Manufacturing Predictive Downtime Analysis API

Overview

This project implements a machine learning-powered RESTful API to predict manufacturing machine downtime using Python, Flask, and scikit-learn.

Features

- Synthetic data generation
- Machine learning model training
- Predictive downtime analysis
- RESTful API endpoints for data upload, model training, and predictions

Prerequisites

- Python 3.8+
- pip

Installation

1. Clone the repository

bash
Copy

```
git clone
https://github.com/yourusername/manufacturing-predictive-api.git
cd manufacturing-predictive-api
```

2. Create a virtual environment

bash
Copy

```
python -m venv venv
source venv/bin/activate  # Unix/macOS
# Or
```

```
venv\Scripts\activate     # Windows
```

3. Install dependencies

bash
Copy

```
pip install -r requirements.txt
```

Project Structure

Copy

```
manufacturing-predictive-api/
│
├── data/                # Store uploaded CSV files
├── models/              # Trained model and scaler storage
├── app.py               # Flask API application
├── train_model.py       # Model training and data generation script
├── requirements.txt     # Project dependencies
└── README.md            # Project documentation
```

Usage

Generate Initial Model

bash
Copy

```
python train_model.py
```

This script generates synthetic data and trains an initial machine learning model.

Run API Server

bash
Copy

```
python app.py
```

The API will start on `http://localhost:5000`

API Endpoints

1. Upload Data

- **Endpoint**: `POST /upload`
- **Description**: Upload manufacturing CSV data
- **Request**: Multipart form-data with 'file' key

2. Train Model

- **Endpoint**: `POST /train`
- **Description**: Train model on uploaded data
- **Returns**: Model performance metrics (accuracy, F1 score)

3. Predict Downtime

- **Endpoint**: `POST /predict`
- **Description**: Predict machine downtime probability
- **Request Body**:

json
Copy
```
{
  "Temperature": 85.5,
  "Run_Time": 150
}
```

- **Response**:

json
Copy
```
{
  "Downtime": "Yes/No",
  "Confidence": 0.85
}
```

Testing Endpoints

Using cURL

bash
Copy
```
# Upload data
curl -F "file=@data/manufacturing_data.csv"
http://localhost:5000/upload

# Train model
```

```
curl -X POST http://localhost:5000/train

# Predict downtime
curl -X POST http://localhost:5000/predict \
     -H "Content-Type: application/json" \
     -d '{"Temperature": 80, "Run_Time": 120}'
```

Model Details

- Algorithm: Decision Tree Classifier
- Features: Temperature, Run Time
- Target: Machine Downtime Flag
- Data: Synthetic or user-uploaded

Customization

- Modify `train_model.py` to adjust data generation rules
- Update feature selection in model training
- Experiment with different machine learning algorithms

Potential Improvements

- Add more feature engineering
- Implement more advanced ML models
- Create more comprehensive error handling
- Add logging and monitoring