Ensemble Methodes

(AML Project)

Team Members

1. Mahmoud Mohamed Ahmed Sayed	20210876
2. Adel Ali Ibrahim Ali	20210481
3. Sayed Yosry El-Sayed Ragab	20220217
4. Mohamed Ahmed Mohamed Abu El-Hassan	20220380
5. Kareem Ehab Mahmoud Mohamed	20220342
6. Hussein Mohamed Hussein	20220139

Project Overview

Instructor: Dr. Salwa Emam.

Project description: This project focuses on leveraging ensemble learning techniques to accurately predict housing prices in California. Instead of relying on a single predictive model, multiple machine learning algorithms are combined to form a stronger, more robust predictor. The core idea is that ensemble methods—such as Random Forest, Gradient Boosting, and Voting Regressors—can reduce model variance and bias, leading to improved performance over individual models. Using the California Housing dataset, the project explores how different ensemble strategies contribute to higher accuracy and generalization in real-world housing price prediction

The Dataset: California Housing Price Prediction

Github Repository Link: https://github.com/MahmoudDiab152/housing-prices-ensemble-methods.git

Ensemble Learning:

Ensemble learning is a machine learning paradigm that combines multiple base models to produce a more accurate, stable, and robust predictive model. Rather than relying on a single learning algorithm, ensemble methods integrate the strengths of several models to minimize errors, reduce variance, and improve generalization. Common ensemble techniques include bagging (e.g., Random Forest), boosting (e.g., Gradient Boosting), stacking, and voting.

- Why Ensemble Learning is Used:
    1. Improved Accuracy:

       Combining multiple models often leads to higher predictive performance by averaging out individual errors.
    2. Reduction in Overfitting and Variance:

       Techniques like bagging reduce model variance, making predictions more stable and reliable on unseen data.
    3. Increased Robustness:

       Ensembles are less sensitive to noise and outliers, providing better generalization in real-world applications.
    4. Leverages Model Diversity:

       Different models may capture different patterns in the data. Ensemble methods aggregate these varied perspectives for more comprehensive learning.

5. Flexibility:

   Ensemble techniques can incorporate models of different types (e.g., linear, tree-based) and exploit their complementary strengths.

Single Decision Tree .vs Ensemble Techniques

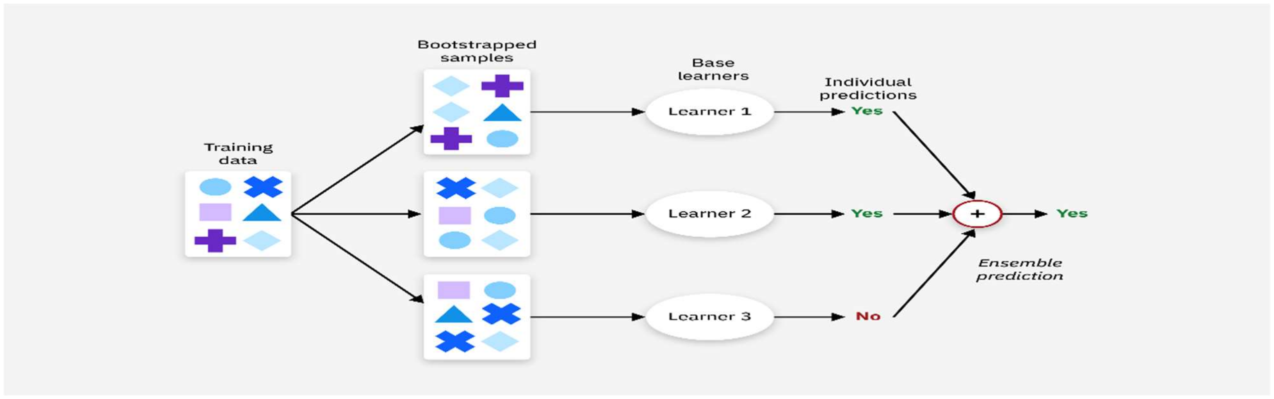
Single Decision Tree Regressor:

A Decision Tree Regressor splits the data into subsets based on feature values to make predictions. It builds a tree-like structure where each internal node represents a decision on a feature, and each leaf node gives a predicted value. It's simple, interpretable, and can model non-linear relationships.

