**MACHINE LEARNING FINAL PROJECT**

REAL-ESTATE AI  
 ZANEB AKHTAR

About Project:

The project is a real estate application featuring three key machine learning models integrated with a user-friendly GUI built using Anvil. The first module, a Loan Predictor, utilizes K-Nearest Neighbours (KNN), Naive Bayes, and Logistic Regression to predict loan eligibility, dynamically selecting the best-performing model. The second module, a Property Price Predictor, leverages a Neural Network Regression model to estimate property prices accurately. The final module, a Housing Price Predictor, incorporates unsupervised techniques for data preprocessing and employs Linear Regression and Random Forest models for price prediction.

1. **A screenshot of a computer

   Description automatically generatedProperty Price Predictor Model:**

The model is a neural network designed to predict house prices based on six input features: year of sale, house age, distance from the city , number of stores, latitude, and longitude. It achieves this by learning patterns and relationships within the provided data through the following steps:

1. **Data Preprocessing**:  
   Features are normalized to bring them to a similar scale, ensuring efficient optimization during training.
2. **Model Architecture**:  
   The model has:
   * An input layer with 10 neurons to process the 6 input features.
   * Two hidden layers with 20 and 5 neurons, using the ReLU activation function to learn complex, non-linear relationships.
   * An output layer with a single neuron to predict house prices.
3. **Training**:  
   The model minimizes the Mean Squared Error (MSE) loss using the Adam optimizer. It iteratively adjusts weights by comparing predictions to actual prices and backpropagating errors.
4. **Early Stopping**:  
   Training halts automatically if validation loss stops improving, preventing overfitting.
5. **Prediction**:  
   After training, the model generates predictions on unseen test data, which are evaluated using metrics like MSE, MAE, and R² to measure accuracy.

**Dataset Information**

The dataset used for this project is stored in a CSV file named buying\_house.csv. It contains the following features:

1. Year of Sale: The year when the house was sold.
2. Age of the House: Age of the house at the time of sale.
3. Distance from City: Distance of the house from the city.
4. Number of Stores: The number of stores available in the locality.
5. Latitude: Geographical latitude of the house.
6. Longitude: Geographical longitude of the house.
7. Price (Label): The price of the house, which is the target value to be predicted.

A graph of a blue line

Description automatically generated**Exploratory Data Analysis (EDA)** was performed to understand the dataset and its features. Key steps included visualizing the distribution of house prices using histograms and examining correlations between features and the target variable using a heatmap. Missing data was checked to ensure data completeness. These analyses helped identify relationships, trends, and potential outliers, setting the foundation for effective preprocessing and model building.

A screenshot of a graph

Description automatically generated

**Building the Neural Network**

**A screenshot of a computer program

Description automatically generated**

**Purpose**: Defines a neural network with:

* Input layer: 6 features, 10 neurons.
* Hidden layers: 20 and 5 neurons with ReLU activation.
* Output layer: 1 neuron for price prediction.

**Loss Function**: Mean Squared Error (mse) measures prediction error.

**Optimizer**: Adam optimizes weights during training.

A white sheet with black text and green text

Description automatically generatedA graph with blue line and orange line

Description automatically generated

**Model Evaluation:**

The model's evaluation results demonstrate its strong performance in predicting house prices. The Mean Squared Error (MSE) of 0.16 indicates the average squared difference between predicted and actual values is low. The Mean Absolute Error (MAE) of 0.34 highlights that, on average, predictions deviate by 0.34 units from the true prices. The R² Score of 0.82 signifies that the model explains 82% of the variance in house prices, showcasing its accuracy and reliability.

