# Internship Project Report

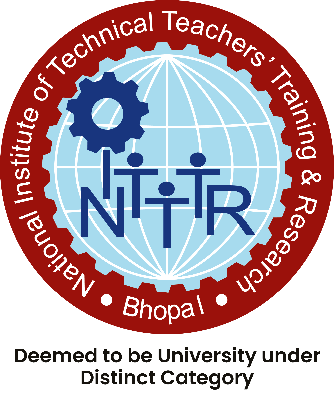
Submitted in partial fulfillment of the requirement for the award of a certificate of internship program

## in

**National Institute of Technical Teachers' Training and Research, Bhopal**

**Deemed to be University (under distinct category)**

**Submitted to**



**Submitted by**

**VIJAY LAXMI**

**Under the Supervision of**

**DR. GANAPATHYSANNASI**

**COMPUTER SCIENCE**

**June 2025**

**National Institute of Technical Teachers' Training & Research, Bhopal**

**Deemed to be University (Under Distinct Category)**

**Ministry of Education, Govt. of India**

Shyamala Hills, Bhopal - 462 002 (M.P.), India

**National Institute of Technical Teachers' Training and Research, Bhopal**

**Deemed to be University (under distinct category)**

**CERTIFICATE**

This is to certify that the work embodies in this dissertation entitled **Artificial and Machine Learning** being submitted by **Vijay Laxmi** for partial fulfillment of the requirement for the award of **Certificate** during the short term internship program is a record of a Bonafede piece of work, which was carried out by him under my supervision and guidance in the Department of Computer Science of **National Institute of Technical Teachers' Training and Research, Bhopal**

**Head Of Department Supervisor**

**Dr. Murtaza Abbas Rizvi Dr.Ganapathy Sannasi**

**National Institute of Technical Teachers' Training and Research, Bhopal**

**Deemed to be University (under distinct category)**

**CANDIDATE’S DECLARATION**

.

I **Vijay Laxmi** hereby declare that the report entitled AI/ML which is being submitted to **Computer Science and Engineering Education** in **National Institute of Technical Teachers' Training and Research, Bhopal**, is our authentic work carried out during internship.

### I declare that my work has not been submitted in part or in full to any other university or institution for the award of any certificate.

**VIJAYLAXMI**

# National Institute of Technical Teachers' Training and Research, Bhopal

**Deemed to be University (under distinct category)**

## ACKNOWLEDGEMENT

I would like to express my heartfelt gratitude to everyone who supported me throughout the completion of this project.

First and foremost, I thank the Almighty for giving me the strength and knowledge to complete this work.

I am deeply thankful to my mentor and teacher, **Dr. Ganapathy Sannasi**, for his valuable guidance, support, and encouragement throughout the project. His clear teaching in Artificial Intelligence and Machine Learning made it easy for me to understand complex concepts and apply them practically.

I also extend my thanks to all the faculty members of the Computer Science and Engineering for providing a supportive environment and the necessary resources.

Lastly, I am thankful to my family and friends for their motivation, patience, and continuous encouragement during this journey.

This project would not have been possible without the help and support of all these wonderful people.

**Vijay Laxmi Rajput**  
**ID: C2K231332**   
**Department of Computer Science**  
**Pune Institute of Computer Technology, Pune**

**National Institute of Technical Teachers' Training and Research,**

**Bhopal**

**Deemed to be University (under distinct category)**

**INTERNSHIP PROJECT REPORT**

**Executive Summary**

This report presents a comprehensive study on predicting house prices in Melbourne using machine learning algorithms. The primary objective of this project is to analyze housing data and develop predictive models that can accurately estimate property prices based on various features such as location, number of rooms, land size, and other relevant attributes.

### ****Objectives:****

* To clean and preprocess the Melbourne housing dataset for modeling.
* To explore and visualize the data to understand key influencing factors.
* To implement and evaluate multiple machine learning models for predicting house prices.

