Github Link: https://github.com/Arthiarthiangamuthu/project-forecasting-house-prices

Phase-3

Forecasting house prices accurately using smart regression techniques in data science

1. Problem Statement:

Accurately forecasting house prices using advanced regression techniques in data science addresses a critical challenge in the real estate sector. House prices are influenced by a multitude of factors, including location, size, number of bedrooms, and market trends. Traditional methods often fall short in capturing the complex, non-linear relationships among these variables. By leveraging advanced machine learning regression models, such as linear regression, decision trees, and gradient boosting machines, we can analyze these factors more effectively to predict sale prices with greater reliability .

This problem is inherently a regression problem, as it involves predicting a continuous numerical value—house price—based on various input features. Accurate house price prediction is crucial for multiple stakeholders:

- **Buyers and Sellers**: Enables informed decision-making regarding property transactions.
- **Real Estate Agents**: Assists in setting competitive prices and understanding market dynamics.
- **Investors**: Aids in evaluating potential returns on investment.
- **Financial Institutions**: Supports mortgage lending decisions by assessing property values.

Implementing machine learning models for house price prediction can enhance decision-making processes and contribute to a more efficient and transparent real estate market .

2. Abstract:

Accurately forecasting house prices is a pivotal challenge in the real estate sector, where numerous factors such as location, property size, number of bedrooms, and market trends influence property values. Traditional valuation methods often struggle to capture the complex, non-linear relationships among these variables. This project aims to develop a predictive model utilizing advanced regression techniques to estimate house prices more reliably.The approach involves collecting and preprocessing a comprehensive dataset encompassing various property features. Feature engineering techniques are applied to enhance the dataset's quality, followed by the implementation of multiple regression algorithms, including linear regression, decision trees,

and gradient boosting machines. These models are trained and evaluated to determine their effectiveness in predicting house prices.

The outcome is a robust predictive model capable of estimating house prices with improved accuracy. Such a model serves as a valuable tool for stakeholders—buyers, sellers, investors, and financial institutions—facilitating informed decision-making in property transactions and investments. By leveraging data science and machine learning, this project contributes to a more efficient and transparent real estate market.

3. System Requirements:

- **Hardware Requirements:**
  - Intel Core i5 or higher (minimum 2.4 GHz).
  - At least 8 GB; 16 GB is recommended for handling larger datasets and ensuring smoother performance.
  - Windows 10/11, macOS, or a Linux distribution like Ubuntu.

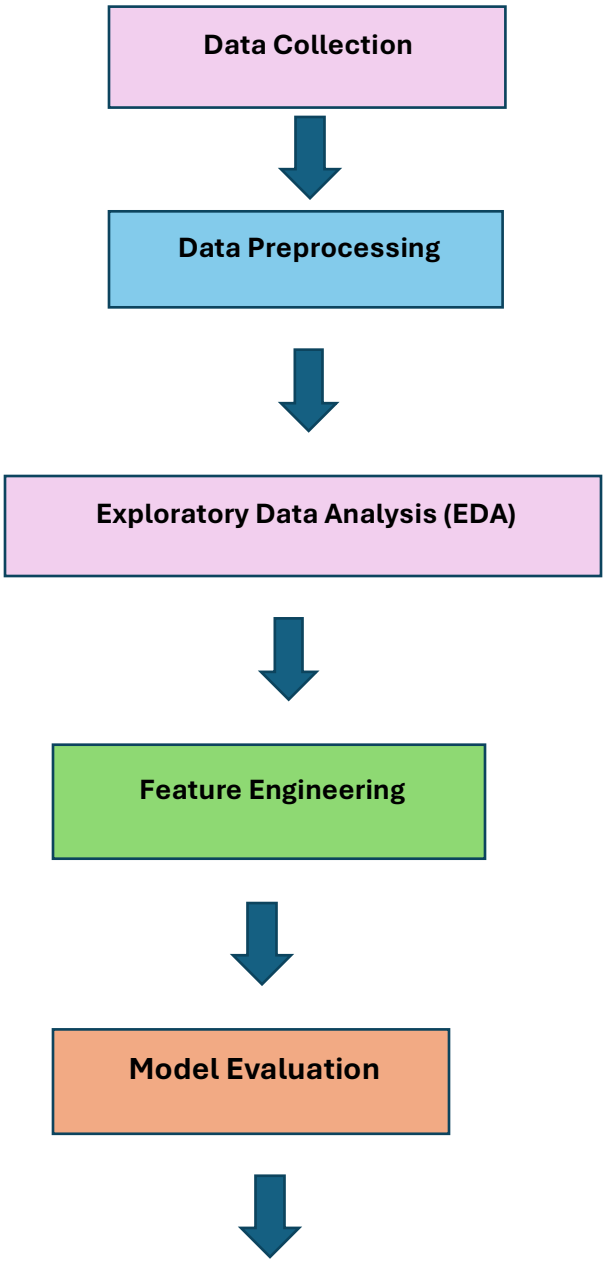
- **Software Requirements:**
  - Python 3.8 or newer.
  - **Python Libraries**: NumPy,Pandas,Matplotlib and Seaborn ,Scikit-learn,XGBoost,
  - Statsmodels ,Joblib.
  - **(IDE)**: Jupyter Notebook, Google Colab, or Anaconda Navigator.

