Name: Revanth Kumar Rayi

Student Number: 23090414

GitHub Repository: https://github.com/therayi/Machine-Learning-Assignment.git

XGBOOST REGRESSION

Introduction

XGBoost (Extreme Gradient Boosting) is an advanced implementation of gradient boosting, designed for speed, efficiency, and accuracy. It is widely used in real-world machine learning applications due to its scalability, parallel computation, and ability to handle missing values.

Why Use XGBoost?

- ✅ Handles large datasets efficiently
- ✅ Reduces overfitting with regularization techniques
- ✅ Parallel processing for faster execution
- ✅ Supports missing values
- ✅ Provides feature importance insights

Dataset Overview: California Housing

The California Housing Dataset is a regression dataset used to predict median house values based on various features of California districts.

Features & Target Variable

Feature	Description
MedInc	Median income in the district
HouseAge	Average age of houses in the district
AveRooms	Average number of rooms per household
AveBedrms	Average number of bedrooms per household
Population	Population of the district
AveOccup	Average number of occupants per household
Latitude	Geographical latitude
Longitude	Geographical longitude

Target Variable: MedHouseVal → Median house value (in $100,000s)

How XGBoost Works (Step-by-Step Explanation)

Step 1: Initialize a Weak Model

XGBoost starts with an initial prediction (usually the mean value for regression). The error (residuals) is calculated between predicted and actual values.

Step 2: Compute Gradients (Loss Function Derivatives)

For each data point, the algorithm calculates the gradient of the loss function (how much the prediction is off).

Step 3: Train Decision Trees on Residuals

Instead of fitting on actual values, XGBoost trains a new decision tree to predict the residuals (errors of previous models).

Step 4: Update Predictions

New predictions are updated using the previous prediction + a learning rate times the new tree's output.

Step 5: Repeat for Multiple Rounds

More trees are added sequentially, each improving the overall prediction.

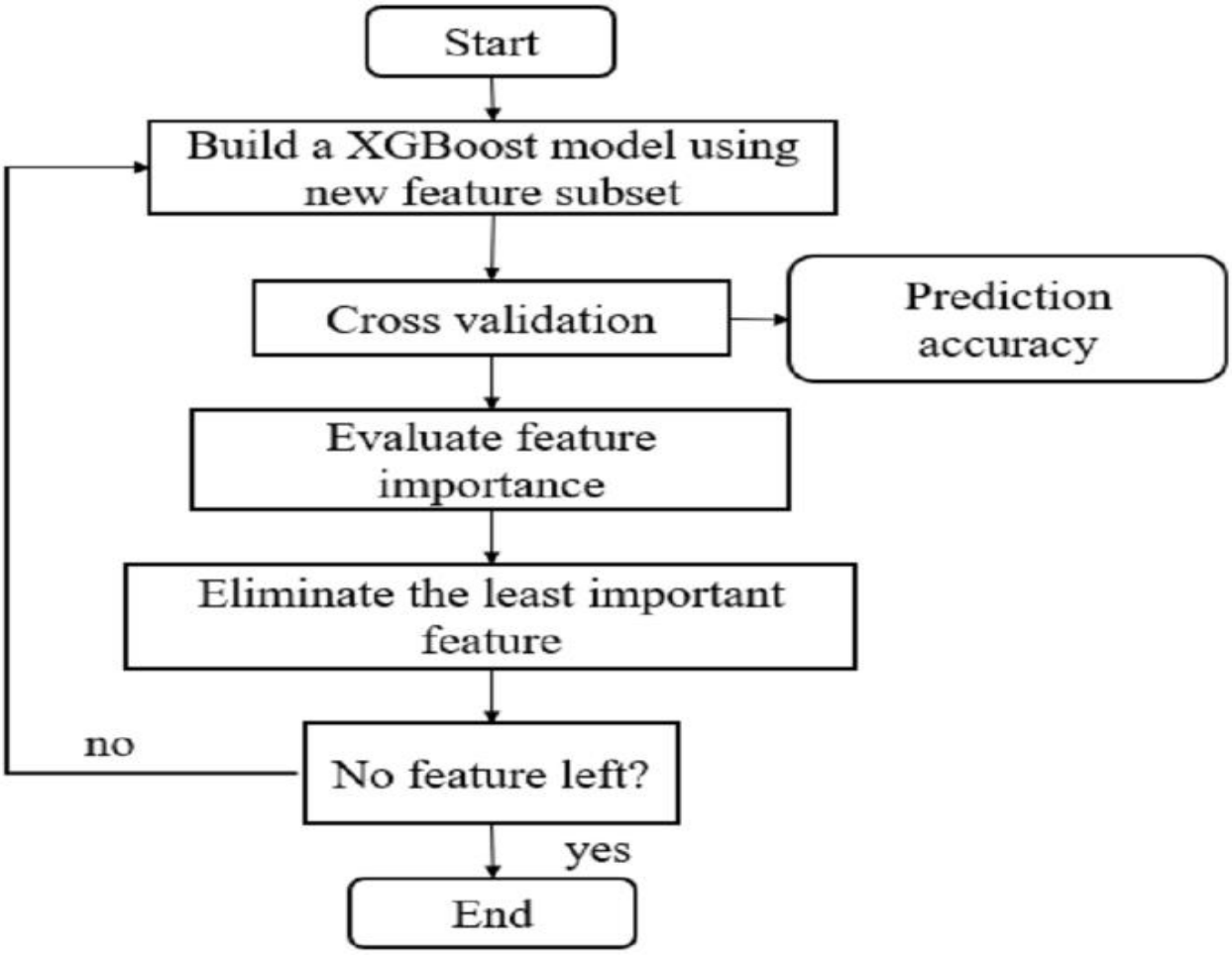
Step 6: Regularization (Prevent Overfitting)

