**House Price Prediction Web Application Report**

**Project Title: House Price Prediction Web Application**

**Author:**

**Kishore Kumar S**

**Master of Technology in Computer Science and Engineering  
VIT University**

**Contact Information:**

* **LinkedIn:** [**LinkedIn**](https://www.linkedin.com/in/kishorekumar1409/)
* **Email: kishorekumar1409@gmail.com**

**House Price Prediction Web Application Project Report**

**Table of Contents**

1. [Project Overview](#project-overview)
2. [Project Context](#project-context)
3. [Objectives](#objectives)
4. [Methodology](#methodology)
   * [Data Collection](#data-collection)
   * [Data Preprocessing](#data-preprocessing)
   * [Model Training](#model-training)
   * [Web Application Development](#web-application-development)
   * [Result Generation](#result-generation)
5. [Results](#results)
6. [Conclusion](#conclusion)
7. [Future Work](#future-work)
8. [References](#references)

**1. Project Overview**

The House Price Prediction Web Application is designed to predict house prices using a machine learning model integrated into a web application. Developed with Python Flask, this application allows users to input various property features and obtain predicted prices. The results, including selected features and predicted prices, are displayed on a results page and can be downloaded as a PDF report.

**Key Features**

* **User Interface:** A simple and interactive form for users to input features such as area, number of bedrooms, and other attributes.
* **Prediction Engine:** A trained Random Forest model that predicts house prices based on user inputs.
* **Results Display:** Predicted prices and selected features are shown on a results page.
* **PDF Report:** Users can download the results as a PDF document for offline review.

**2. Project Context**

**2.1 Background**

The real estate market relies heavily on accurate property valuations for buying, selling, and investing decisions. Predictive models in real estate can help stakeholders make informed decisions by providing estimates of property values based on historical data and various features.

**2.2 Motivation**

The motivation behind this project is to develop an accessible tool that leverages machine learning to predict house prices. By providing a web-based interface, the project aims to democratize access to predictive analytics in real estate, allowing users without specialized knowledge to benefit from advanced forecasting techniques.

**2.3 Objectives**

1. **Develop a Predictive Model:** Train a machine learning model to predict house prices using historical data.
2. **Create a Web Interface:** Design and implement a web application to interact with the model and collect user inputs.
3. **Generate PDF Reports:** Provide functionality to download prediction results as PDF reports for user convenience.

**3. Objectives**

1. **Model Training:**
   * Develop a machine learning model capable of predicting house prices based on input features.
   * Ensure the model is trained and validated for accuracy.
2. **User Interface Development:**
   * Build a user-friendly web application using Flask.
   * Design forms to collect input features from users and display results effectively.
3. **Result Handling:**
   * Implement functionality to display prediction results.
   * Create a system to generate and download PDF reports summarizing the results.

**4. Methodology**

**4.1 Data Collection**

The dataset used in this project includes various features relevant to house pricing. The data was collected from real estate listings and includes the following columns:

* **price:** The sale price of the house.
* **area:** The area of the house in square feet.
* **bedrooms:** The number of bedrooms.
* **bathrooms:** The number of bathrooms.
* **stories:** The number of stories.
* **mainroad:** Whether the house is located on a main road (yes/no).
* **guestroom:** Whether the house has a guest room (yes/no).
* **basement:** Whether the house has a basement (yes/no).
* **hotwaterheating:** Whether the house has hot water heating (yes/no).
* **airconditioning:** Whether the house has air conditioning (yes/no).
* **parking:** The number of parking spaces.
* **prefarea:** Whether the house is in a preferred area (yes/no).
* **furnishingstatus:** The furnishing status of the house (furnished/semi-furnished/unfurnished).

**4.2 Data Preprocessing**

1. **Handling Categorical Variables:**
   * Categorical variables such as mainroad, guestroom, basement, hotwaterheating, airconditioning, prefarea, and furnishingstatus were encoded using one-hot encoding to convert them into numerical values suitable for the model.
2. **Data Splitting:**
   * The dataset was split into training and testing sets to evaluate the model's performance and avoid overfitting.
3. **Normalization:**
   * Numerical features were scaled to ensure that the model treats all features equally.

**4.3 Model Training**

1. **Algorithm Selection:**
   * A Random Forest Regressor was chosen due to its ability to handle a mix of numerical and categorical data and its robustness in regression tasks.
2. **Training:**
   * The Random Forest model was trained using the training dataset.
   * Hyperparameters such as the number of trees and maximum depth were optimized using cross-validation.
3. **Evaluation:**
   * Model performance was evaluated using metrics such as Mean Absolute Error (MAE) and R-squared on the testing set to ensure the model's accuracy and reliability.

**4.4 Web Application Development**

1. **Framework:**
   * Python Flask was used to build the web application due to its simplicity and ease of integration with Python code.
2. **Application Structure:**
   * **app.py:** Contains the main Flask application logic, including routes for the home page and result display.
   * **train\_model.py:** Script to train the Random Forest model and save it along with the feature list.
3. **User Interface:**
   * HTML and CSS were used to create a responsive and user-friendly interface.
   * The home page includes a form for users to input features.
   * The result page displays the predicted price and allows users to download the results as a PDF.

**4.5 Result Generation**

1. **Prediction:**
   * User inputs are processed by the trained model to generate predictions.
2. **PDF Report:**
   * The FPDF library was used to create PDF reports containing the prediction results, selected features, and other relevant information.
   * Users can download the PDF directly from the result page.

**5. Results**

The web application successfully integrates the Random Forest model to predict house prices based on user inputs. Key results include:

* **Model Accuracy:** The Random Forest model achieved satisfactory accuracy in price predictions, with performance metrics indicating good fit to the data.
* **User Experience:** The web application provides an intuitive interface for users to input features and view predictions.
* **PDF Reporting:** The PDF report generation functionality works as expected, allowing users to download their prediction results for offline review.

**6. Conclusion**

The House Price Prediction Web Application demonstrates the effective application of machine learning in a real-world scenario. By integrating a predictive model into a web application, the project provides a valuable tool for real estate analysis. The application is user-friendly, accurate, and capable of generating detailed reports, making it a useful resource for users interested in property valuations.

**7. Future Work**

1. **Model Improvement:**
   * Explore additional machine learning algorithms or advanced techniques such as Gradient Boosting or Neural Networks to improve prediction accuracy.
   * Incorporate more features or external datasets to enhance the model’s performance.
2. **User Feedback:**
   * Gather feedback from users to identify areas for improvement in the user interface and overall application functionality.
3. **Feature Expansion:**
   * Add features such as property location analysis, market trends, or comparative analysis with similar properties to provide more comprehensive insights.
4. **Scalability:**
   * Consider deploying the application on a cloud platform to handle a larger number of users and improve scalability.

**8. References**

* Scikit-learn Documentation
* Flask Documentation
* [FPDF Documentation](http://www.fpdf.org/)
* [Random Forest Algorithm Overview](https://en.wikipedia.org/wiki/Random_forest)