House Price Prediction using Neural Networks

Team : Fuzzy Wizards

TEAM MEMBERS

E/19/143 - Induwara G.L.A

E/19/203 - Kulugammana V.R.B.

E/19/338 - T.Sajan

E/19/369 - Shanilka H.Y.P.

E/19/447 - Wijesiri W.N.G.P.R.

PROBLEM STATEMENT

“The project focuses on predicting house prices in Bengaluru using a neural network model. The goal is to develop a predictive model that accurately estimates house prices based on various features such as total square feet, bedrooms , bathrooms, and location. The problem is relevant as real estate pricing is influenced by multiple factors, and an accurate model can assist buyers, sellers, and real estate professionals in making informed decisions”

INTRODUCTION

- Estimating house prices is crucial for buyers, sellers, and real estate companies.
- Traditional methods rely on manual valuation and rule-based models.
- Machine learning & neural networks improve predictions by analyzing complex patterns.

NEURAL NETWORK ARCHITECTURE

Neural Network Architecture: Multilayer Perceptron (MLP)

Why MLP for House Price Prediction?

- MLP is well-suited for structured tabular data, where inputs are numerical and categorical features.
- Unlike CNNs (Convolutional Neural Networks), which are designed for images, and RNNs (Recurrent Neural Networks), which work well with sequential data, MLP efficiently captures relationships in tabular datasets.

ARCHITECTURE DETAILS

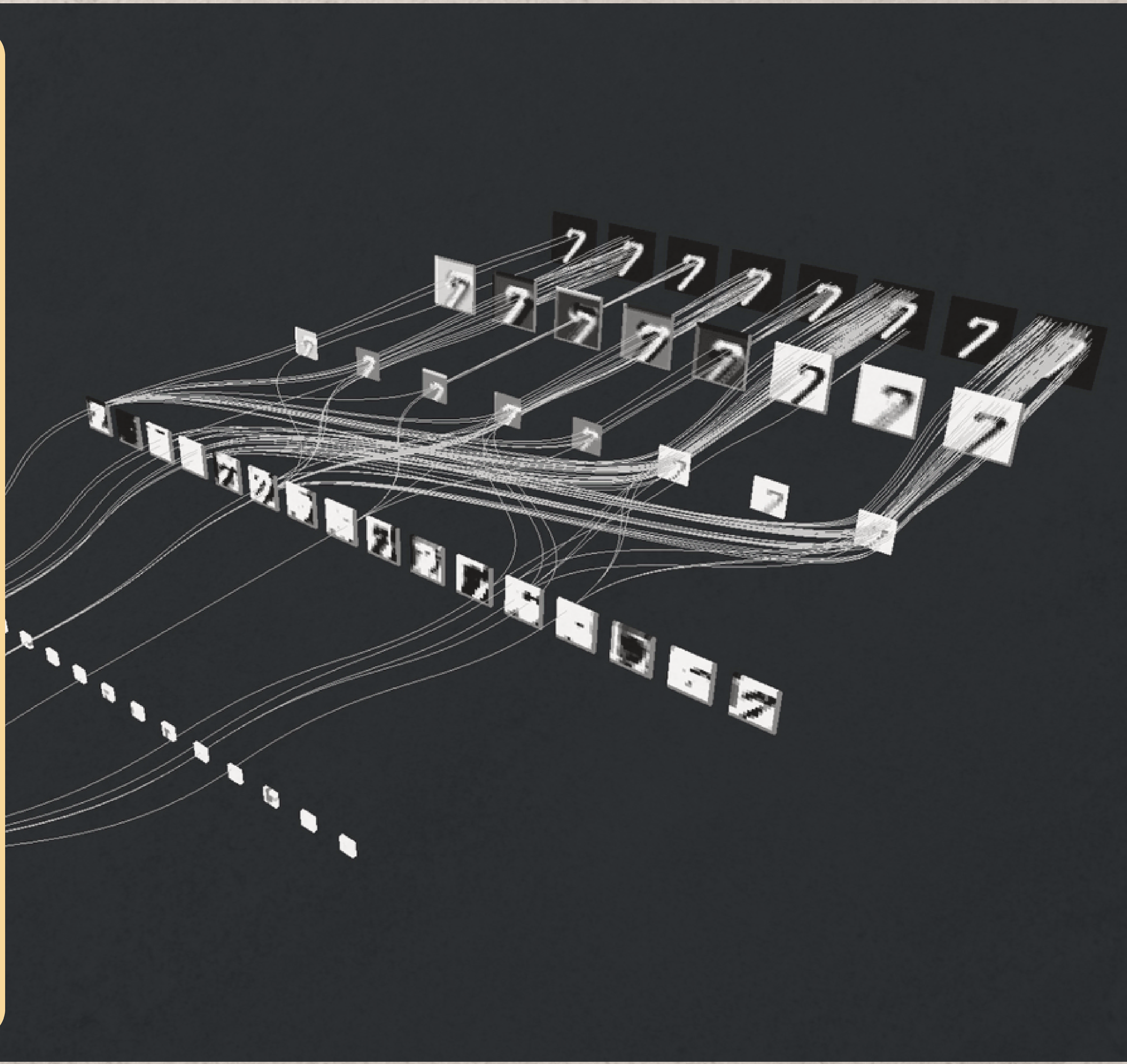
4 Layer Neural Network is built with:

- **Input Layer :** 64 neurons with ReLU activation function
  - Total Square feet
  - BHK
  - Bathrooms
  - Location
- **2 Hidden Layers :** 32 neurons with ReLU activation function
- **Output Layer :** 1 neuron
  - House price

ARCHITECTURE DETAILS

- **Optimizer :** Adam
  - Efficient for training deep learning models
  - Adapts learning rates for different parameters

- **Loss Function :** Mean Squared Error (MSE)
  - Suitable for regression problems
  - Helps in capturing the overall variance of house price prediction



DATASET

- **Dataset used :** Bengaluru House price data, from kaggle
- **Default features :**
  - area type
  - availability
  - society
  - balcony
  - location
  - total squarefeet
  - size (BHK)
  - bathroom
- Out of this 8 features, we drop 4 features that not affects to the price

area_type	availability	location	size	society	total_sqft	bath	balcony	price
Super built-up	19-Dec	Electronic City	2 BHK	Coomee	1056	2	1	39.07
Plot Area	Ready To Move	Chikka Tirupati	4 Bedroom	Theanmp	2600	5	3	120
Built-up Area	Ready To Move	Uttarahalli	3 BHK		1440	2	3	62

PREPROCESSING

- Find the datapoints with missing values and drop that datapoints.

	0
location	1
size	16
total_sqft	0
bath	73
price	0

- Size values appear under different names (BHK and Bedroom) in the same column, causing potential training errors.

size
2 BHK
4 Bedroom
3 BHK

BHK
2
4
3

PREPROCESSING

- Convert square feet ranges to single values using the average.

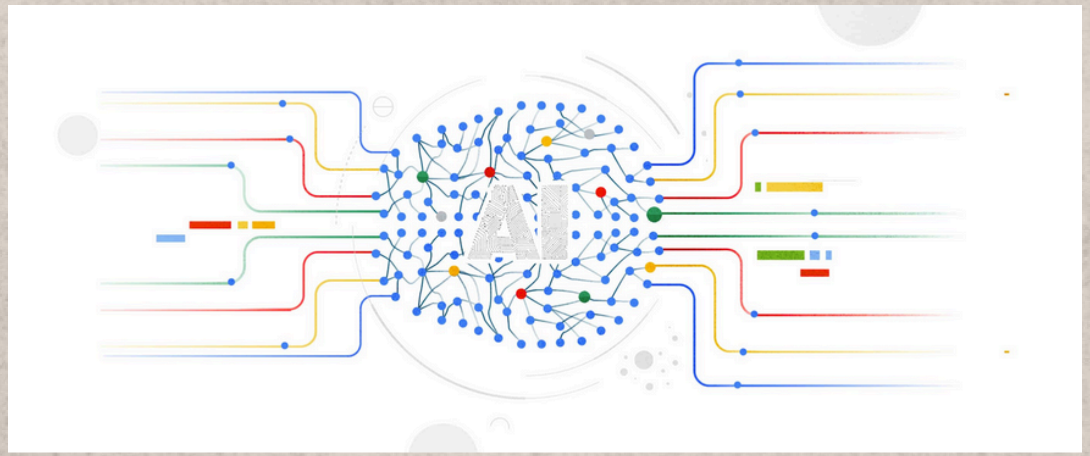
total_sqft
1025
2100 - 2850
3010 - 3410

- Converts location names into a numerical labels using **LabelEncoder()**

```
{' Anekal': 0, ' Banaswadi': 1,
' Basavangudi': 2, ' Bhoganhalli': 3,
```