XGBoost applies L1/L2 regularization (like Lasso & Ridge regression) to penalize complex trees.

Step 7: Make Final Prediction

The final output is the sum of all weak learners' predictions.

XGBoost Workflow Diagram



Implementation Details

Preprocessing

- Loaded dataset using fetch_california_housing()

- Standardized numerical features using StandardScaler()
- Split into training (80%) and testing (20%) sets

Model Training

- Used XGBoost Regressor (XGBRegressor) with:
  - ✅ n_estimators=100 (100 boosting rounds)
  - ✅ learning_rate=0.1 (balances convergence speed and performance)
  - ✅ objective='reg:squarederror' (suitable for regression tasks)
  - ✅ random_state=42 (ensures reproducibility)

📊 **Model Evaluation**

Metric	Value
Mean Absolute Error (MAE)	0.3178
Mean Squared Error (MSE)	0.2419
Root Mean Squared Error (RMSE)	0.4918
R^2 Score	0.81

Interpretation

✅ Low MAE/MSE/RMSE → Model makes accurate predictions

✅ R^2 Score of 0.81 → 81% of house price variation is explained

Feature Importance Analysis

1 XGBoost Feature Importance

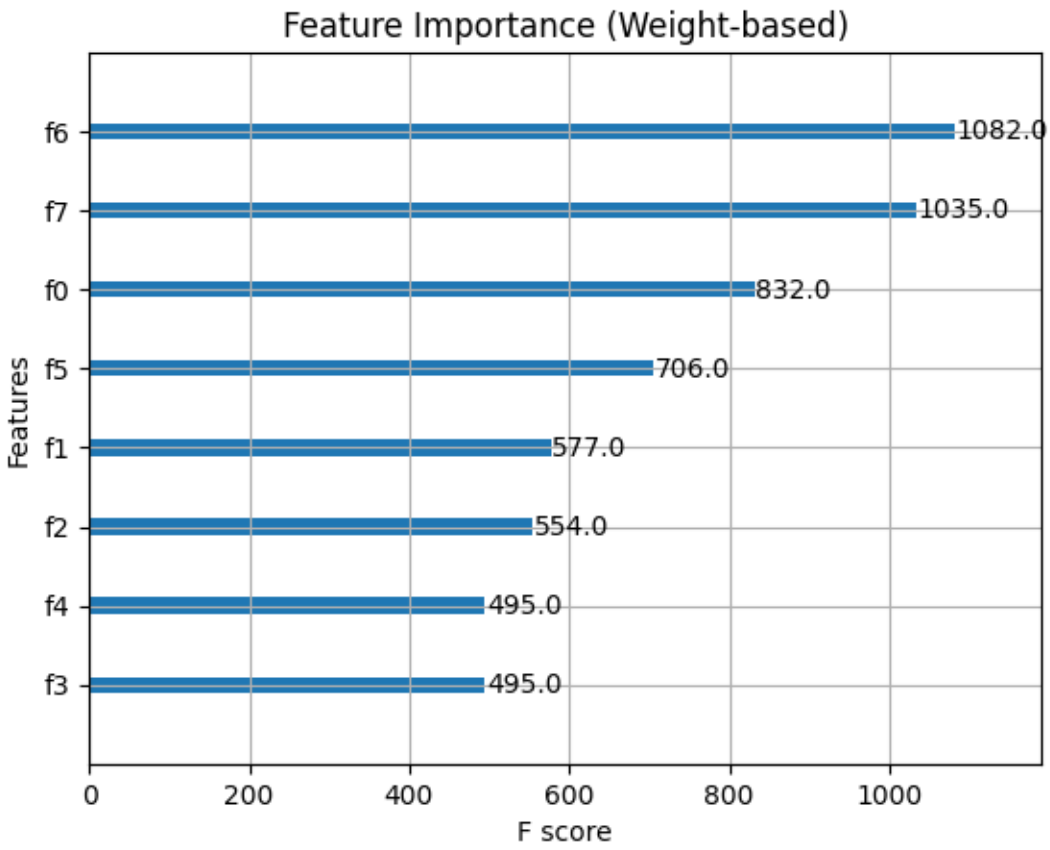
✅ Top Features Influencing House Prices:

1. Median Income (MedInc) - Strongest predictor
2. Average Number of Rooms (AveRooms)
3. House Age (HouseAge)

2 Model-Agnostic Permutation Importance

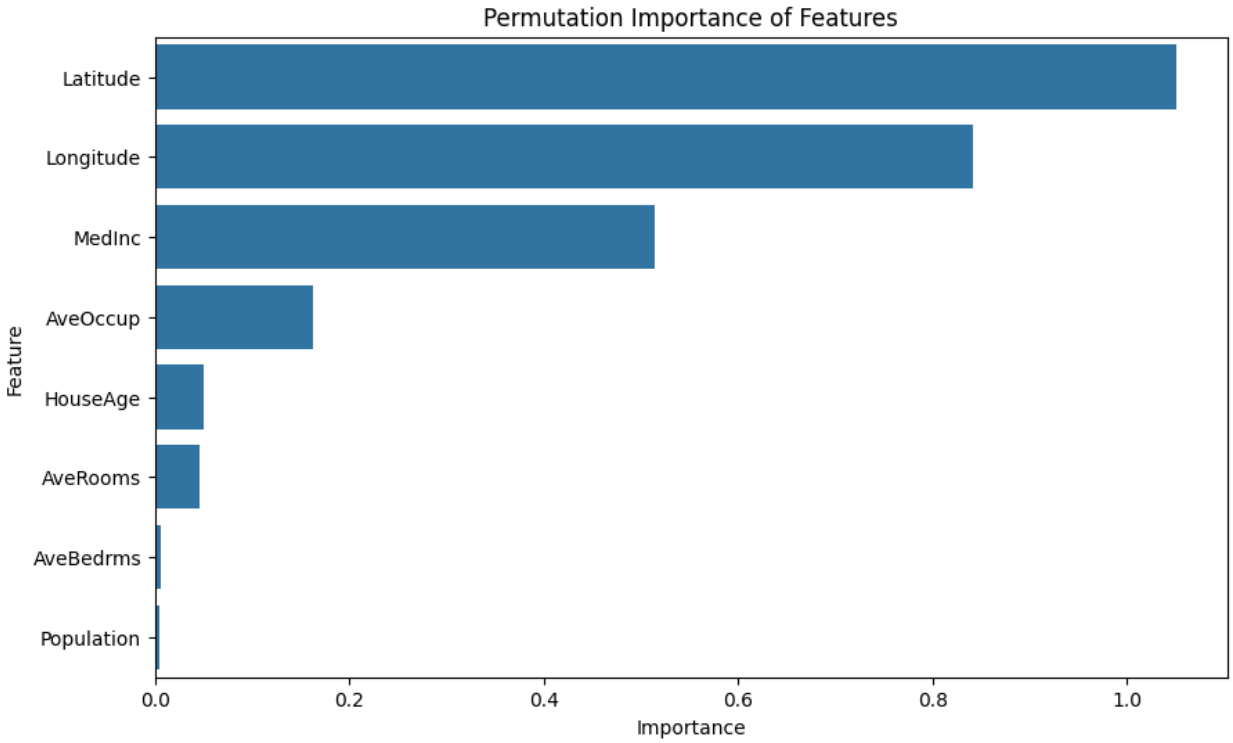
Feature	Importance
MedInc	Highest impact
AveRooms	Moderate impact
Latitude	Somewhat important

Key Insight: Median Income is the most important predictor.