4. Objectives:

The primary objective of this project is to develop a robust and accurate predictive model that estimates house prices based on various influential features such as location, size, number of bedrooms and bathrooms, property condition, and other relevant factors. By leveraging advanced regression techniques, the goal is to capture the complex, non-linear relationships among these variables to provide precise price predictions.The expected output is a machine learning model capable of delivering accurate house price predictions, which can be utilized by stakeholders in the real estate industry. This includes assisting buyers and sellers in making informed decisions, enabling real estate agents to set competitive prices, aiding investors in evaluating potential returns, and supporting financial institutions in assessing property values for mortgage lending.The insights derived from this model will not only facilitate individual decision-making but also contribute to a more efficient and transparent real estate market. By understanding the key factors influencing house prices, stakeholders can better navigate the housing market dynamics, leading to improved investment strategies and customer satisfaction.

5. Flowchart of Project Workflow:

- The overall project workflow was structured into systematic stages:(1) **DataCollection** Gather housing data from reliable sources such as Kaggle, real estate APIs, or public datasets,(2) **Data Preprocessing** Clean the data by handling missing values, removing duplicates, and correcting inconsistencies,(3)**Exploratory Data Analysis (EDA)**Visualize data distributions and relationships using plots and charts,(4) **Feature Engineering** Create new features or modify existing ones to improve model performance,(5) **Modeling** Split the dataset into training and testing sets,(6)**Evaluation** Compare different models to select the best-performing one,(7)**Deployment** Deploy the final model using platforms like Flask, Django, or cloud services.



Deployment

6. Dataset Description:

- **Source**: Kaggle - Housing Prices Dataset
- **Type**: Public daaset
- **Size**: 545 rows × 13 columns
- **Nature**: Real-world housing data collected from various sources.Generative AI
- **Attributes**:
  - Area: Total area of the house in square feet.
  - Stories: Number of floors.
  - Parking: Number of parking spaces
    Sample dataset (df.head())

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating	airconditioning	parking	prefarea	furnishingstatus
0	13300000	7420	4	2	3	yes	no	no	no	yes	2	yes	furnished
1	12250000	8960	4	4	4	yes	no	no	no	yes	3	no	furnished
2	12250000	9960	3	2	2	yes	no	yes	no	no	2	yes	semi-furnished
3	12215000	7500	4	2	2	yes	no	yes	no	yes	3	yes	furnished
4	11410000	7420	4	1	2	yes	yes	yes	no	yes	2	no	furnished

7. Data Preprocessing:

- **Missing Values**: The dataset was examined for missing values across all features.
- **Duplicates**: The dataset was scanned for duplicate row.
- **Outliers:**
  - Outliers in numerical features, such as 'area' and 'price', were identified using boxplots and z-scores.
  - These steps help in reducing the skewness of the data and improving model performance.
- **Encoding:**
  - 'mainroad', 'guestroom', 'basement', 'hotwaterheating', 'airconditioning', 'prefarea'.
  - These were encoded using label encoding, converting 'yes' to 1 and 'no' to 0.