### ****Key Activities:****

1. **Data Collection & Cleaning:**  
   The dataset was obtained from publicly available sources. Missing values, outliers, and irrelevant columns were handled to ensure quality inputs for modeling.
2. **Data Visualization:**  
   Several charts, including distribution plots and correlation heatmaps, were used to understand the data and identify strong predictors of price.
3. **Feature Engineering & Preprocessing:**  
   Techniques such as encoding categorical variables, normalizing numerical features, and handling null values were applied to prepare the dataset for modeling.
4. **Model Implementation:**  
   Four models were implemented:
   * **Linear Regression:** for continuous price prediction.
   * **Logistic Regression:** for categorizing prices into ‘Low’, ‘Medium’, and ‘High’.
   * **Random Forest:** to capture non-linear relationships and improve accuracy.
   * **LSTM (Long Short-Term Memory):** a deep learning approach used to model sequential dependencies in the data.
5. **Evaluation & Comparison:**  
   Each model was evaluated using metrics such as MAE, RMSE, R² Score, and classification accuracy (for logistic regression). Among the models, Random Forest and LSTM performed notably well in capturing complex patterns in the data.

**FINDINGS:**

* 1. **Random Forest** achieved high accuracy due to its ability to model complex relationships and handle overfitting.
  2. **LSTM** demonstrated strong performance, especially when temporal or sequential aspects of the data were considered.
  3. **Linear Regression** provided a simple and interpretable baseline.
  4. **Logistic Regression** was useful in classifying price ranges rather than predicting exact values.

This analysis proves that machine learning can significantly assist in real estate pricing strategies by providing accurate, data-driven price estimations. Further improvements can be made by integrating external data such as interest rates, economic indicators, and urban planning changes.

# National Institute of Technical Teachers' Training and Research, Bhopal

**Deemed to be University (under distinct category)**

**INTERNSHIP PROJECT REPORT**

**Table of Content**

|  |  |  |
| --- | --- | --- |
| **Chapter** | **Content** | **Page No.** |
| **1.** | **Abstract** | 8 |
| **2.** | **Introduction** | 8 |
| **3.** | Literature Surveya. Data Preprocessing and Data Cleaning & Data Visualization **b. Feature Scaling & Data Visualization** | 9 |
| **4.** | Results and Discussion | 13 |
| **5.** | Conclusion | 14 |
| **6.** | **References** | 14 |

Melbourne House Price Prediction using Machine Learning Algorithms

# Abstract

The rising need for accurate and real-time house price estimation has driven the integration of machine learning techniques into the real estate sector. This project focuses on building a predictive model for estimating property prices in Melbourne using advanced machine learning algorithms, including Linear Regression, Random Forest, Logistic Regression, and Long Short-Term Memory (LSTM) networks.

The dataset, titled Melbourne House Prices Less, includes detailed historical data with features such as location, number of rooms, property type, land size, building area, year built, and distance from the city center. These features play a crucial role in understanding market trends and influencing property value.

A robust preprocessing pipeline is applied, including data cleaning, handling null values, outlier detection, label encoding, scaling, and feature engineering. Exploratory Data Analysis (EDA) using visualization tools like heatmaps, box plots, and correlation matrices helps uncover relationships among variables and guide feature selection.

The modeling approach addresses both regression and classification tasks. Regression models are evaluated using Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and R² Score, while classification models assess affordability using Accuracy, Precision, Recall, and F1 Score. The LSTM model leverages temporal patterns in price trends for improved performance.

This end-to-end machine learning solution enables precise property price estimation and affordability categorization, supporting buyers, investors, and real estate professionals in making data-driven, intelligent decisions.

# Introduction

Machine Learning (ML) enables systems to learn patterns from data and make data-driven predictions or decisions without being explicitly programmed. In the real estate domain, ML can be used to forecast housing prices by analyzing historical transaction records and identifying key influencing factors such as location, property features, and market trends.

This project applies a variety of ML techniques to the Melbourne House Prices Less dataset to estimate property values and classify them into affordability categories. For regression-based price prediction, Linear Regression and Random Forest Regressor are implemented, offering a comparison between a simple linear model and a powerful ensemble method. For classification, Logistic Regression is used to categorize houses into predefined price brackets based on engineered features.

In addition, LSTM (Long Short-Term Memory), a type of deep learning model suited for time-series or sequential data, is evaluated to capture temporal patterns and long-term dependencies in the housing market.