TRAINING PROCESS

- Dataset split: 80% training, 20% testing.
- Data normalized using StandardScaler for better training convergence
- Batch size : 32
- Epochs : 100 -> Stopped early if no improvement



Layer (type)	Output Shape	Param #
dense_4 (Dense)	(None, 64)	320
dense_5 (Dense)	(None, 32)	2,080
dense_6 (Dense)	(None, 16)	528
dense_7 (Dense)	(None, 1)	17

Total params: 2,945 (11.50 KB)
Trainable params: 2,945 (11.50 KB)
Non-trainable params: 0 (0.00 B)

HYPERPARAMETER TUNING

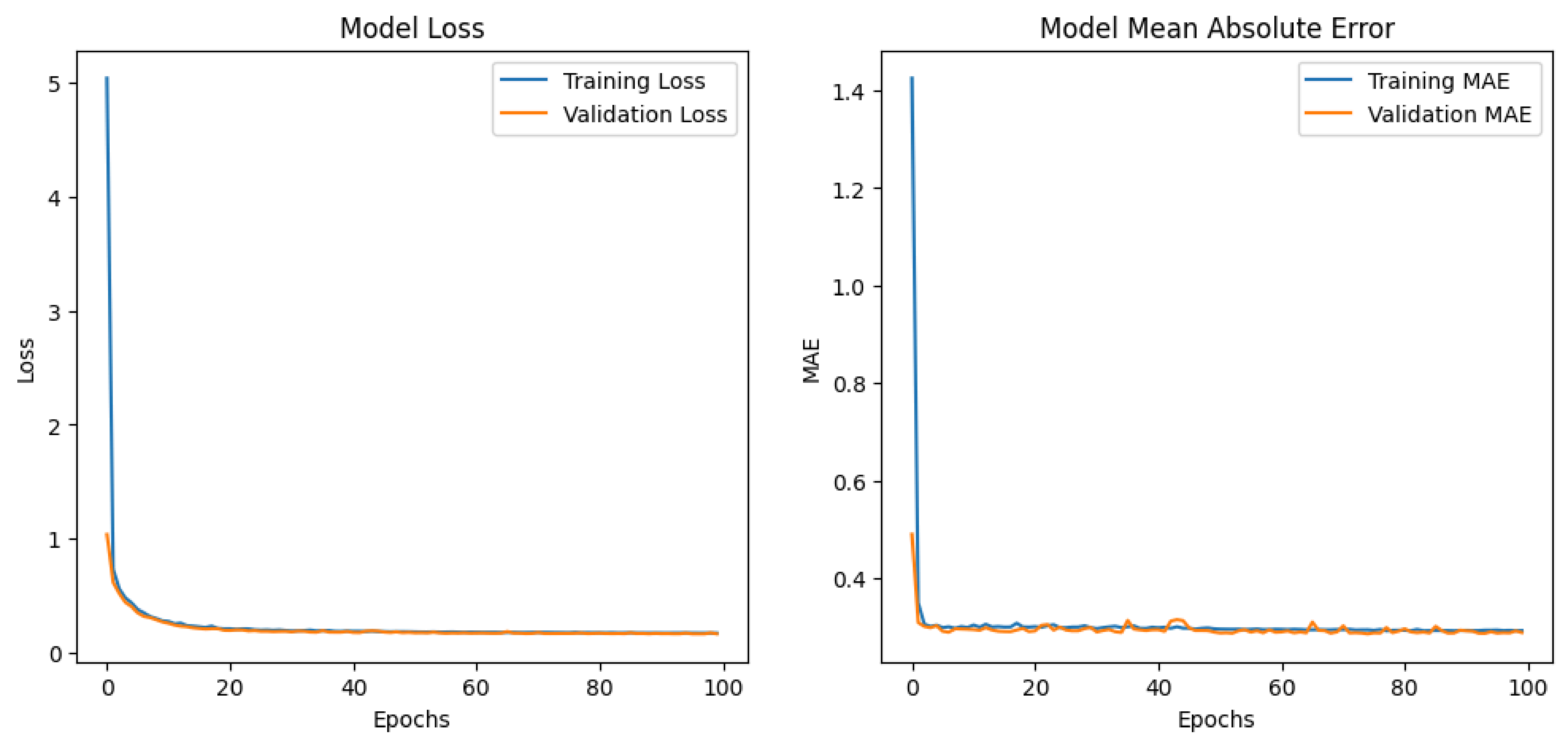
- Learning Rate : 0.001
  - Too high → unstable training; too low → slow convergence.
- Number of Layers & Neurons: More layers and neurons can capture complex patterns but may lead to overfitting. (4 Layers used)
- Batch Size: Number of samples processed before updating weights. Small → better generalization, large → faster training. (32 batches used)
- Epochs: Number of full training cycles. Too many → overfitting, too few → underfitting. (100 Epochs)
- No drop out Layers were used because model validation occurs properly.