A white background with black text

Description automatically generated

**A graph showing different colored dots

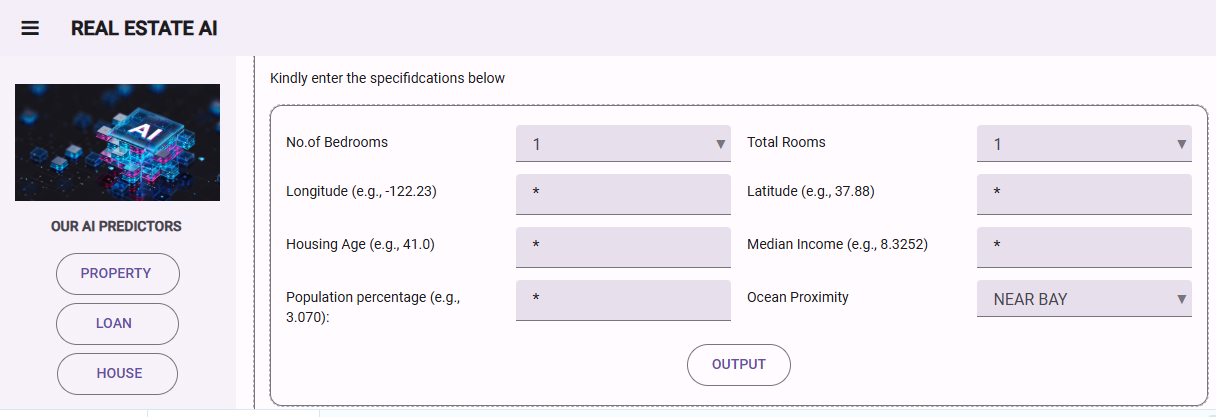
Description automatically generated with medium confidence**

**GUI Integration Using Anvil:**

The GUI for the property price predictor was developed using **Anvil**, a platform for creating web applications. The trained machine learning model was integrated into the backend of the application using Python, allowing users to input property details through the GUI. The application processes the input data, sends it to the model for prediction, and displays the predicted house price seamlessly on the interface, providing an intuitive experience for end-users

**A screenshot of a computer

Description automatically generatedGUI INTERFACE FOR PROPERTY PREDICTOR:**

1. **Housing Price Predictor Model:**

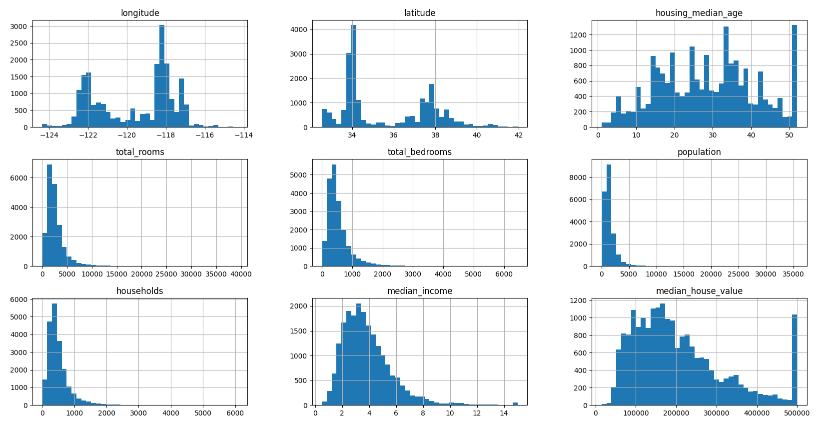
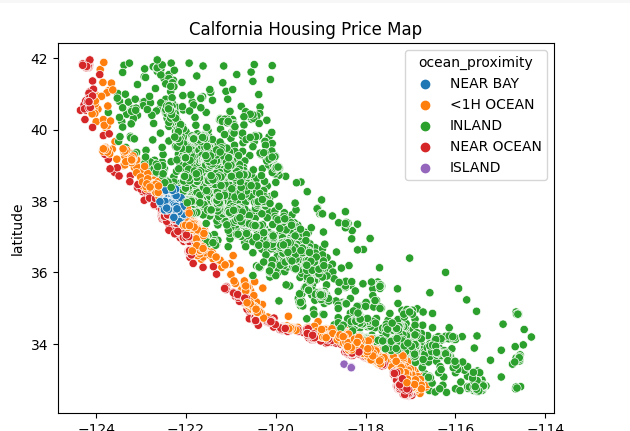
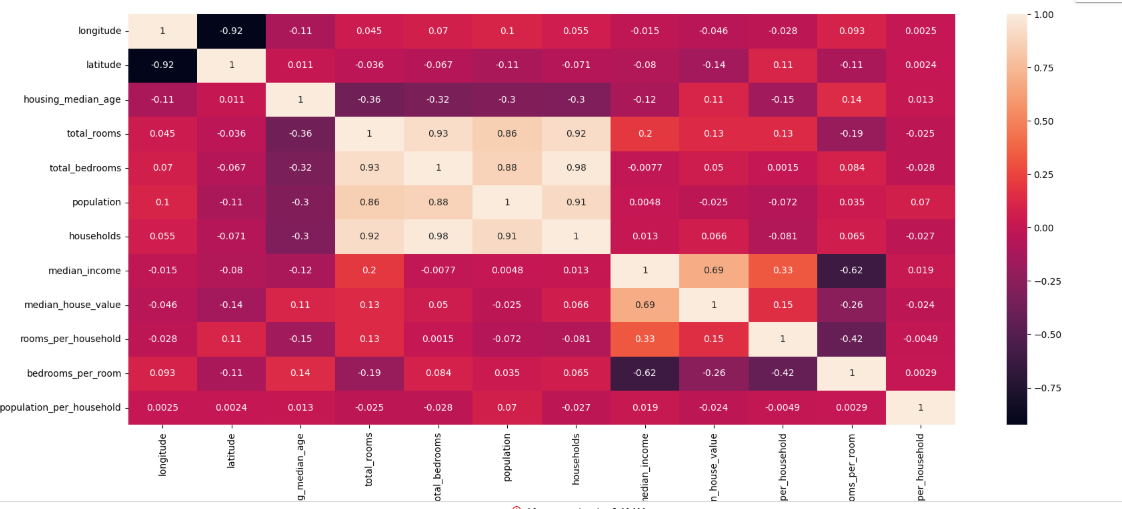
The primary goal of this model is to predict the median housing prices in California based on socio-economic and geographical data using advanced machine learning models. The dataset provides insights into various factors influencing housing prices, and the analysis aims to offer actionable predictions and insights for potential stakeholders.