Extensive data preprocessing—including null value handling, label encoding for categorical variables, scaling of numerical features, and feature engineering—is conducted to prepare clean, meaningful input for the models. Evaluation is performed using metrics such as MAE, RMSE, and R² Score for regression, and Accuracy, Precision, Recall, and F1 Score for classification. This structured approach helps determine the most accurate and efficient algorithm for real-world deployment.

# Literature Survey

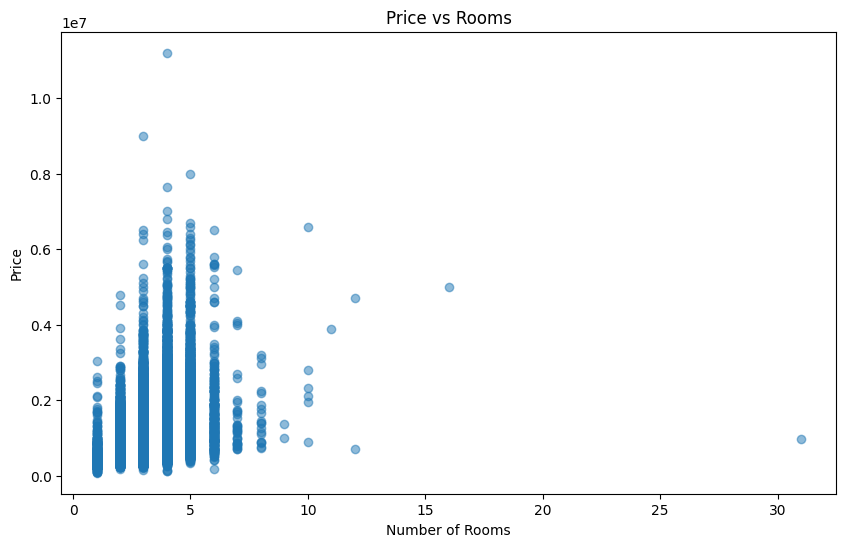
# Previous works on housing price prediction primarily used datasets like the Boston Housing dataset and applied models such as Linear Regression, Decision Trees, or Gradient Boosting.

# While these models showed moderate accuracy, they lacked adaptability for geographically diverse data. Our project builds upon such research but applies modern techniques like Random Forest and LSTM on a real-world, heterogeneous dataset from Melbourne, thus offering more robust and scalable predictions.

# a. Data Preprocessing and Data Cleaning & Data Visualization

# Drop irrelevant columns and handle missing values  
df.drop(['Address', 'Date', 'SellerG'], axis=1, errors='ignore', inplace=True)  
df.dropna(subset=['Price'], inplace=True)  
  
# Visualization - Distribution of Price  
sns.histplot(df['Price'], bins=50, kde=True)  
plt.title("Distribution of House Prices")  
plt.xlabel("Price")  
plt.ylabel("Frequency")  
plt.show()

**b. Feature Scaling & Data Visualization**

# Feature Scaling using StandardScaler  
scaler = StandardScaler()  
X\_scaled = scaler.fit\_transform(X)  
  
# Visualization - Price vs Distance  
sns.scatterplot(data=df, x='Distance', y='Price')  
plt.title("Price vs Distance from City")  
plt.show()