- Advantages:
  1. Intuitive and easy to interpret
  2. Handles both numerical and categorical data without the need for scaling
  3. Captures non-linear relationships effectively
- Disadvantages:
  1. Prone to overfitting, especially on small datasets or deep trees
  2. Unstable; small variations in data can lead to different tree structures
  3. Generally lower accuracy compared to ensemble methods

Ensemble Techniques

Random Forest:

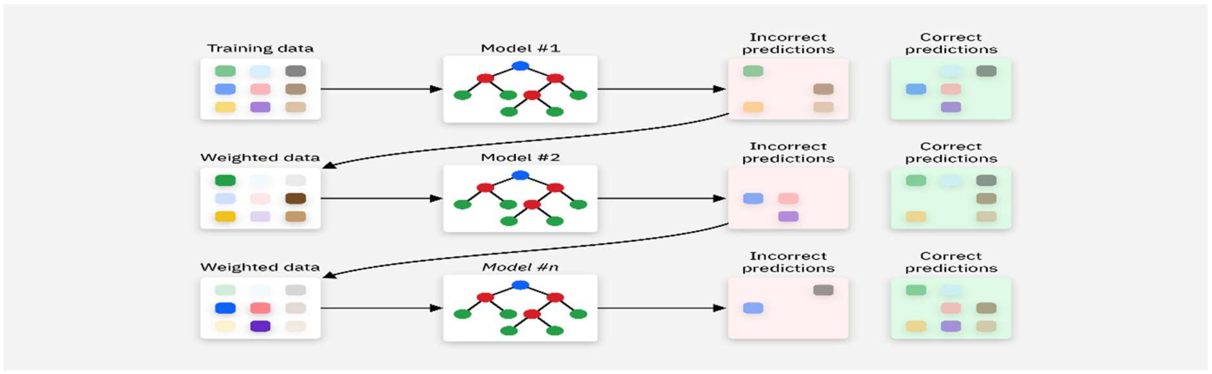


Random Forest is an ensemble technique that builds multiple decision trees using bootstrapped datasets and random feature selection. The final prediction is obtained by averaging the predictions of all individual trees.

- Advantages:
  1. Reduces overfitting by aggregating multiple trees
  2. Provides robust and accurate predictions
  3. Handles high-dimensional data and collinearity well

- Disadvantages:
  1. Less interpretable than a single decision tree
  2. More computationally intensive during training and prediction
  3. Can be biased toward features with more levels or categories

Boosting:

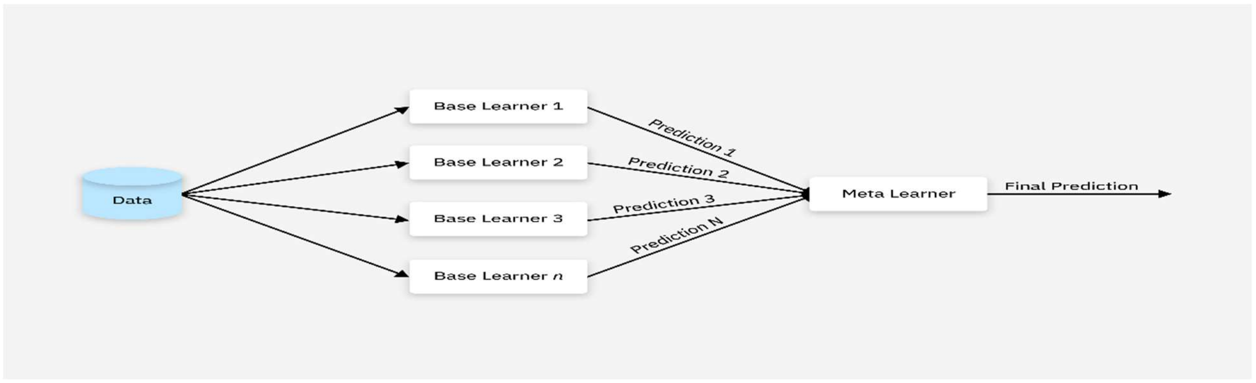


Boosting is an ensemble technique that builds a series of models sequentially. Each new model is trained to correct the errors made by the previous models. The predictions are then combined to form a strong overall prediction. The most common forms include AdaBoost, Gradient Boosting, and XGBoost.