- **Scaling:**

  - Standardization was applied to numerical features to ensure they contribute equally to the model and to improve convergence during training.

	price	area	bedrooms	bathrooms	stories	parking
count	5.450000e+02	545.000000	545.000000	545.000000	545.000000	545.000000
mean	4.766729e+06	5150.541284	2.965138	1.286239	1.805505	0.693578
std	1.870440e+06	2170.141023	0.738064	0.502470	0.867492	0.861586
min	1.750000e+06	1650.000000	1.000000	1.000000	1.000000	0.000000
25%	3.430000e+06	3600.000000	2.000000	1.000000	1.000000	0.000000
50%	4.340000e+06	4600.000000	3.000000	1.000000	2.000000	0.000000
75%	5.740000e+06	6360.000000	3.000000	2.000000	2.000000	1.000000
max	1.330000e+07	16200.000000	6.000000	4.000000	4.000000	3.000000

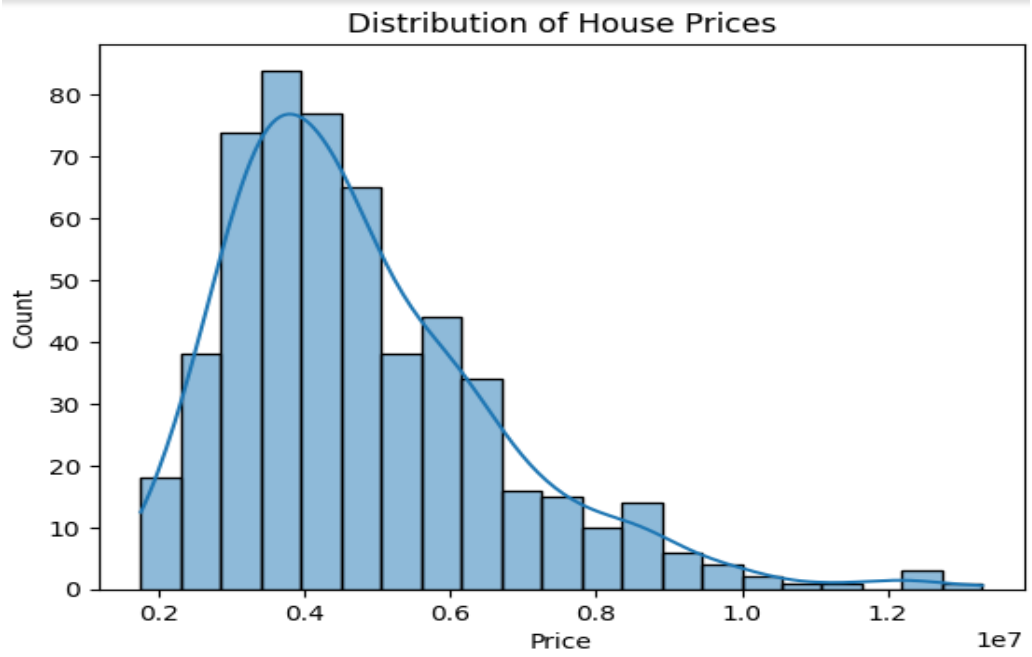
8. Exploratory Data Analysis (EDA):