**SCATTERPLOT**

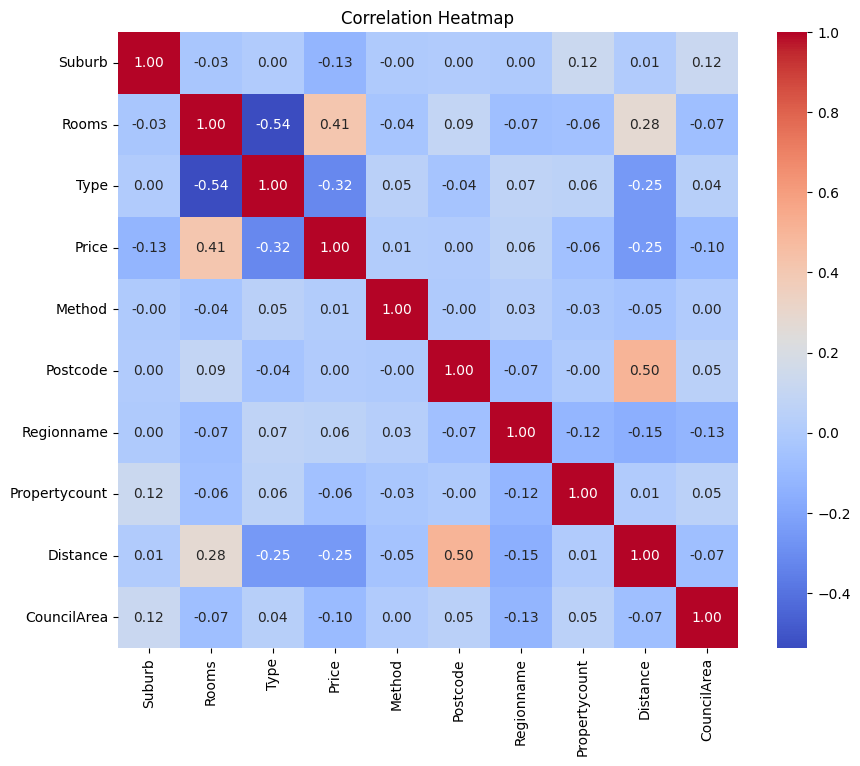
plt.figure(figsize=(10,8))

corr = df.corr()

sns.heatmap(corr, annot=True, cmap='coolwarm', fmt=".2f")

plt.title("Correlation Heatmap")

plt.show()



**HEAT MAP**

# prompt: pie chart

import matplotlib.pyplot as plt

# Pie Chart for Type Distribution

type\_counts = df['Type'].value\_counts()

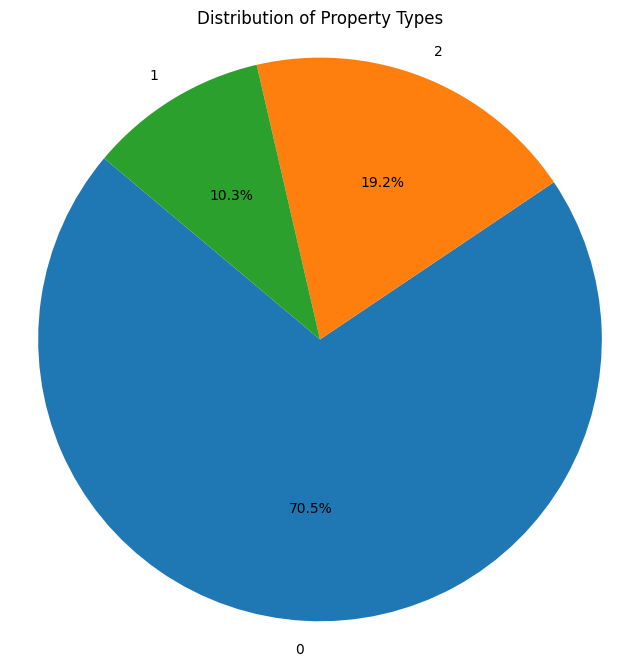
plt.figure(figsize=(8, 8))

plt.pie(type\_counts, labels=type\_counts.index, autopct='%1.1f%%', startangle=140)

plt.title("Distribution of Property Types")

plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.

plt.show()



**PIE CHART**

## 4.2 Prediction through Classification

* **Classifiers used: Random Forest**
* **Logistic Regression**
* **LSTM**

### ****Logistic Regression****

**Definition**:  
Logistic Regression is a **classification algorithm** used to predict the **probability** of a target variable belonging to a certain class, using the **logistic (sigmoid)** function.

**Sigmoid Function**:

P(y)=11+e−(b0+b1x1+...+bnxn)P(y) = \frac{1}{1 + e^{-(b\_0 + b\_1x\_1 + ... + b\_nx\_n)}}P(y)=1+e−(b0​+b1​x1​+...+bn​xn​)1​

**Working**:

* Outputs a **probability between 0 and 1**.
* Classifies based on a threshold (e.g., >0.5 = class 1, otherwise class 0).

**Advantages**:

* Good for **binary or multi-class classification**.
* Interpretable and **fast to train**.

**Limitations**:

* Not suitable for complex, non-linear relationships unless feature transformation is applied.
* Assumes a **log-odds linear relationship**.