1. Feature Importance (Weight-based)

- This bar chart represents the importance of different features in an XGBoost model based on the number of times each feature was used in trees (F-score).
- Features like f6 and f7 have the highest scores, indicating they were used the most in decision trees, making them the most influential variables.
- Lower-ranked features like f3 and f4 contributed less to the model's predictions.
- Understanding feature importance helps in feature selection, reducing dimensionality, and improving model performance.



2. Permutation Importance of Features

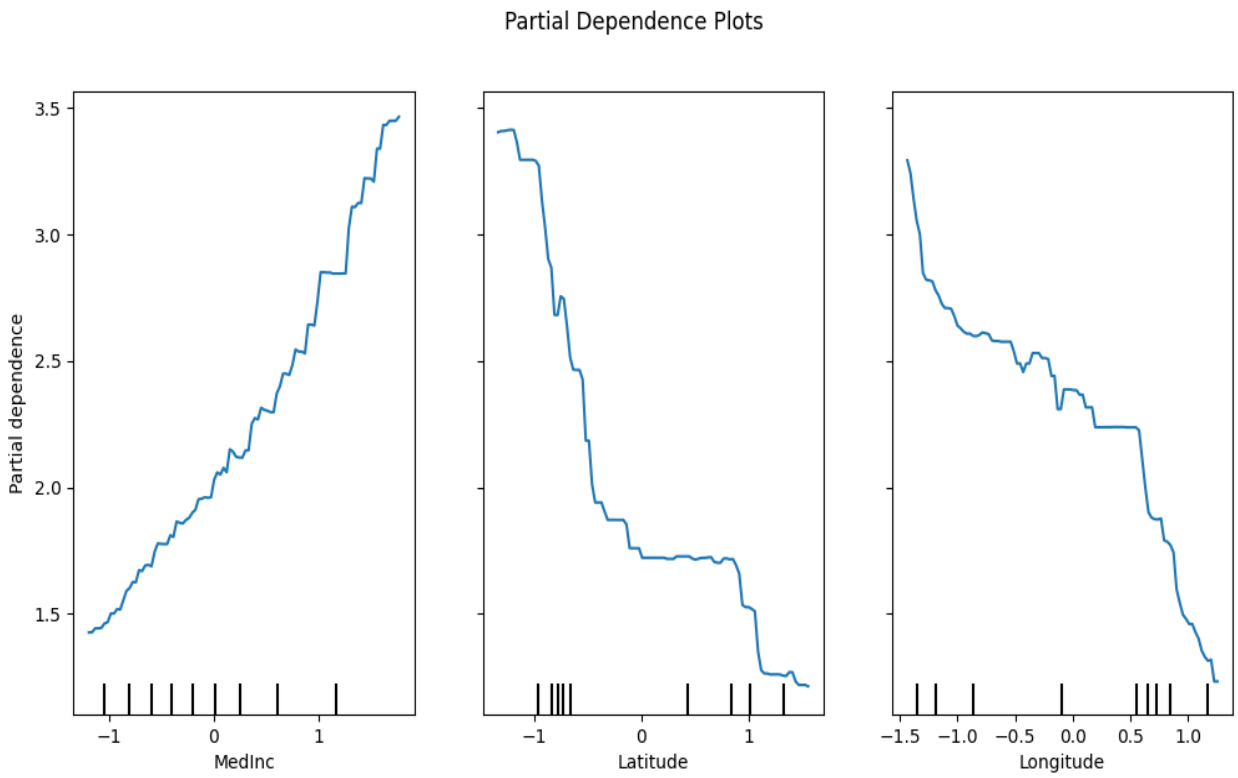
- This chart visualizes feature importance based on permutation importance, which measures the change in model performance when each feature is randomly shuffled.
- Latitude, Longitude, and MedInc (Median Income) are the most impactful features, meaning they have the highest predictive power.
- AveBedrms and Population have near-zero importance, suggesting they do not significantly contribute to the model.
- This analysis helps prioritize key features and discard less relevant ones for model optimization.