EVALUATION METRICS

1. **Mean Absolute Error (MAE):**
   - Measures the average magnitude of errors in the model’s predictions
2. **Root Mean Squared Error (RMSE):**
   - Useful when we need to understand the average magnitude of the error in real-world terms.
3. **R-squared (R^2):**
   - Indicates the proportion of variance in the target variable (House price) explained by the model

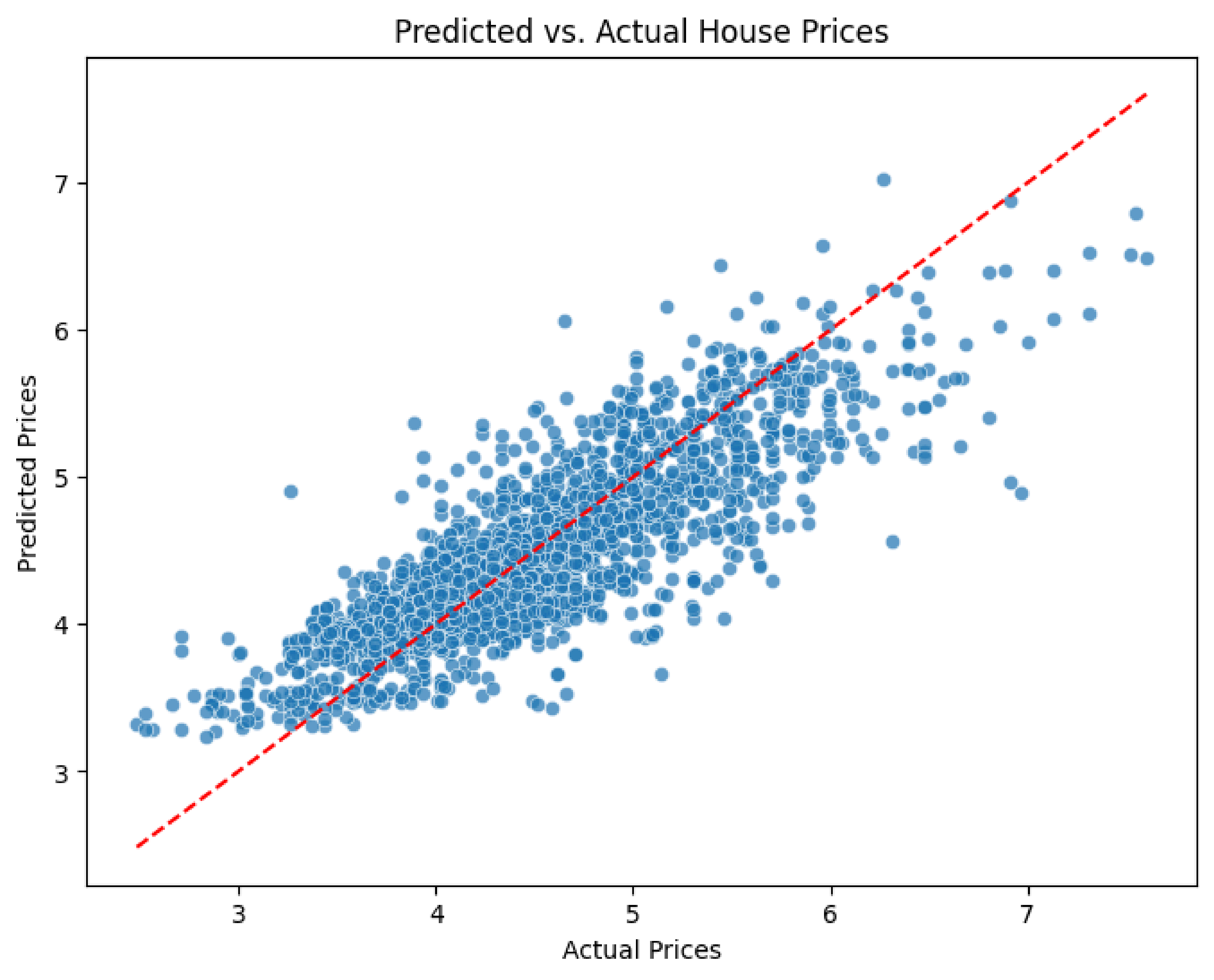
RESULTS

- **Comparison of Training vs Validation results**
  - Graphical representation of model loss after the hyperparameter tuning can be shown as follows