- Advantages:
    1. High prediction accuracy, especially on structured/tabular data
    2. Effective at reducing bias by focusing on previously mispredicted data points
    3. Can be fine-tuned with many hyperparameters for control over complexity

- Disadvantages:
    1. Prone to overfitting if not carefully regularized
    2. Computationally expensive and slower to train
    3. Sensitive to outliers and noise in the data

Stacking:



Stacking is an ensemble learning method that combines multiple base models (level-0) and uses their predictions as input features to train a higher-level model (meta-learner). Unlike voting, the meta-model learns how to best combine the predictions of base models.

- Advantages:
  1. Often outperforms individual models and simpler ensembles
  2. Allows combination of very different model types (e.g., trees, linear models, etc.)
  3. Capable of capturing complex patterns across model outputs

- Disadvantages:
  1. More complex and computationally intensive to implement
  2. Risk of overfitting if not properly cross-validated
  3. Requires careful design to prevent data leakage (especially between layers)

California Housing Dataset

The dataset contains information collected from various districts in California. Each row represents aggregated data for a single district and includes features related to housing and demographics. The goal is to predict the median house value for each district based on its features.

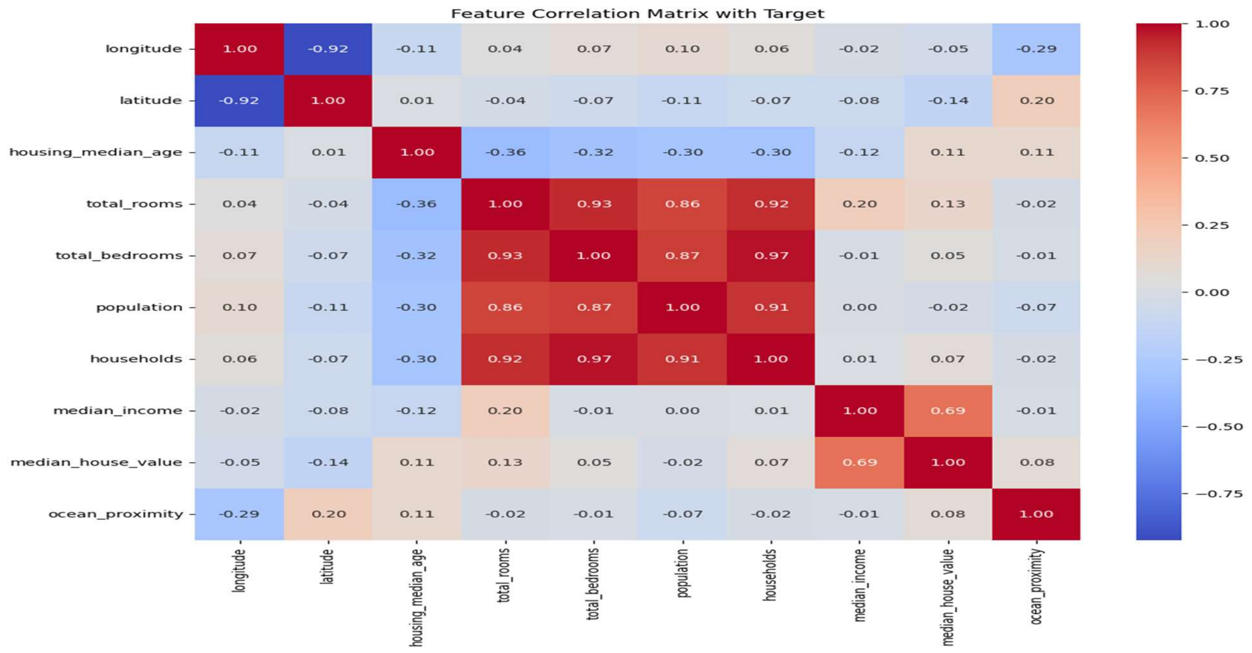
- Features Overview:

Column Name	Description
longitude	Geographical longitude of the block group
latitude	Geographical latitude of the block group
housing_median_age	Median age of the houses in the block group
total_rooms	Total number of rooms in all households in the block group
total_bedrooms	Total number of bedrooms in all households
population	Total population of the block group
households	Total number of households (occupied housing units)
median_income	Median income of households (in tens of thousands of dollars)
median_house_value	Median house value (target variable, in U.S. dollars)
ocean_proximity	Categorical feature indicating location relative to the ocean