**Data Preprocessing**

**Initial Data Exploration**

**A screenshot of a computer

Description automatically generated**

1. **Dataset Overview**:
   * Imported California Housing Prices dataset.
   * Verified data integrity and structure using .info() and .describe().
2. **Missing Values**:
   * Identified 207 missing values in the total\_bedrooms column.
   * Imputed missing values using the median strategy.
3. **Data Distribution**:
   * Visualized distributions using histograms and pair plots.
   * Applied log transformations to normalize skewed features (total rooms, total\_bedrooms, households, median\_house\_value).
4. **Geographical Distribution**:
   * Mapped longitude vs. latitude with ocean proximity categories.
5. **Feature Engineering**:
   * Added new features to address multicollinearity:
     + rooms\_per\_household
     + bedrooms\_per\_room
     + population\_per\_household
   * Dropped redundant columns: total\_rooms, total\_bedrooms, population, households.
6. **Outlier Analysis**:
   * Identified capped values in median\_house\_value (≥500001) accounting for 2.8% of the dataset.
7. **Randomization**:
   * Resampled the dataset multiple times to ensure randomness for train-test split.

A black text on a white background

Description automatically generated

**Data Preprocessing Pipeline**

1. **Numerical Features**:
   * Imputation (median strategy) and standard scaling.
2. **Categorical Features**:
   * One-hot encoding with first category dropped.
3. **Pipeline Integration**:
   * A screenshot of a computer

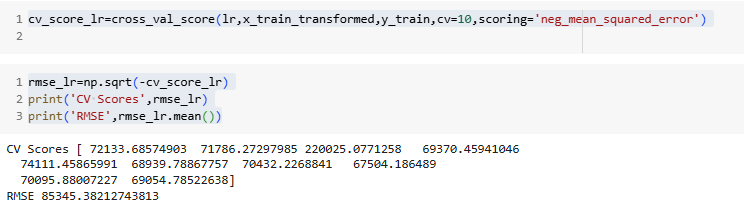
     Description automatically generatedCombined transformations using ColumnTransformer.

**Model Training and Evaluation**

**Linear Regression**

**A computer screen shot of a computer error

Description automatically generated**

* **Performance Metrics**:
  + Mean Absolute Percentage Error (MAPE): 27%
  + Root Mean Squared Error (RMSE): 70804
  + R² Score: 0.63
* For my Linear Regression model, I applied 10-fold cross-validation to evaluate its performance. Using cross\_val\_score, I computed the negative mean squared error (MSE) and then calculated the root mean squared error (RMSE) for each fold. The resulting RMSE values were [72133.69, 71786.27, 220025.08, 69370.46, 74111.46, 68939.79, 70432.23, 67504.19, 70095.88, 69054.79], with a mean RMSE of 85345.38.
* **Conclusion**: Linear Regression underfitted the data and failed to capture patterns effectively.