RESULTS

- **Predicted vs. Actual House Prices**



- Red dashed line represents an ideal perfect prediction

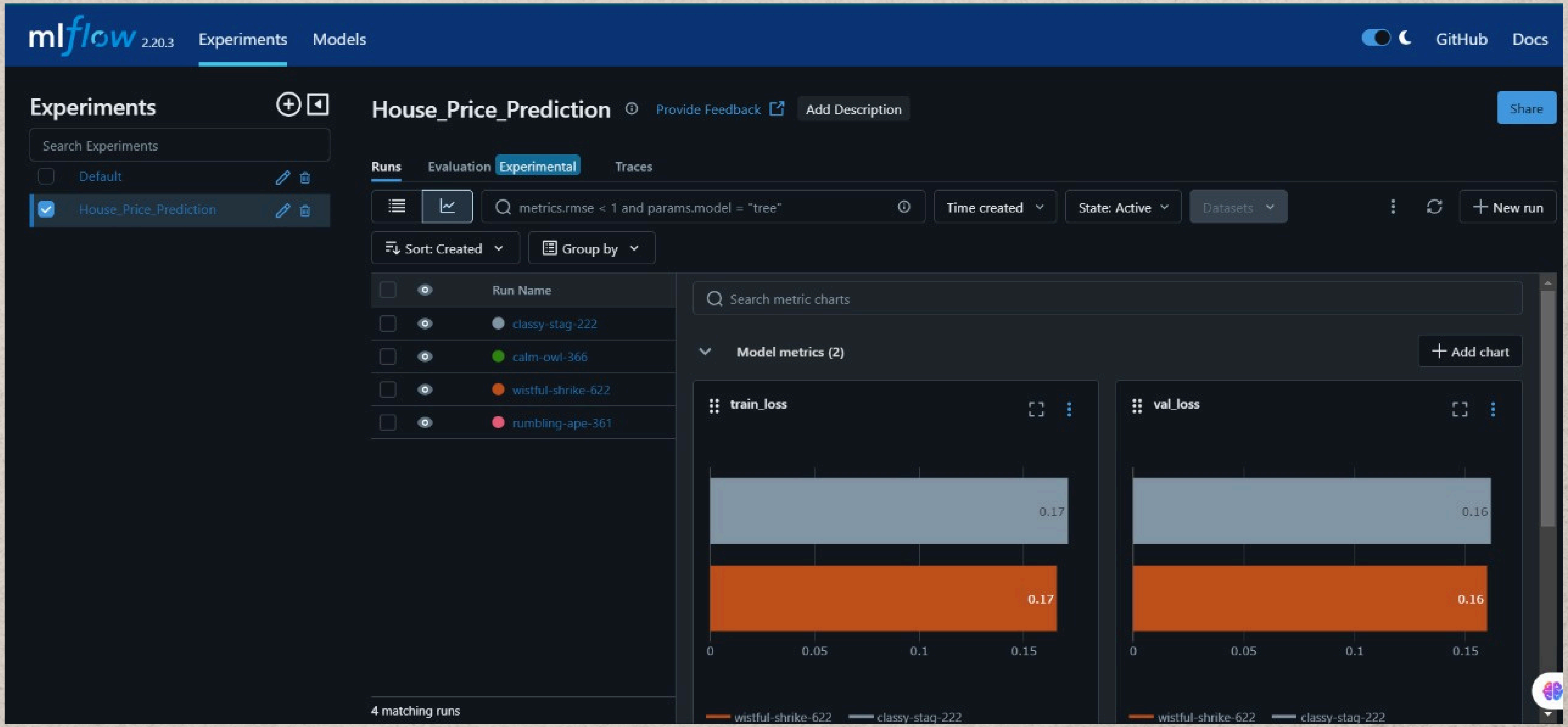
RESULTS

- **Model Comparison & Justification**

	Neural Network	Random Forest	XGBoost
MAE	0.28	0.38	0.7092
RMSE	0.23	0.34	0.7648
R-sqrd	0.24	0.33	0.7745