- Target Variable:
  1. **More median_house_value**: Median house value for households within a block group (in U.S. dollars)

- Dataset Characteristics:
  1. · **Total Instances (Rows):** ~20,000
  2. · **Number of Features:** 9 input features + 1 target variable
  3. · **Data Types:**
     - **Numerical:** 8 features + 1 target
     - **Categorical:** 1 feature (ocean_proximity)
  4. · **Missing Values:** Possible in total_bedrooms (typically needs imputation)
  5. · **Prediction Task:** Regression

- Feature Correlation matrix with target variable:

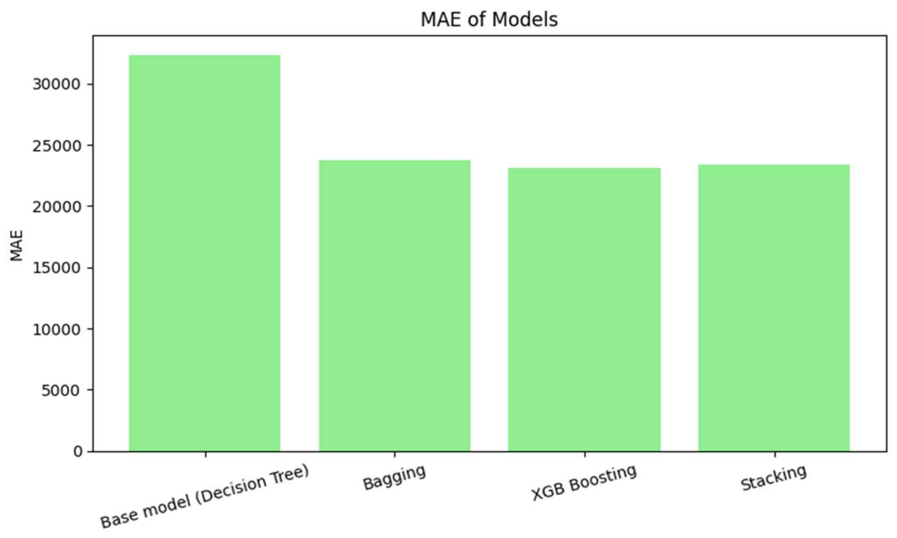


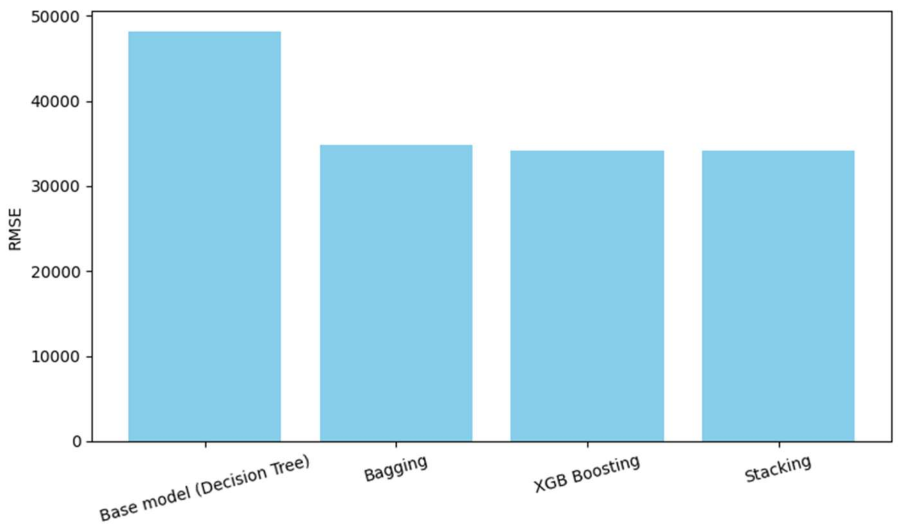
Evaluation Metrics Used

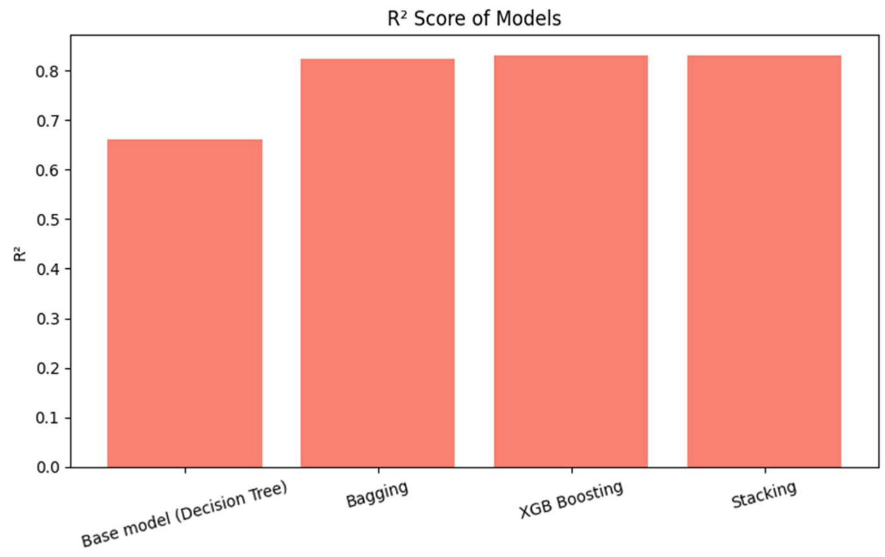
Metric	Description
MAE (Mean Absolute Error)	Measures the average magnitude of errors in predictions, without considering their direction. Smaller values indicate better performance.
MSE (Mean Squared Error)	Penalizes larger errors more heavily by squaring the residuals. Lower values indicate better performance.
RMSE (Root Mean Squared Error)	Square root of MSE, giving error in the same units as the target variable.
R^2 Score (Coefficient of Determination)	Represents the proportion of variance in the target variable explained by the model. Closer to 1 indicates stronger predictive power.

Models Performance Summary:

Model	R2 Score	MAE	RMSE
Decision Tree	~0.66	~32,298.25	~48,126.36
Random Forest	~0.82	~23834.49	~34900.38
XGBoost	~0.84	~22906.83	~33440.66
Stacking Regressor	~0.83	~23,376.36	~34,121.20


MAE of Models
MAE
30000
25000
20000
15000
10000
5000
0
Base model (Decision Tree)
Bagging
XGB Boosting
Stacking


RMSE
50000
40000
30000
20000
10000
0
Base model (Decision Tree)
Bagging
XGB Boosting
Stacking


R² Score of Models
R²
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Base model (Decision Tree)
Bagging
XGB Boosting
Stacking

Analysis and Insights:

- T he Decision Tree Regressor, while straightforward and interpretable, demonstrated relatively poor performance across all metrics, likely due to overfitting and lack of averaging.

- Random Forest significantly improved upon the Decision Tree, benefiting from bagging and model averaging to reduce variance.

- XGBoost offered further gains in accuracy, leveraging boosting to correct errors iteratively, resulting in the lowest MAE and a slightly better RMSE.

- The Stacking Regressor combined multiple models, delivering performance comparable to XGBoost, indicating its strength in learning meta-level patterns from base learners.

- ## Conclusion:

Among all models tested, XGBoost and Stacking Regressor provided the best performance, with nearly identical evaluation metrics. XGBoost had a slightly better MAE and RMSE, while Stacking was very close, suggesting either method is viable depending on specific deployment needs (e.g., interpretability, training time).

Tools & Techniques Used

Programming Language & Environment

- **Python** – Core programming language used.
- **Jupyter Notebook** – Interactive development environment for code execution and visualization.

Python Libraries

- **NumPy** – For numerical computations and array manipulations.
- **Pandas** – For data loading, preprocessing, and manipulation.
- **Matplotlib** – For basic data visualizations and plotting.
- **Seaborn** – For advanced and aesthetically pleasing statistical plots.
- **Scikit-learn (sklearn)** – For:
    - Data preprocessing (StandardScaler, OneHotEncoder)
    - Model selection and evaluation (train_test_split, GridSearchCV, cross_val_score)
    - Regression models:
        - DecisionTreeRegressor
        - RandomForestRegressor
        - VotingRegressor
        - StackingRegressor
    - Performance metrics:
        - mean_squared_error
        - mean_absolute_error
        - r2_score
- **XGBoost** – For implementing the gradient boosting regressor with high performance.

Thank you