Personality Detection Project with DVC, MLflow, and DAGsHub

This project demonstrates the integration of Data Version Control (DVC), MLflow, and DAGsHub for a machine learning project that predicts personality types (Introvert/Extrovert) based on various behavioral patterns.

■ Features

- Personality prediction using machine learning
- Data version control with DVC
- Experiment tracking with MLflow
- Remote tracking with DAGsHub
- Model versioning and management
- Metrics and parameter logging
- Artifact storage and management

■ Prerequisites

- Python 3.8+
- Git
- DAGsHub account
- Required Python packages (see requirements.txt): `bash pip install -r requirements.txt`

■ Project Structure

```
personality-detection-project/
■■■ data/
■   ■■■ personality_dataset.csv    # Dataset for personality prediction
■■■ .dvc/                         # DVC configuration and cache
■■■ mlruns/                       # MLflow tracking directory
■■■ ml-project.py                 # Main ML implementation
■■■ requirements.txt              # Project dependencies
■■■ README.md                     # Project documentation
```

■■■■ Setup and Running

Clone the repository: `bash git clone https://github.com/vijaytakbhate2002/basic-personality-detection-project-with-dvc-mlflow-dagshub-git.gi`
`cd basic-personality-detection-project-with-dvc-mlflow-dagshub-git`

Set up DAGsHub credentials: `bash export MLFLOW_TRACKING_URI=https://dagshub.com/<username>/<repo-name>.mlflow export MLFLOW_TRACKING_USERNAME=<your-dagshub-username> export MLFLOW_TRACKING_PASSWORD=<your-dagshub-token>`

Install dependencies: `bash pip install -r requirements.txt`

Initialize DVC: `bash dvc init`

Add data to DVC: `bash dvc add data/personality_dataset.csv`

Run the ML project: `bash python ml-project.py`

■ Features Used for Prediction

- Time spent alone
- Stage fear
- Social event attendance
- Going outside frequency
- Energy level after socializing
- Friends circle size
- Social media post frequency

■ Technologies Used

DVC (Data Version Control)

- Tracks data versions
- Manages large data files
- Enables data sharing and collaboration

MLflow

- Experiment tracking
- Parameter logging
- Metric tracking
- Model versioning
- Artifact storage

DAGsHub

- Remote experiment tracking
- Model registry
- Collaboration platform
- Data versioning

■ Model Performance

The model's performance is tracked using various metrics: - Accuracy - Precision - Recall - F1 Score

■ Environment Variables

Required environment variables: - `MLFLOW_TRACKING_URI`: Your DAGsHub MLflow tracking URI - `MLFLOW_TRACKING_USERNAME`: Your DAGsHub username - `MLFLOW_TRACKING_PASSWORD`: Your DAGsHub token

■ Contributing

1. Fork the repository

2. Create your feature branch (`git checkout -b feature/AmazingFeature`)
3. Commit your changes (`git commit -m 'Add some AmazingFeature'`)
4. Push to the branch (`git push origin feature/AmazingFeature`)
5. Open a Pull Request

■ License

This project is licensed under the MIT License - see the LICENSE file for details.

■ Author

- Vijay Takbhate

■ Acknowledgments

- DVC team for the amazing data version control tool
- MLflow team for experiment tracking
- DAGsHub for providing the remote tracking infrastructure