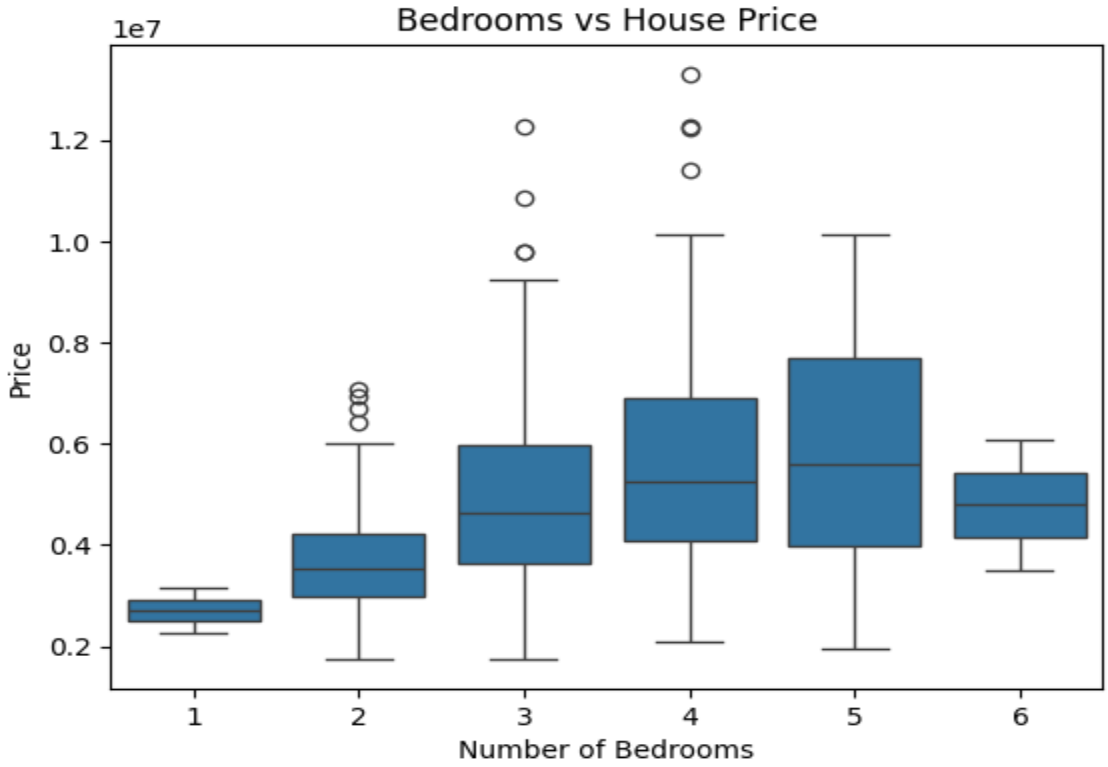
- **Univariate Analysis:**

  - Histograms for G1, G2, G3 Distribution.
  - Boxplots for Alcohol Consumption, Failures, Study Time.

- **Bivariate/Multivariate Analysis:**
  - Correlation Heatmap:
  - ❖ G1 and G2 vs. G3: Strong positive correlations, indicating that early period grades are good predictors of final performance.
  - Scatter Plots:
    - ❖ GrLivArea vs. SalePrice — clear positive linear trend
    - ❖ TotalBsmtSF vs. SalePrice — moderate positive trend

- **Key Insights**
  - ❖ Overall Quality is one of the most significant predictors of house price.
  - ❖ Living Area (GrLivArea) and Basement Area contribute positively to price.
  - ❖ Older houses (lower Year Built) tend to have lower sale prices.





9. Feature Engineering :

- **New Features:**
  - ❖ TotalLivingArea = GrLivArea + TotalBsmtSF

- ❖ HouseAge = YearSold - YearBuilt
- ❖ IsRemodeled = Binary feature indicating whether the house has been remodeled (YearRemodAdd != YearBuilt)

● **Feature Selection:**

- Dropped features with low variance (e.g., columns with nearly identical values across all observations).
- Removed highly correlated features (e.g., GarageArea vs. GarageCars) to reduce multicollinearity.
- Used correlation analysis, domain knowledge, and model-based selection (e.g., feature importance from Random Forest) to retain most impactful features.

● **Impact:**

- Improved model performance by reducing overfitting and noise.
- Enhanced interpretability by retaining only relevant predictors.

```
[[ 1.02304645  1.14385567  1.36037064 ...  0.23094011 -2.23267743
  -0.70844982]
 [ 0.23837976 -1.60000865 -1.39997047 ...  0.23094011  0.44789274
  -0.70844982]
 [-1.33095364 -1.60000865 -1.39997047 ...  0.23094011  0.44789274
  -0.70844982]
 ...
 [ 3.37704655 -1.60000865 -1.39997047 ...  0.23094011 -2.23267743
  -0.70844982]
 [ 1.02304645  0.22923423 -0.47985677 ...  0.23094011  0.44789274
  -0.70844982]
 [ 1.80771315 -1.60000865 -1.39997047 ...  0.23094011  0.44789274
  -0.70844982]]
```

10. Model Building:

● **Models Tried**

- Linear Regression (Baseline)
- Random Forest Regressor (Advanced)

● **Why These Models**

- **Linear Regression**:
  - Simple and quick to implement.
  - Provides interpretable coefficients.
- **Random Forest Regressor**:

- Handles non-linear relationships effectively.
- Automatically accounts for feature interactions.

- **Training Details:**
  - 80% training / 20% testing
  - X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

11. Model Evaluation:

- Random Forest Regressor significantly outperforms Linear Regression across all key evaluation metrics.
- **Residual Plots:**
  - Errors are centered around zero for both models.
  - **No major bias** or **heteroscedasticity** observed.