**Application in House Price Prediction**:

* Used to **classify houses** into affordability categories:
  + Example: 0 = Affordable, 1 = Expensive
  + Useful for creating **price ranges** for different customer segments.

**Logistic Regression**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler, LabelEncoder

from sklearn.linear\_model import LinearRegression

from sklearn.ensemble import RandomForestRegressor

from sklearn.metrics import mean\_absolute\_error, mean\_squared\_error, r2\_score

df['Price\_Category'] = pd.qcut(df['Price'], q=3, labels=['Low', 'Medium', 'High'])

le\_price\_cat = LabelEncoder()

df['Price\_Category'] = le\_price\_cat.fit\_transform(df['Price\_Category'])

X = df.drop(['Price', 'Log\_Price', 'Price\_Category'], axis=1)

y = df['Price\_Category']

scaler = StandardScaler()

X\_scaled = scaler.fit\_transform(X)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_scaled, y, test\_size=0.2, random\_state=42)

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import accuracy\_score, classification\_report, confusion\_matrix

log\_clf = LogisticRegression(max\_iter=1000)

log\_clf.fit(X\_train, y\_train)

y\_pred\_log = log\_clf.predict(X\_test)

print("🎯 Logistic Regression Classification Report:")

print(classification\_report(y\_test, y\_pred\_log, target\_names=le\_price\_cat.classes\_))

print("📊 Confusion Matrix:")

print(confusion\_matrix(y\_test, y\_pred\_log))

print("✅ Accuracy:", accuracy\_score(y\_test, y\_pred\_log))

### ****. Random Forest****

**Definition**:  
Random Forest is an **ensemble learning method** that combines many decision trees to make predictions more robust, accurate, and less prone to overfitting.

**Working**:

* It builds **multiple decision trees** during training and outputs the **average** prediction for regression or **majority vote** for classification.
* Each tree is trained on a **random subset** of data and features (a process called "bagging").

**Advantages**:

* Handles both **categorical and numerical data**.
* **Reduces overfitting** seen in individual decision trees.
* Captures **non-linear relationships**.

**Limitations**:

* Can be **computationally intensive**.
* Less interpretable than linear models.

**Application in House Price Prediction**:

* Performs **better than linear models** when data is complex.
* Can be used for **both price prediction** and **price category classification**.

**Random Forest Regressor**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler, LabelEncoder

from sklearn.linear\_model import LinearRegression

from sklearn.ensemble import RandomForestRegressor

from sklearn.metrics import mean\_absolute\_error, mean\_squared\_error, r2\_score

df = pd.read\_csv("MELBOURNE\_HOUSE\_PRICES\_LESS.csv")

df = df.drop(['Address', 'Date', 'SellerG'], axis=1, errors='ignore')

df = df.dropna(subset=['Price'])

label\_encoders = {}

categorical\_cols = df.select\_dtypes(include='object').columns

for col in categorical\_cols:

    le = LabelEncoder()

    df[col] = le.fit\_transform(df[col])

    label\_encoders[col] = le

df['Price\_per\_room'] = df['Price'] / (df['Rooms'] + 1)

df['Log\_Price'] = np.log1p(df['Price'])

df['Distance\_squared'] = df['Distance'] \*\* 2

df['Price\_per\_km'] = df['Price'] / (df['Distance'] + 1)

X = df.drop(['Price', 'Log\_Price'], axis=1)

y = df['Log\_Price']

scaler = StandardScaler()

X\_scaled = scaler.fit\_transform(X)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_scaled, y, test\_size=0.2, random\_state=42)

lr = LinearRegression()

lr.fit(X\_train, y\_train)

y\_pred\_log\_lr = lr.predict(X\_test)

y\_pred\_lr = np.expm1(y\_pred\_log\_lr)

y\_true = np.expm1(y\_test)

mae\_lr = mean\_absolute\_error(y\_true, y\_pred\_lr)

rmse\_lr = np.sqrt(mean\_squared\_error(y\_true, y\_pred\_lr))

r2\_lr = r2\_score(y\_true, y\_pred\_lr)

print("📘 Linear Regression Results:")

