■ Medical Insurance Cost Prediction (SVR)

Python 3.x Flask 2.x Docker ready Kubeflow Pipeline CI CD

This project predicts **medical insurance charges** based on demographic and lifestyle factors such as **age, BMI, smoking habits, and region**. The model is built using **Support Vector Regression (SVR)** and fine-tuned with **GridSearchCV**.

■ Workflow (End-to-End MLOps)

I have implemented complete **MLOps practices** for this project:

1. **Experiments with Jupyter Notebook** → finalized best model and preprocessing steps.
2. **Packaged Data Processing & Training Pipelines** → reproducible results with Label Encoding, MinMax Scaling, and scaled targets for SVR.
3. **GitHub Actions** → CI pipeline with tests for continuous & trustworthy deployment.
4. **GridSearchCV** → tested 458 parameter combinations to find the best fit model.
5. **AWS (MLflow Tracking)** → ran 4–5 experiments, tracked remotely on EC2, and learned remote URI tracking.
6. **Model Loading** → used `pyfunc` to fetch best model from AWS to local.
7. **Flask Application** → built an interactive UI for easy predictions.
8. **Containerization with Docker** → made project machine-independent and portable.
9. **Kubeflow Pipelines** → used Minikube for local K8s experiments, created pipeline with data processing & model training components.
10. **AWS Deployment** → used AWS EC2 instance for flask app deployment

Deployed app (AWS EC2 instance)

■ App will be available at: **http://ec2-54-234-204-18.compute-1.amazonaws.com:5000/**

```
               ■■■■■■■■■■■■■■■■■■■
              ■    Raw Data     ■
              ■■■■■■■■■■■■■■■■■■■
                      ■
            ■■■■■■■■■■■▼■■■■■■■■■■■
            ■ Preprocessing      ■
            ■■■■■■■■■■■■■■■■■■■■■■
                      ■
            ■■■■■■■■■■■▼■■■■■■■■■■■
            ■ Model Training     ■ (SVR)
            ■■■■■■■■■■■■■■■■■■■■■■
                      ■
            ■■■■■■■■■■■▼■■■■■■■■■■■
            ■ Flask App (UI)     ■
            ■  ■■■■■■■■■■■■■■■■■  ■
            ■  User enters input ■
            ■   → Prediction made ■
            ■■■■■■■■■■■■■■■■■■■■■■
                      ■
            ■■■■■■■■■■■▼■■■■■■■■■■■
```

```
          ■ Docker Container   ■
          ■■■■■■■■■■■■■■■■■■■■■■
                  ■
          ■■■■■■■■■■■▼■■■■■■■■■■■
          ■ Kubeflow Pipeline  ■
          ■   (Automation)     ■
          ■■■■■■■■■■■■■■■■■■■■■■
                  ■
          ■■■■■■■■■■■▼■■■■■■■■■■■
          ■ Cloud Deployment   ■
          ■■■■■■■■■■■■■■■■■■■■■■■
```

■ Results

Base SVR Model

MSE: 0.0107 | MAE: 0.099 | RMSE: 0.103 | Train R^2: 0.698 | Test R^2: 0.728

Tuned SVR (GridSearchCV) ■

MSE: 0.0053 | MAE: 0.034 | RMSE: 0.073 | Train R^2: 0.837 | Test R^2: 0.865

■ Project Structure

```
medical-insurance-cost-prediction-SVR/
■
■■■ data/                          # Dataset
■   ■■■ raw_data/insurance.csv
■   ■■■ processed_data/
■
■■■ jupyter_code/                  # Experimentation notebooks
■■■ src/                           # Core source code
■   ■■■ data_processing.py
■   ■■■ training_pipeline.py
■   ■■■ ...
■
■■■ processing_pipeline_runner.py     # Run preprocessing
■■■ training_pipeline_runner.py       # Run training
■■■ app.py                         # Flask application
■■■ requirements.txt
■■■ README.md
```

■■ Project Interface

■ Built a **Flask web application** with an interactive UI for **user-friendly prediction** of medical insurance costs. Users can input **age, sex, BMI, children, smoker status, and region** → and instantly get the predicted charges.

■■ Kubeflow Pipeline Status

■ Running the Project

Run with Docker

```
docker pull vijaytakbhate1/medical_insurance_cost_prediction_svr:latest
docker run -d -p 5000:5000 vijaytakbhate1/medical_insurance_cost_prediction_svr:latest
```

■ App will be available at: **http://localhost:5000**

Run without Docker

```
git clone https://github.com/vijaytakbhate2002/medical-insurance-cost-prediction-SVR.git
cd medical-insurance-cost-prediction-SVR

python -m venv venv
source venv/bin/activate   # Linux/Mac
venv\Scripts\activate      # Windows

pip install -r requirements.txt
python processing_pipeline_runner.py
python training_pipeline_runner.py
python app.py
```

■ Deployed Application

■ URL pending (to be added after deployment)

■■ Tech Stack

- Python 3.x
- Scikit-learn
- Pandas, NumPy
- Flask
- Docker
- Kubeflow (Minikube)
- AWS (MLflow, EC2)

■ Future Roadmap

✔■ Completed

- Flask application with interactive UI for predictions.

- Dockerized for machine-independent setup.
- Kubeflow pipelines for reproducible ML workflows.
- CI/CD with GitHub Actions.

■ **Planned**

- Streamlit version for modern UI and easier sharing.
- Experiment with ensemble models (Random Forest, XGBoost).
- Integrate cloud-native deployment (AWS EKS / GCP GKE).
- Add monitoring & logging for deployed model.

✍■ **Author**: Vijay Dipak Takbhate