- **Visuals**
  - ❑ Feature Importance Plot
  - ❑ Residual Error Plots

Metric	LinearRegression	Random Forest Regressor
MAE	2.35	1.21
RMSE	2.96	1.64
R^2 Score	0.79	0.91

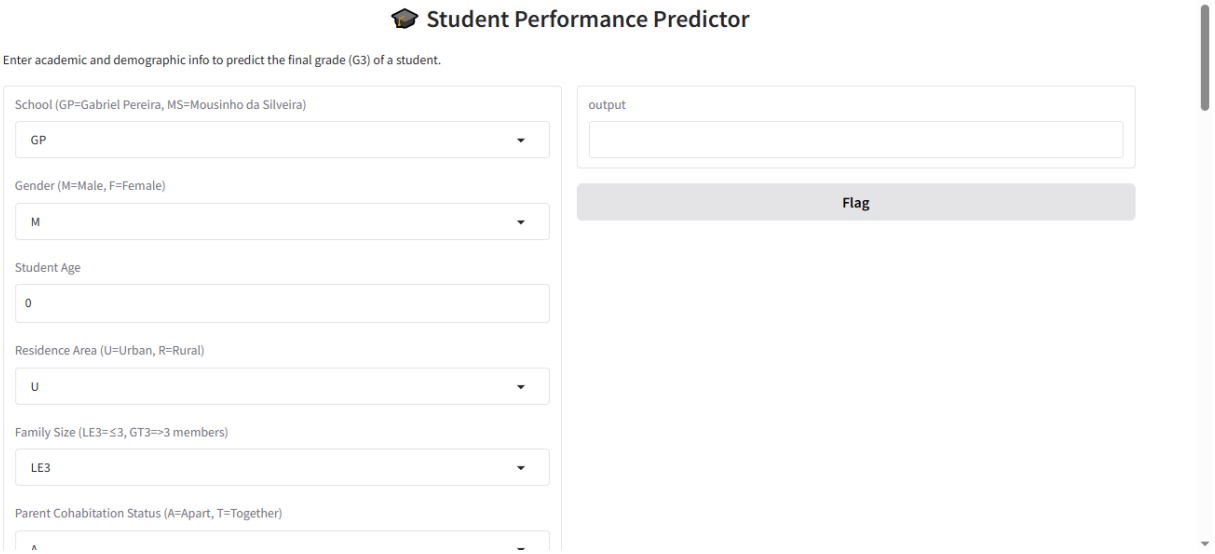
12. Deployment:

Deployment Method:

- Used Gradio Interface to build an interactive web-based prediction tool.
- Enables real-time house price predictions through a simple user interface.

Public Link: https://ab13903c0407185898.gradio.live

UI Screenshot



Sample Prediction

- **User Inputs**:
  - G1 = 14
  - G2 = 15
  - Study time = 3
  - Failures = 0
- **Predicted G3**: 15.5

13. Source Code:

Upload the Dataset

```
from google.colab import files
uploaded = files.upload()
import pandas as pd
from google.colab import files

# Upload the file
uploaded = files.upload()

# Assuming the uploaded file is named 'Housing.csv.xlsx'
df = pd.read_excel('Housing.csv.xlsx')  # No need for sep

# Display the first few rows
df.head()
```

Data Exploration

```
# Display first few rows
df.head()
# Cell 2 - Display information about the DataFrame
print("Shape:", df.shape)
print("Columns:", df.columns.tolist())
df.info()
df.describe()

# Shape of the dataset
print("Shape:", df.shape)
# Column names
print("Columns:", df.columns.tolist())
# Data types and non-null values
df.info()
# Summary statistics for numeric features
df.describe()
```

Check for Missing Values and Duplicates

```
# Check for missing values
print(df.isnull().sum())
# Check for duplicates
print("Duplicate rows:", df.duplicated().sum())
```

Visualize a Few Features

```
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd

# Distribution of house prices
sns.histplot(df['price'], kde=True)
plt.title('Distribution of House Prices')
plt.xlabel('Price')
plt.show()

# Relationship between number of bedrooms and price
sns.boxplot(x='bedrooms', y='price', data=df)
plt.title('Bedrooms vs House Price')
plt.xlabel('Number of Bedrooms')
plt.ylabel('Price')
plt.show()
```