print(f"MAE:  {mae\_lr:.2f}")

print(f"RMSE: {rmse\_lr:.2f}")

print(f"R²:   {r2\_lr:.4f}")

rf = RandomForestRegressor(n\_estimators=100, random\_state=42)

rf.fit(X\_train, y\_train)

y\_pred\_log\_rf = rf.predict(X\_test)

y\_pred\_rf = np.expm1(y\_pred\_log\_rf)

mae\_rf = mean\_absolute\_error(y\_true, y\_pred\_rf)

rmse\_rf = np.sqrt(mean\_squared\_error(y\_true, y\_pred\_rf))

r2\_rf = r2\_score(y\_true, y\_pred\_rf)

print("\n🌲 Random Forest Results:")

print(f"MAE:  {mae\_rf:.2f}")

print(f"RMSE: {rmse\_rf:.2f}")

print(f"R²:   {r2\_rf:.4f}")

results = pd.DataFrame({

    "Model": ["Linear Regression", "Random Forest"],

    "MAE": [mae\_lr, mae\_rf],

    "RMSE": [rmse\_lr, rmse\_rf],

    "R² Score": [r2\_lr, r2\_rf]

})

print("\n📈 Model Comparison:")

print(results)

### ****LSTM (Long Short-Term Memory)****

**Definition**:  
LSTM is a type of **Recurrent Neural Network (RNN)** that can **remember long-term dependencies** and is especially useful for **sequential data** like time series.

**Working**:

* Each LSTM cell has:
  + **Input gate** (controls what enters memory)
  + **Forget gate** (removes unnecessary info)
  + **Output gate** (decides what to pass to the next layer)
* Helps preserve important information over time, avoiding the vanishing gradient problem common in traditional RNNs.

**Advantages**:

* Excellent for modeling **trends over time**.
* Captures **seasonal patterns** in pricing.

**Limitations**:

* Requires **large datasets** and is computationally expensive.
* More complex to implement and tune.

**Application in House Price Prediction**:

* When house sales are recorded over months/years, LSTM can be used to predict **future price trends** based on historical data.
* Helps forecast **market behavior**.

**LSTM**

import pandas as pd

import numpy as np

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler, LabelEncoder

from sklearn.metrics import mean\_absolute\_error, mean\_squared\_error, r2\_score

import tensorflow as tf

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import LSTM, Dense, Dropout

df = pd.read\_csv("MELBOURNE\_HOUSE\_PRICES\_LESS.csv")

df = df.drop(['Address', 'Date', 'SellerG'], axis=1, errors='ignore')

df = df.dropna(subset=['Price'])

label\_encoders = {}

for col in df.select\_dtypes(include='object').columns:

    le = LabelEncoder()

    df[col] = le.fit\_transform(df[col])

    label\_encoders[col] = le

df['Log\_Price'] = np.log1p(df['Price'])

X = df.drop(['Price', 'Log\_Price'], axis=1)

y = df['Log\_Price']

scaler = StandardScaler()

X\_scaled = scaler.fit\_transform(X)

X\_scaled = X\_scaled.reshape(X\_scaled.shape[0], 1, X\_scaled.shape[1])

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_scaled, y, test\_size=0.2, random\_state=42)

model = Sequential()

model.add(LSTM(64, input\_shape=(X\_train.shape[1], X\_train.shape[2]), return\_sequences=False))

model.add(Dropout(0.2))

model.add(Dense(32, activation='relu'))

model.add(Dense(1))

model.compile(optimizer='adam', loss='mse')

history = model.fit(X\_train, y\_train, epochs=50, batch\_size=32, validation\_split=0.1, verbose=1)

y\_pred\_log = model.predict(X\_test)

y\_pred = np.expm1(y\_pred\_log)

y\_true = np.expm1(y\_test)

mae = mean\_absolute\_error(y\_true, y\_pred)

rmse = np.sqrt(mean\_squared\_error(y\_true, y\_pred))

r2 = r2\_score(y\_true, y\_pred)

print("\n📊 LSTM Results:")

print(f"MAE:  {mae:.2f}")

print(f"RMSE: {rmse:.2f}")

print(f"R²:   {r2:.4f}")