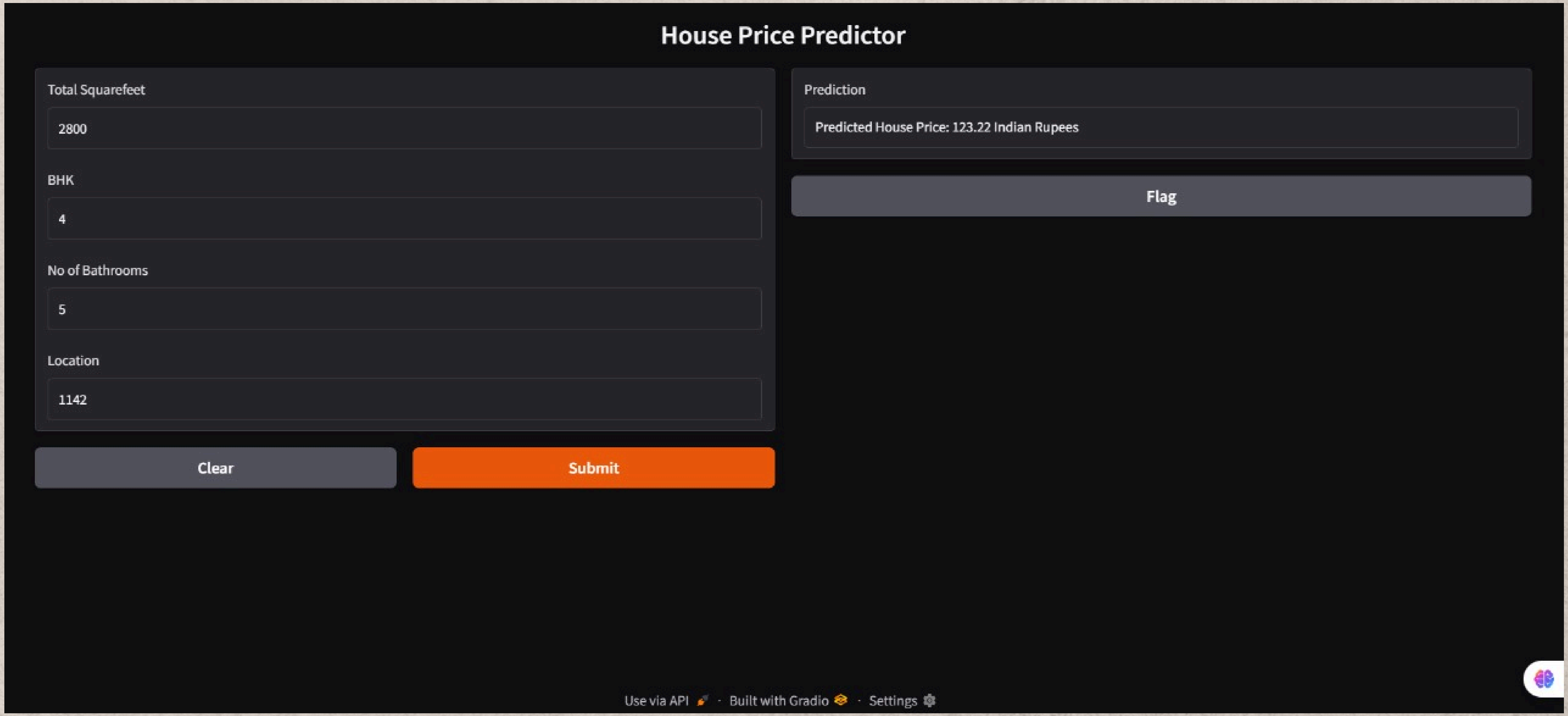
- Used Neural Network, Random Forest, and XGBoost to compare deep learning.
- Neural Network was chosen for its ability to learn complex relationships and scalability for larger datasets.

ML FLOW IN OUR PROJECT



- Tracking Experiment
- Model Registry
- Model Loading
- Deployment

UI FOR THE NEURAL NETWORK DEPLOYMENT



CHALLENGES ENCOUTERED

- Initial Models Got Overfitted : Model performed well on training data but poorly on test data.
- Computational Cost: The process of training multiple neural networks to identify the optimal model is computationally expensive and time-consuming.
- The **Location** feature contained string values. Encoded location names into numerical values to ensure model compatibility.

FUTURE IMPROVEMENTS

- Develop a more comprehensive dataset that includes real estate data relevant to Sri Lanka.
- Introduce new meaningful features to improve prediction accuracy
- Test deeper neural network architectures to enhance model performance
- Convert the trained model into a real-time web or mobile application for user-friendly predictions.

Thank You.