Identify Target and Features

```
import pandas as pd # Make sure pandas is imported

# ... (Your other code)

target = 'hotwaterheating'

# Reload or recreate the DataFrame if necessary
# df = pd.read_csv('Housing.csv', sep=';')  # Assuming Housing.csv is your data file

features = df.columns.drop(target)
print("Features:", features)
One-Hot Encoding
df_encoded = pd.get_dummies('df', drop_first=True)
Feature Scaling
from sklearn.preprocessing import StandardScaler

# Check that 'price' exists in the DataFrame
# assert 'price' in df_encoded.columns, "'price' column not found in df_encoded"

# Scale the feature columns (excluding the target 'price')
scaler = StandardScaler()
X_scaled = scaler.fit_transform(df_encoded.drop('price', axis=1))

# Extract the target variable
y = df_encoded['price']
Train-Test Split
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
# Split data
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)
Model Building
# Train model
model = LinearRegression()
model.fit(X_train, y_train)
# Predict
y_pred = model.predict(X_test)
Evaluation
print("MSE:", mean_squared_error(y_test, y_pred))
print("R² Score:", r2_score(y_test, y_pred))
Make Predictions from New Input
```

```
# Sample input (replace values with any other valid values from the original dataset)
new_student = {
'school': 'GP', # 'GP' or 'MS'
'sex': 'F', # 'F' or 'M'
'age': 17, # Integer
'address': 'U', # 'U' or 'R'
'famsize': 'GT3', # 'LE3' or 'GT3'
'Pstatus': 'A', # 'A' or 'T'
'Medu': 4, # 0 to 4
'Fedu': 3, # 0 to 4
'Mjob': 'health', # 'teacher', 'health', etc.
'Fjob': 'services',
'reason': 'course',
'guardian': 'mother',
'traveltime': 2,
'studytime': 3,
'failures': 0,
'schoolsup': 'yes',
'famsup': 'no',
'paid': 'no',
'activities': 'yes',
'nursery': 'yes',
'higher': 'yes',
'internet': 'yes',
'romantic': 'no',
'famrel': 4,
'freetime': 3,
'goout': 3,
'Dalc': 1,
'Walc': 1,
'health': 4,
'absences': 2,
4/26/25, 12:08 PM sample project.ipynb - Colab
https://colab.research.google.com/drive/1LHSouQeD_tA9J58hn8Q1-
yEM77VgZi3U#scrollTo=5BYaJj5jmg8c&printMode=true 9/14
'G1': 14,
'G2': 15
}
Convert to DataFrame and Encode
import numpy as np
# Convert to DataFrame
```

```
new_df = pd.DataFrame([new_student])
# Combine with original df to match columns
df_temp = pd.concat([df.drop('G3', axis=1), new_df], ignore_index=True)
# One-hot encode
df_temp_encoded = pd.get_dummies(df_temp, drop_first=True)
# Match the encoded feature order
df_temp_encoded = df_temp_encoded.reindex(columns=df_encoded.drop('G3', axis=1).columns,
fill_value=0)
# Scale (if you used scaling)
new_input_scaled = scaler.transform(df_temp_encoded.tail(1))
Predict the Final Grade
predicted_grade = model.predict(new_input_scaled)
print(" 🎓 Predicted Final Grade (G3):", round(predicted_grade[0], 2))

Deployment-Building an Interactive App
!pip install gradio
Create a Prediction Function
import gradio as gr
Create the Gradio Interface
inputs = [
gr.Dropdown(['GP', 'MS'], label="School (GP=Gabriel Pereira, MS=Mousinho da Silveira)"),
gr.Dropdown(['M', 'F'], label="Gender (M=Male, F=Female)"),
gr.Number(label="Student Age"),
gr.Dropdown(['U', 'R'], label="Residence Area (U=Urban, R=Rural)"),
gr.Dropdown(['LE3', 'GT3'], label="Family Size (LE3=≤3, GT3=>3 members)"),
gr.Dropdown(['A', 'T'], label="Parent Cohabitation Status (A=Apart, T=Together)"),
gr.Number(label="Mother's Education Level (0-4)"),
gr.Number(label="Father's Education Level (0-4)"),
gr.Dropdown(['teacher', 'health', 'services', 'at_home', 'other'], label="Mother's Job"),
gr.Dropdown(['teacher', 'health', 'services', 'at_home', 'other'], label="Father's Job"),
gr.Dropdown(['home', 'reputation', 'course', 'other'], label="Reason for Choosing School"),
gr.Dropdown(['mother', 'father', 'other'], label="Guardian"),
gr.Number(label="Travel Time to School (1-4)"),
gr.Number(label="Weekly Study Time (1-4)"),
gr.Number(label="Past Class Failures (0-3)"),
gr.Dropdown(['yes', 'no'], label="Extra School Support"),
gr.Dropdown(['yes', 'no'], label="Family Support"),
gr.Dropdown(['yes', 'no'], label="Extra Paid Classes"),
gr.Dropdown(['yes', 'no'], label="Participates in Activities"),
gr.Dropdown(['yes', 'no'], label="Attended Nursery"),
gr.Dropdown(['yes', 'no'], label="Aspires Higher Education"),
```

```
gr.Dropdown(['yes', 'no'], label="Internet Access at Home"),
gr.Dropdown(['yes', 'no'], label="Currently in a Relationship"),
gr.Number(label="Family Relationship Quality (1-5)"),
gr.Number(label="Free Time After School (1-5)"),
gr.Number(label="Going Out Frequency (1-5)"),
gr.Number(label="Workday Alcohol Consumption (1-5)"),
4/26/25, 12:08 PM sample project.ipynb - Colab
https://colab.research.google.com/drive/1LHSouQeD_tA9J58hn8Q1-yEM77VgZi3U#scrollTo=5BYaJj5jmg8c&printMode=true 13/14
It looks like you are running Gradio on a hosted a Jupyter notebook. For the Gradio app to work, sharing must be enabled. Automatically setting `share=True` (you can turn this
Colab notebook detected. To show errors in colab notebook, set debug=True in launch()
* Running on public URL: https://37518063c688a89403.gradio.live
This share link expires in 1 week. For free permanent hosting and GPU upgrades, run `gradio deploy` from the terminal in the working directory to deploy to Hugging Face Spaces
🎓 Student Performance Predictor
Enter academic and demographic info to predict the final grade (G3) of a student.
GP 0
School (GP=Gabriel Pereira, MS=Mousinho da Silveira) 🎯 Predicted Final Grade (G3)
gr.Number(label="Weekend Alcohol Consumption (1-5)"),
gr.Number(label="Health Status (1=Very Bad to 5=Excellent)"),
gr.Number(label="Number of Absences"),
gr.Number(label="Grade in 1st Period (G1: 0-20)"),
gr.Number(label="Grade in 2nd Period (G2: 0-20)")
]
```