**Random Forest Regressor**

**A screenshot of a computer program

Description automatically generated**

* **Hyperparameters**: 100 estimators, max features = 8
* **Performance Metrics**:
  + MAPE: ~
  + RMSE: 48094
  + R² Score: 0.81
  + Explained Variance: 0.81
* **Feature Importance**:
  + A graph with blue bars

    Description automatically generated with medium confidenceVisualized feature importance, indicating significant predictors.
* **Conclusion**: Random Forest outperformed Linear Regression, achieving lower error and higher R².

**Support Vector Regressor (SVR)**

* **Kernel**: Linear
* **Performance Metrics**:
  + MAPE:
  + RMSE: 80722
  + R² Score:
* **Conclusion**: SVM performed poorly compared to other models.

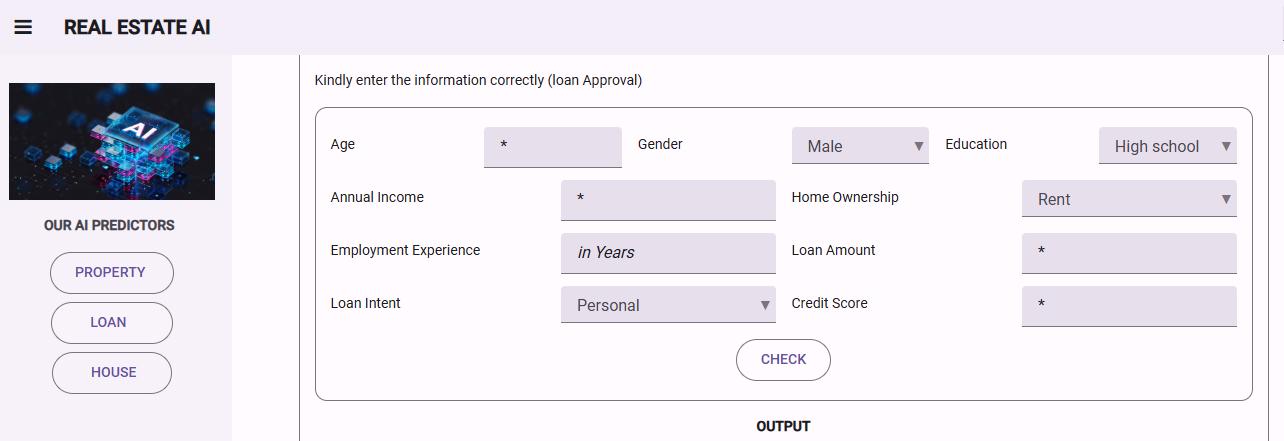
As the linear Regression model is performing poorly compared to Random Forest so we will be using RF to predict the values.

A screenshot of a computer screen

Description automatically generated**GUI Integration Using Anvil:**



corresponding anvil function:

**3.Loan Approval Predictor Model**

**1. Introduction**

The loan approval predictor model is designed to evaluate an individual's eligibility for a loan based on various personal, financial, and loan-related features. The dataset used for this model includes details about applicants, such as age, income, employment experience, credit score, and more, with the goal of predicting whether an individual will be approved for a loan.

**EDA and** **Data Preprocessing:**

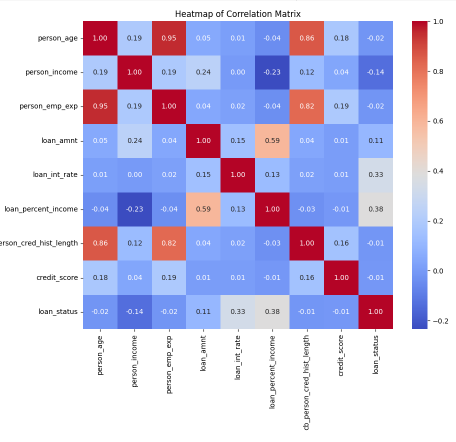
A screenshot of a computer

Description automatically generated

The dataset consists of various features that are crucial for predicting loan approval

* **person\_age**: Applicant's age
* **person\_income**: Applicant's income
* **person\_emp\_exp**: Applicant's employment experience
* **loan\_amnt**: Amount of loan requested
* **credit\_score**: Applicant's credit score
* **person\_gender**: Applicant's gender (categorical)
* **person\_education**: Applicant's education level (categorical)
* **person\_home\_ownership**: Applicant's home ownership status (categorical)
* **loan\_intent**: Intended purpose of the loan (categorical)

The data preprocessing involved handling missing values by dropping rows with missing data, encoding categorical variables using **LabelEncoder**, and removing outliers based on Z-scores for numerical columns. Numerical features were standardized with **StandardScaler**, and the dataset was split into 80% training and 20% testing sets, ensuring balanced target distribution. These steps ensured the data was clean and ready for model training.



