# House Price Prediction - Final Project Report

## 1. Introduction

This project aims to build and deploy a machine learning model for predicting house prices based on real-world features. Using the **California Housing Dataset**, we trained multiple regression models, optimized hyperparameters, and deployed the best-performing model using a Flask API hosted on Azure.

## 2. Data Preprocessing & Feature Engineering

### 2.1 Data Cleaning

* Loaded the **California Housing Dataset** using **sklearn.datasets**.
* Checked for **missing values**, ensuring data completeness.
* Handled outliers using **Interquartile Range (IQR) Method**.
* Applied **log transformation** to MedHouseVal to normalize skewed price distribution.

### 2.2 Feature Engineering

To improve model performance, we derived the following additional features:

* **RoomsPerHousehold** = AveRooms / AveOccup
* **BedrmsPerRoom** = AveBedrms / AveRooms
* **PopPerHousehold** = Population / AveOccup

These new features helped capture meaningful relationships between room count, occupancy, and household distribution.

## 3. Model Selection & Optimization

### 3.1 Baseline Model

* Trained a **RandomForestRegressor** as a baseline model.
* Achieved an initial **R² score of 0.8167**.

### 3.2 Hyperparameter Tuning

* Used GridSearchCV and optuna to fine-tune hyperparameters for **RandomForest and XGBoost**.
* Final optimized model: **XGBoost**, with:
  + n\_estimators=443
  + max\_depth=12
  + learning\_rate=0.015706
  + subsample=0.7077
  + colsample\_bytree=0.9181
* Best **R² Score: 0.86188.**

### 3.3 Model Evaluation

|  |  |  |  |
| --- | --- | --- | --- |
| **Model** | **MAE** | **RMSE** | **R² Score** |
| **Random Forest (Baseline)** | 0.04583 | 0.00440 | 0.84448 |
| **Optimized XGBoost** | 0.04322 | 0.00391 | 0.86188 |

XGBoost was chosen for deployment due to its **higher accuracy and generalization ability**.

## 4. Deployment Strategy

### 4.1 Flask API Implementation

* Developed a **Flask API (app.py)** to serve the model.
* Created an API **endpoint (/predict)** to receive user inputs in JSON format and return house price predictions.

### 4.2 Docker Containerization

* Packaged the Flask app into a **Docker container** using a Dockerfile.
* Built and ran the container locally for testing:
* docker build -t house\_price\_api .

docker run -p 5000:5000 house\_price\_api

### 4.3 Azure Deployment

* Created an **Azure App Service** to host the model.
* Pushed the **Docker image** to Azure Container Registry:
* az acr create --resource-group HousePriceGroup --name housepriceacr --sku Basic
* docker tag house\_price\_api housepriceacr.azurecr.io/house\_price\_api:v1

docker push housepriceacr.azurecr.io/house\_price\_api:v1

* Deployed the container to **Azure Web App**:
* az webapp create --resource-group HousePriceGroup --plan HousePricePlan \

--name house-price-app --deployment-container-image-name housepriceacr.azurecr.io/house\_price\_api:v1

### 4.4 API Usage Guide

* **Access the API** at:

https://house-price-app.azurewebsites.net/predict

* **Request Format (JSON):**

{

"MedInc": 8.6,

"HouseAge": 23,

"AveRooms": 6,

"AveBedrms": 5,

"Latitude": 37.45,

"Longitude": -122.14,

"Population": 1000,

"AveOccup": 3

}

* **Response:**

{

"Predicted Price": 3.5

}

## 5. Front-End UI

* Designed a **simple Bootstrap-based UI (index.html)** for user-friendly interaction.
* Users can enter input values and receive predictions in real-time.
* Hosted alongside the Flask app for seamless access.

## 6. Conclusion & Next Steps

### Achievements

Data preprocessing and feature engineering improved model accuracy. XGBoost with hyperparameter tuning achieved **best performance (R² = 0.8216).** Model successfully deployed on **Azure App Service with Flask & Docker.** ✔ Simple **web UI** built for easy user interaction.

### Future Enhancements

* **Deploy on AWS Lambda** for cost-efficient serverless inference.
* **Enhance UI with React.js** for better user experience.

Project Repository:[**GitHub Link**](https://github.com/Yaswanthv5/House_Price_Prediction.git)

**Authors:** YaswanthReddy Varikuti  
**License:** Apache  
**Contact:** yaswanthreddyvarikuti@gmail.com