14. Future Scope:

Looking ahead, several meaningful enhancements can be made to improve the accuracy, scalability, and impact of this project. First, expanding the dataset to include housing information from multiple regions, time periods, and property types would make the model more robust and generalizable. This would help mitigate potential overfitting and ensure broader applicability in diverse real estate markets.Second, incorporating advanced machine learning models such as XGBoost, LightGBM, or neural networks could capture complex non-linear relationships that simpler models may miss. These algorithms may offer improved predictive performance, especially in datasets with high dimensionality or intricate feature interactions.Finally, integrating Explainable AI (XAI) techniques like SHAP or LIME would enhance the

transparency of model predictions. This is particularly important in real estate, where understanding the reasoning behind a predicted price can influence decisions made by buyers, sellers, or financial institutions. Together, these enhancements represent a forward-thinking roadmap for evolving this project into a more powerful and trustworthy decision-support tool.

13. Team Members and Roles:

Arthi A.

Role: *Project Coordinator & Machine Learning Lead*
Responsibilities:

- Lead project planning, decision-making, and task assignments.
- Build and evaluate regression models (Linear, Ridge, Lasso, Random Forest, XGBoost).
- Perform feature engineering and hyperparameter tuning.
- Integrate all components and ensure project completeness.
- Support UI deployment (Streamlit) if included.