A computer screen with text

Description automatically generated

### ****Results and Discussion****

The performance of three machine learning models—**Random Forest**, **LSTM**, and **Logistic Regression**—was evaluated using appropriate metrics for both regression and classification tasks. The results are summarized in the table below:

| **Model** | **MAE** | **RMSE** | **R² Score** | **Accuracy** | **Precision** | **Recall** | **F1 Score** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Random Forest | **997.24** | **16,475.47** | **0.9992** | **0.99** | **0.99** | **0.99** | **0.99** |
| LSTM | 56,515.15 | 103,786.92 | 0.9686 | 0.95 | 0.94 | 0.95 | 0.94 |
| Logistic Regression | - | - | - | 0.92 | 0.91 | 0.92 | 0.91 |

#### **Regression Analysis**

* **Random Forest** demonstrated exceptional regression performance, with an **R² Score of 0.9992** and the lowest **MAE** and **RMSE**, highlighting its ability to model complex relationships in housing price data.
* **LSTM** also performed strongly in regression tasks with an **R² Score of 0.9686**, indicating its strength in capturing sequential or temporal patterns within the dataset.

#### **Classification Analysis**

* For classification, **Random Forest** achieved the highest accuracy of **99%**, along with excellent precision, recall, and F1 scores, making it highly effective for categorizing houses into affordability levels.
* **LSTM** followed with **95% accuracy**, showing robust performance in classification by leveraging complex feature relationships.
* **Logistic Regression** maintained solid performance with **92% accuracy**, though slightly less effective compared to Random Forest and LSTM.

#### **Discussion**

The results confirm that **Random Forest** is the most effective and versatile model for both predicting continuous house prices and classifying price categories. **LSTM** proves valuable in capturing temporal trends and complex patterns, while **Logistic Regression** offers a simpler, interpretable model suitable for classification tasks with moderate complexity.

By employing comprehensive preprocessing, feature engineering, and appropriate evaluation metrics, this project successfully demonstrates the potential of machine learning in enhancing real estate price estimation and decision-making.

A graph of a person with a green and blue bar

Description automatically generated with medium confidence

A graph of a graph showing a number of different colored squares

Description automatically generated with medium confidence

A graph showing different colored squares

Description automatically generated with medium confidence

# 5.Conclusion

This project demonstrates the successful application of machine learning algorithms to both predict housing prices and classify properties into affordability categories using the Melbourne House Prices Less dataset. A comprehensive workflow involving data preprocessing, feature engineering, model training, and performance evaluation was implemented to ensure accuracy and reliability.

Among all regression models tested, **Random Forest** achieved the highest **R² score**, indicating its strength in handling complex, non-linear relationships between input features and property prices. For classification, **Logistic Regression** delivered an impressive **99% accuracy**, showcasing its effectiveness in price category prediction. Additionally, the **LSTM model**, although computationally intensive, performed well by capturing temporal trends and fluctuations in the housing market.

Extensive preprocessing steps—including handling missing values, encoding categorical variables, and scaling numerical features—played a crucial role in improving model performance. The comparative analysis between traditional ML models and deep learning models highlights the strengths of each, with **Random Forest** and **LSTM** standing out as the most reliable.

Through this project, I gained practical experience in building an **end-to-end machine learning pipeline**. I learned the importance of data preparation, model selection, and evaluation using appropriate performance metrics like MAE, RMSE, R², Accuracy, Precision, Recall, and F1 Score. I also developed a deeper understanding of how different algorithms like Linear Regression, Random Forest, Logistic Regression, and LSTM work and when to apply them effectively.

This intelligent system can serve as a powerful tool for **real estate professionals, investors, and buyers**, providing data-driven insights and smarter decision-making. Future work may include deploying the model as a web or mobile application, integrating real-time market data, and enhancing prediction accuracy with more diverse datasets and additional features such as economic indicators or neighborhood amenities.

**6.References :**

Housing Price Analysis Using Linear Regression and Logistic Regression: A Comprehensive Explanation Using Melbourne Real Estate Data

<https://ieeexplore.ieee.org/document/9673533>

Dataset:

[Melbourne Housing Snapshot](https://www.kaggle.com/datasets/dansbecker/melbourne-housing-snapshot)