Hospital Waiting Time Prediction Project

December 19, 2025

1 Problem Statement

This project addresses the problem of predicting hospital patient waiting times in minutes. Accurate predictions can help hospitals optimize patient flow, allocate resources efficiently, and improve patient satisfaction. This is formulated as a regression problem, where the target variable is the patient waiting time.

2 Dataset and Model Description

2.1 Dataset

The dataset consists of hospital patient records including:

- Patient demographics
- Appointment details
- Consultation type
- Financial information

2.2 Features

Categorical: `doctor_type`, `financial_class`, `patient_type`
Numerical: `medication_revenue`, `lab_cost`, `consultation_revenue`, `entry_hour`, `entry_dayofweek`, `entry_minute`, `year`, `month`, `dayofweek`
Target: `waiting_time` (in minutes)

2.3 Modeling

- Preprocessing: One-hot encoding for categorical features, log transformation of the target variable
- Models trained:
  - Random Forest Regressor

  - Gradient Boosting Regressor
  - XGBoost Regressor

- Best model: XGBoost Regressor

2.4 Performance Metrics

Model	MAE (minutes)	RMSE (minutes)	R^2
Random Forest	22.53	1849.22	0.0429
Gradient Boosting	22.35	1842.53	0.0463
XGBoost	22.35	1836.65	0.0494

Table 1: Model performance metrics

Observation: XGBoost slightly outperforms the other models. Low R^2 values indicate that waiting time depends on unobserved or noisy factors, consistent with prior hospital scheduling studies.

3 Dockerization Steps

1. Create `Dockerfile`:

```
FROM python:3.10-slim
WORKDIR /app
COPY requirements.txt .
RUN pip install --no-cache-dir -r requirements.txt
COPY app.py .
COPY hospital_waiting_model.pkl .
EXPOSE 8000
CMD ["uvicorn", "app:app", "--host", "0.0.0.0", "--port", "
  ↪ 8000"]
```

2. Build the Docker image:

```
docker build -t hospital-waiting-api .
```

3. Run locally:

```
docker run -p 8000:8000 hospital-waiting-api
```

4. Push to Docker Hub:

```
docker tag hospital-waiting-api <dockerhub-username>/
   ↪ hospital-waiting-api:latest
docker push <dockerhub-username>/hospital-waiting-api:
   ↪ latest
```

4 API Endpoints

- **/healthz** [GET]: Returns {`"status": "ok"`} to confirm the service is running
- **/predict** [POST]: Accepts patient data as JSON and returns predicted waiting time in minutes

Example /predict JSON:

```
{
  "doctor_type": "General",
  "financial_class": "Insurance",
  "patient_type": "New",
  "medication_revenue": 50,
  "lab_cost": 30,
  "consultation_revenue": 100,
  "entry_hour": 10,
  "entry_dayofweek": 2,
  "entry_minute": 15,
  "year": 2024,
  "month": 5,
  "dayofweek": 2
}
```

5 Kubernetes Deployment Steps

1. Create deployment:

```
kubectl create deployment hospital-waiting --image=<
   ↪ dockerhub-username>/hospital-waiting-api:latest
```

2. Expose service:

```
kubectl expose deployment hospital-waiting --type=NodePort
   ↪ --port=8000
```

3. Apply health probes:

```
kubectl set probe deployment/hospital-waiting --liveness --
    ↪ get-url=http://:8000/healthz --initial-delay-seconds
    ↪ =10
kubectl set probe deployment/hospital-waiting --readiness
    ↪ --get-url=http://:8000/healthz --initial-delay-
    ↪ seconds=5
```

4. Export YAML:

```
kubectl get deployment hospital-waiting -o yaml >
    ↪ deployment.yaml
kubectl get service hospital-waiting -o yaml > service.yaml
```

6 Health Checks

- **Liveness Probe**: Automatically restarts pods if '/healthz' fails
- **Readiness Probe**: Ensures traffic is only sent to ready pods

7 Horizontal Pod Autoscaler Configuration

- Minimum pods: 1
- Maximum pods: 5
- CPU threshold: 50%

Apply via kubectl:

```
kubectl autoscale deployment hospital-waiting --cpu=50% --min=1
    ↪  --max=5
kubectl get hpa
```

8 Comparison to Related Works

The XGBoost model achieved:

- MAE = 22.35 minutes
- RMSE = 1836.65 minutes
- $R^2 = 0.0494$

Low R^2 is consistent with prior studies predicting hospital waiting times, where many unrecorded factors influence patient flow. Ensemble tree models (Random Forest, Gradient Boosting, XGBoost) are appropriate for tabular hospital data and provide similar predictive performance.