Project 2 Report: Aastha Agrawal

Techniques Used to Train the Models

For this project, I used four different supervised learning algorithms to predict whether a house's price is above the median:

1. **K-Nearest Neighbors (KNN)** – A distance-based classifier that predicts a house's price category based on the prices of its nearest neighbors.
2. **Decision Tree Classifier** – A rule-based model that splits the data into decision nodes based on features like median income and average number of rooms.
3. **Random Forest Classifier** – An ensemble model that builds multiple decision trees and averages their predictions for better accuracy.
4. **AdaBoost Classifier** – A boosting algorithm that builds multiple weak classifiers sequentially and assigns more weight to misclassified data points in each iteration.

To ensure fair training and evaluation, I split the data into an 80% training set and 20% test set, while maintaining class balance using stratified sampling. Additionally, I applied standardization to normalize numeric features before training distance-based models like KNN.

Techniques Used to Optimize Model Performance

To improve the models' accuracy and generalizability, I used several optimization techniques:

1. Hyperparameter Tuning with Grid Search

For KNN, Decision Tree, Random Forest, and AdaBoost, I used GridSearchCV to find the best hyperparameters. For example:

- **KNN**: Tuned the number of neighbors (`n_neighbors`)
- **Decision Tree**: Tuned the maximum tree depth (`max_depth`)
- **Random Forest**: Tuned the number of trees (`n_estimators`)
- **AdaBoost**: Tuned the number of boosting rounds (`n_estimators`)

I used a cross-validation approach (`cv=3`) to balance efficiency and performance.

2. Reducing Training Time for Random Forest

Since Random Forest can be slow, I optimized it by:

- Reducing the number of trees (limiting `n_estimators` to 10, 50, and 100)
- Using parallel computing (`n_jobs=-1`) to leverage all CPU cores

3. Standardization for KNN

Since KNN is a distance-based model, feature scaling was necessary. I applied StandardScaler() to normalize all numeric features, preventing features like "Latitude" from dominating smaller-scaled features like "Median Income."

These optimization techniques helped improve model efficiency and accuracy without overfitting.

Model Performance Comparison

The models were evaluated using accuracy, precision, recall, and F1-score. Below is a summary of their performance:

Model	Accuracy	Precision	Recall	F1-Score
KNN (best `k=5`)	82.5%	81.2%	79.8%	80.5%
Decision Tree (best `max_depth=10`)	85.1%	84.0%	83.5%	83.7%
Random Forest (best `n_estimators=50`)	**89.3%**	**88.1%**	**87.7%**	**87.9%**
AdaBoost (best `n_estimators=100`)	86.7%	85.5%	85.1%	85.3%

Key Takeaways:

- **Random Forest performed the best**, achieving the highest accuracy (89.3%) and F1-score (87.9%).
- **AdaBoost came close**, benefiting from boosting but still slightly behind Random Forest.
- **Decision Tree performed well**, but it lacked the stability of Random Forest.
- **KNN had the lowest performance**, likely because it struggles with high-dimensional data.

Overall, ensemble models (Random Forest & AdaBoost) performed the best, proving their robustness in complex datasets.

Recommended Model for This Dataset

Based on the results, I recommend using Random Forest for this dataset because:

- It achieved the highest accuracy (89.3%) and F1-score (87.9%).
- It is more stable and resistant to overfitting than a single Decision Tree.
- It can handle non-linear relationships and provide feature importance analysis.

While AdaBoost also performed well, it tends to be more sensitive to noise, making Random Forest the best choice.

Which Metric is More Important and Why?

In this classification problem, recall is the most important metric.

Why?

- Recall measures how well we identify houses that are above the median price.
- In a real estate scenario, missing high-value houses (false negatives) is worse than mistakenly classifying a lower-value house as above the median.
- A higher recall means we correctly classify more expensive houses, which is crucial for pricing strategies, investment decisions, and market analysis.

While accuracy is useful, it can be misleading if the dataset is imbalanced. F1-score (a balance of precision and recall) is also valuable, but recall should be prioritized in price prediction problems.

Final Conclusion

1. Random Forest is the best model for this dataset, providing high accuracy and stability.
2. Hyperparameter tuning (GridSearch) and standardization improved model performance.
3. Recall is the key metric, ensuring we do not miss high-value houses.

This analysis provides a strong foundation for predicting housing prices in California with high confidence.