# House Price Prediction - Project Report

**Part 1: Data Preprocessing and Feature Engineering**

**Data Overview**

* **Dataset Used:** California Housing Dataset (from Scikit-learn)
* **Number of Rows:** 20,640
* **Number of Features:** 9
* **Target Variable:** MedHouseVal (Median House Value)

**Steps Taken**

1. **Data Cleaning:**
   * Verified no missing values were present in the dataset.
   * Confirmed that all features were numerical with no categorical values.
2. **Outlier Treatment:**
   * Detected outliers using the **IQR (Interquartile Range)** method.
   * Applied **capping** for extreme values using the 5th and 95th percentiles to maintain data integrity.
3. **Feature Scaling:**
   * Applied StandardScaler to normalize the feature values for improved model performance.
4. **EDA (Exploratory Data Analysis):**
   * Visualized feature distributions using sns.histplot() and sns.scatterplot().
   * Examined correlation between features and the target variable to identify key predictors.

**Part 2: Model Selection and Optimization**

**Models Implemented**

* **Linear Regression**
* **Decision Tree Regressor**
* **Random Forest Regressor**
* **XGBoost Regressor**
* **Voting Regressor** (ensemble of multiple models)
* **Stacking Regressor** (layered ensemble approach)

**Model Evaluation (RMSE Scores)**

* **Linear Regression:** 0.4549
* **Decision Tree Regressor:** 0.5049
* **Random Forest Regressor:** 0.2543
* **XGBoost Regressor:** **0.2213** (Best Performing Model)
* **Voting Regressor:** 0.3036
* **Stacking Regressor:** 0.4115

**Hyperparameter Tuning**

* Utilized **RandomizedSearchCV** for tuning:
  + **XGBoost Best Parameters:**
    - subsample: 1.0
    - n\_estimators: 400
    - max\_depth: 10
    - learning\_rate: 0.008
    - colsample\_bytree: 0.9

**Part 3: App Development**

**Flask Application Development**

* Created a REST API using Flask.
* Added a /predict endpoint to receive JSON data and return predicted house prices.
* Integrated pickle for loading the trained model efficiently.

**Deployment**

* Used **Ngrok** to expose the local Flask server for public testing.
* The app accepts user input in JSON format and returns predicted prices based on model inference.

**Sample API Request Format**

{

"MedInc": 8.3252,

"HouseAge": 41,

"AveRooms": 6.984127,

"AveBedrms": 1.02381,

"Population": 322,

"AveOccup": 2.555556,

"Latitude": 37.88,

"Longitude": -122.23

}

**Sample API Response Format**

{

"predicted\_price": 3.52

}

**Part 4: Future Enhancements**

* Add more advanced feature engineering techniques.
* Implement additional ML models like LightGBM and CatBoost for potential performance improvements.
* Enhance the frontend interface for better user interaction.
* Deploy the app on cloud platforms like AWS, Azure, or Google Cloud for scalability.

**Conclusion**

This project successfully demonstrates the end-to-end process of building, training, optimizing, and deploying a machine learning model for predicting house prices. The XGBoost model delivered the best performance with optimal RMSE scores and effective hyperparameter tuning.

For further queries or feedback, feel free to reach out via [GitHub](https://github.com/nachidixit) or email at [**erdnachiket@gmail.com**](mailto:erdnachiket@gmail.com).