3 Partial Dependence Plots (PDP)

Key Observations:

✅ Higher Median Income → Higher house prices

✅ More Rooms per Household → Higher house prices (to an extent)



3. Partial Dependence Plots (PDP)

- These plots show the **marginal effect** of three features (**MedInc, Latitude, Longitude**) on the predicted output.
- **MedInc:** As income increases, the predicted target value rises steadily, indicating a strong positive correlation.
- **Latitude:** Shows a downward trend, meaning that the target variable decreases with increasing latitude.
- **Longitude:** Also has a decreasing effect, suggesting regional variations affect predictions.
- PDPs help in interpreting how features influence predictions in a non-linear model, making them crucial for explainability.

🌎 **Real-World Applications of XGBoost**

🏠 **Real Estate Pricing**

- Predict house prices based on economic & geographical factors
- Help buyers, sellers & agents

📈 **Financial Forecasting**

- Stock market prediction
- Loan risk assessment in banks

🏥 **Healthcare Analytics**

- Predict disease risks (diabetes, cancer)
- Hospital resource optimization

🛒 **E-Commerce & Marketing**

- Customer churn prediction
- Personalized recommendations

- 🔹 XGBoost is a powerful regression algorithm for structured datasets.
- 🔹 Feature Importance Analysis reveals which factors influence predictions the most.
- 🔹 Permutation Importance & PDP offer deeper interpretability.
- 🔹 XGBoost is widely used in real-world applications like real estate, finance, healthcare, and marketing.

Conclusion

XGBoost has proven to be a powerful and efficient machine learning algorithm for regression tasks, as demonstrated in predicting California housing prices. With its ability to handle large datasets, prevent overfitting through regularization, and provide insightful feature importance analysis, XGBoost is widely used across various industries, including real estate, finance, healthcare, and e-commerce. Our model achieved a strong R^2 score of 0.81, indicating a high level of accuracy in predicting house prices based on economic and geographical factors. The analysis highlighted that Median Income is the most influential factor affecting house prices, emphasizing the importance of economic conditions in real estate valuation. By leveraging additional techniques like permutation importance and partial dependence plots, we improved the model's interpretability, making it more transparent and trustworthy. Moving forward, fine-tuning hyperparameters, exploring SHAP values, and applying the model to real-world datasets can further enhance its predictive power and real-world applicability. In conclusion, XGBoost is a versatile, high-performance algorithm that can drive data-driven decision-making across multiple domains.

References

1. Chen, T., & Guestrin, C. (2016). *XGBoost: A Scalable Tree Boosting System.* Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. https://arxiv.org/abs/1603.02754
2. Lundberg, S. M., & Lee, S. I. (2017). *A Unified Approach to Interpreting Model Predictions.* Advances in Neural Information Processing Systems (NeurIPS). https://arxiv.org/abs/1705.07874
3. Breiman, L. (2001). *Random Forests.* Machine Learning, 45(1), 5-32. https://link.springer.com/article/10.1023/A:1010933404324
4. Molnar, C. (2022). *Interpretable Machine Learning: A Guide for Making Black Box Models Explainable.* https://christophm.github.io/interpretable-ml-book/
5. Friedman, J. H. (2001). *Greedy Function Approximation: A Gradient Boosting Machine.* The Annals of Statistics, 29(5), 1189-1232. https://projecteuclid.org/euclid.aos/1013203451

6. Pedregosa, F., et al. (2011). *Scikit-learn: Machine Learning in Python.* Journal of Machine Learning Research, 12, 2825-2830.
https://jmlr.csail.mit.edu/papers/v12/pedregosa11a.html
7. Fisher, A., Rudin, C., & Dominici, F. (2019). *All Models are Wrong, but Many are Useful: Learning a Variable's Importance by Studying an Entire Class of Prediction Models Simultaneously.* Journal of Machine Learning Research, 20(177), 1-81.
https://www.jmlr.org/papers/v20/18-760.html
8. Shapley, L. S. (1953). *A Value for n-Person Games.* Contributions to the Theory of Games, 2(28), 307-317.
https://www.rand.org/pubs/papers/P295.html
(This is the basis for SHAP values, used in feature importance analysis.)
9. Lemaître, G., Nogueira, F., & Aridas, C. K. (2017). *Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning.* Journal of Machine Learning Research, 18(1), 559-563.
https://jmlr.org/papers/v18/16-365.html
10. Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer.
https://web.stanford.edu/~hastie/Papers/ESLII.pdf