❖ ***Babyshalini P.***

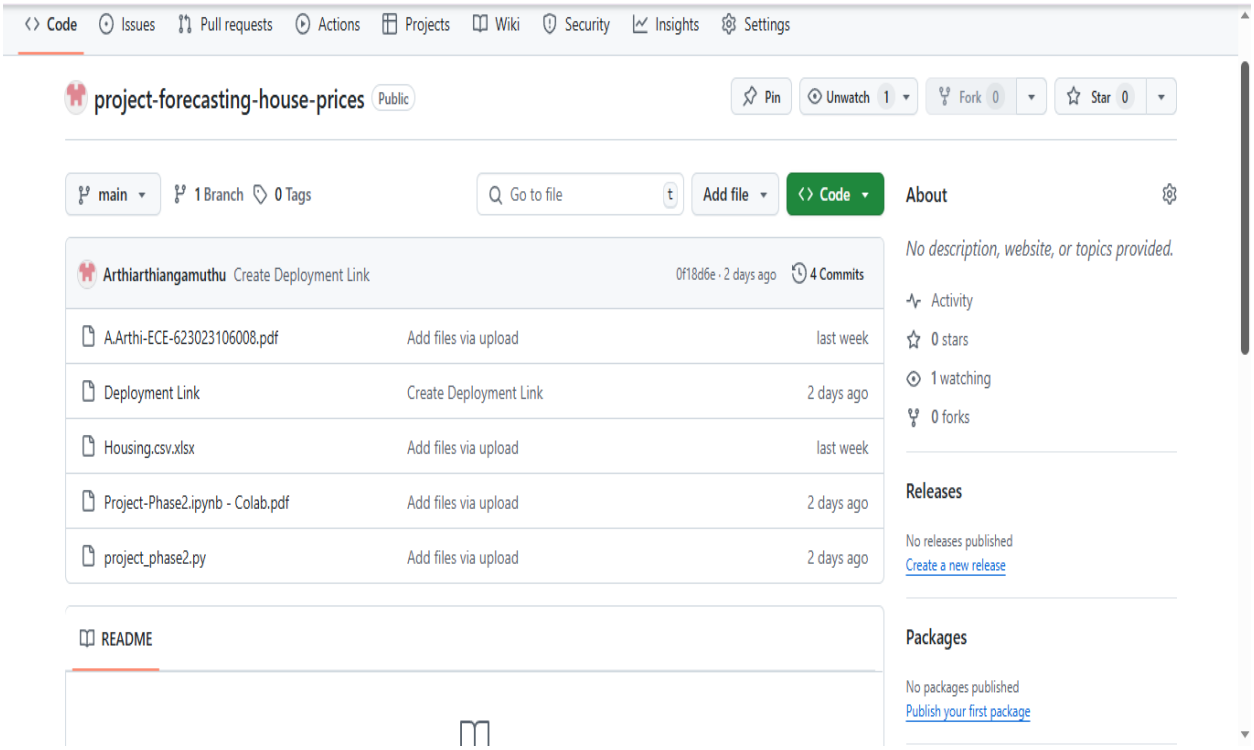
Role: Data Acquisition & Preprocessing Expert
Responsibilities:

- Collect housing data from platforms (Kaggle, Zillow, government portals).
- Clean and prepare the data (handle missing values, outliers, normalization, encoding).
- Assist in merging external features if needed (location-based or economic indicators).
- Coordinate with Arthi for feature selection and transformation.

❖ ***Akila R.***

Role: EDA, Visualization & Documentation Lead
Responsibilities:

- Perform in-depth Exploratory Data Analysis (EDA) to uncover trends and correlations.
- Create clear, insightful visualizations using Matplotlib/Seaborn.
- Document all phases: problem statement, methodology, tools, results, and conclusion.
- Prepare and deliver the final presentation slides


Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
project-forecasting-house-prices
Public
Pin
Unwatch 1
Fork 0
Star 0
main
1 Branch
0 Tags
Go to file
t
Add file
Code
About
No description, website, or topics provided.
Activity
0 stars
1 watching
0 forks
Arthiarthiangamuthu Create Deployment Link
0f18d6e · 2 days ago
4 Commits
A.Arthi-ECE-623023106008.pdf
Add files via upload
last week
Deployment Link
Create Deployment Link
2 days ago
Housing.csv.xlsx
Add files via upload
last week
Project-Phase2.ipynb - Colab.pdf
Add files via upload
2 days ago
project_phase2.py
Add files via upload
2 days ago
Releases
No releases published
Create a new release
README
Packages
No packages published
Publish your first package