana (port 3000) for visualization dashboards

* 1. *Logging:* Structured logging with timestamps and input/output data

1. CI/CD Pipelines
   1. *Platform*: GitHub Actions with automated workflows
   2. *Triggers*: Push to main branch, pull requests, data changes
   3. *Pipeline Stages*:

1. Code checkout and Python setup

2. Dependency installation and caching

3. Code quality checks (Flake8, Black)

4. Docker image building and publishing

5. Model training validation

6. Automated testing

* 1. *Security*: Docker Hub authentication via secrets

1. Infrastructure & Orchestration
   1. *Container Orchestration*: Docker Compose with 4 services

- API service (FastAPI)

- Prometheus (metrics collection)

- Grafana (visualization)

- MLflow (experiment tracking)

* 1. *Networking*: Internal service communication with external port mapping
  2. *Volumes*: Persistent storage for Grafana and MLflow data
  3. *Environment*: Python 3.10 with all dependencies in requirements.txt

1. Data Pipeline Automation
   1. *DVC Pipeline*: Automated training pipeline (`dvc.yaml`)
   2. *Dependencies*: Clear dependency tracking between data, code, and models
   3. *Reproducibility*: Deterministic training with fixed random seeds
   4. *Artifact* *Management*: Automatic model artifact generation and tracking

***Key Metrics & KPIs***

* *Model Performance:* MSE (Mean Squared Error) tracking
* *API Performance:* Request latency, throughput, error rates
* *System Health:* Container status, resource utilization
* *Data Quality:* Input validation success rates
* *Deployment Success:* CI/CD pipeline success rates

This architecture demonstrates a **production-ready MLOps pipeline with comprehensive monitoring, automated deployment, and scalable infrastructure following industry best practices.**

***Appendix:***

* Github repository : l[ink 🔗](https://github.com/niranjanjoshi/MLOPS-Pipeline)
* Video demo: l[ink 🔗](https://wilpbitspilaniacin0-my.sharepoint.com/personal/2023ac05011_wilp_bits-pilani_ac_in/_layouts/15/stream.aspx?id=%2Fpersonal%2F2023ac05011%5Fwilp%5Fbits%2Dpilani%5Fac%5Fin%2FDocuments%2FMLOPS%20Assignment%2Emp4&ga=1&referrer=StreamWebApp%2EWeb&referrerScenario=AddressBarCopied%2Eview%2Ef19a960e%2D863f%2D4fb7%2Db71d%2D34fe3129822d)
* Published docker image: [link 🔗](https://hub.docker.com/r/niranjanjoshi14/housing-api)
* System Architecture (mermaid diagram): [link 🔗](https://tinyurl.com/azwzfeud)