**3. Model Selection and Evaluation**

Three machine learning models are chosen to predict loan approval:

1. **k-Nearest Neighbours (kNN)**
2. **Naive Bayes**
3. **Logistic Regression**

Each model is trained using the training data and evaluated on the testing data. The performance of the models is measured using accuracy, confusion matrix, and classification report.

**3.1 k-Nearest Neighbors (kNN)**

The kNN model is initialized with 10 neighbors and trained on the training data. The model is then evaluated on the test set:

A screenshot of a computer code

Description automatically generated

**3.2 Naive Bayes**

The Naive Bayes model is trained similarly, and its performance is evaluated:

A screenshot of a computer

Description automatically generated

**3.3 Logistic Regression**

The Logistic Regression model is initialized and trained:

A black text on a white background

Description automatically generated

A screenshot of a computer code

Description automatically generated

**Model Performance Comparison**

The model performance comparison shows that the **k-Nearest Neighbors (KNN)** and **Logistic Regression (LR)** models both achieve an accuracy of **0.82**, while the **Naive Bayes (NB)** model achieves an accuracy of **0.81**. This indicates that KNN and LR are slightly better at predicting loan eligibility than NB.

**Model Selection Criteria**

To determine the best performing model, the following criteria were used:

* **If the highest accuracy is from KNN**: The KNN model is chosen as the best.
* **If the highest accuracy is from NB**: The Naive Bayes model is selected.
* **Otherwise, Logistic Regression is chosen**.

Since both KNN and LR have identical accuracy in this case, KNN is selected as the best model due to it being prioritized in the comparison condition.

**Discussion of Model Selection**

The selection criteria prioritize the model with the highest accuracy. However, in cases of ties (like between KNN and LR), the model encountered first in the condition is chosen. While accuracy is an important metric, further considerations such as model interpretability, training time, and performance in specific subsets of data could influence the choice of the final model in real-world applications.

**Integration with Anvil for GUI:**

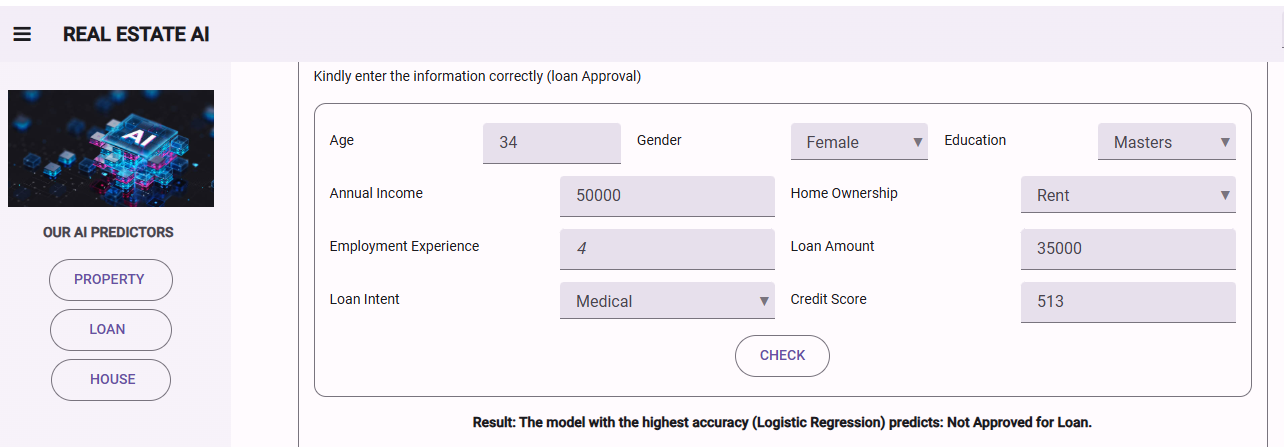
Anvil is used to integrate the model with a web application for making predictions. The app accepts user inputs, preprocesses them, and predicts loan eligibility using the best model.

The user input is preprocesses using the same preprocessing steps applied during training, including categorical encoding and numerical scaling. Once the user input is processed, the model predicts loan eligibility and displays the result.

Corresponding Anvil function:

A screenshot of a computer code

Description automatically generated

**Running GUI Interface:**