Artificial Intelligence & Machine Learning – Task 2

Feature Engineering, Model Optimization & Performance Comparison

Name: Sahil Chhimpa
Internship Domain: Artificial Intelligence & Machine Learning
Organization: Maincrafts Technology
Task: Feature Engineering, Model Optimization & Performance Comparison
Dataset: California Housing Dataset

---

1. Introduction

In Task 1, a basic Linear Regression model was built to predict house prices.
The objective of **Task 2** is to move beyond a basic model and understand how **Machine Learning engineers improve model performance in real-world projects**.

This task focuses on:

- Feature scaling
- Training multiple regression models
- Comparing model performance
- Selecting the best-performing model using evaluation metrics

---

2. Dataset Description

The **California Housing Dataset** is a real-world dataset provided by scikit-learn.
It contains housing-related information collected from different districts of California.

Target Variable

- **HousePrice (Median House Value)**

Input Features

- Median Income
- House Age
- Average Rooms
- Average Bedrooms
- Population
- Average Occupancy

- Latitude
- Longitude

The dataset contains **20,640 rows and 9 columns** with **no missing values**, making it suitable for regression analysis.

---

3. Feature Engineering & Data Preprocessing

3.1 Feature Scaling

Feature scaling is a critical step in Machine Learning.
Since features exist on different numerical ranges, **StandardScaler** was used to normalize the input data.

Why Feature Scaling is Important

- Prevents features with large values from dominating others
- Improves model stability and convergence
- Essential for regularized models like Ridge Regression

After scaling, all features were brought to a common scale.

---

4. Train–Test Split

The dataset was split into:

- **80% Training data**
- **20% Testing data**

This ensures that model evaluation is performed on unseen data, reducing overfitting and improving generalization.

---

5. Models Trained

Instead of relying on a single algorithm, multiple models were trained and evaluated.

5.1 Linear Regression

- Used as a baseline model
- Simple and interpretable
- Helps compare improvements with other models

5.2 Ridge Regression

- Regularized version of Linear Regression
- Reduces overfitting by penalizing large coefficients
- Performs better when features are correlated

5.3 Decision Tree Regressor

- Captures non-linear relationships
- Handles complex feature interactions
- Can overfit if not controlled

---

6. Model Evaluation Metrics

Models were evaluated using:

- **RMSE (Root Mean Squared Error)**
- **R^2 Score (Coefficient of Determination)**

Metric Interpretation

- **Lower RMSE** → Better prediction accuracy
- **Higher R^2 Score** → Better explanatory power

A structured comparison table was created to objectively compare model performance.

---

7. Model Comparison & Selection

After evaluating all models:

- Linear Regression provided a strong baseline
- Ridge Regression showed improved generalization
- Decision Tree captured non-linear patterns but risked overfitting

Based on **balanced performance and stability**, the **best-performing model** was selected using RMSE and R^2 scores.

---

8. Visualization & Performance Validation

An **Actual vs Predicted House Price plot** was generated for the selected model.

Observations

- Points closer to the diagonal line indicate better predictions
- The visualization confirms that the model predictions closely match actual values

This visual validation supports the numerical evaluation metrics.

---

9. Conclusion

Task 2 successfully demonstrated how real-world Machine Learning workflows go beyond basic model training.

Key Learnings

- Importance of feature scaling
- Benefits of comparing multiple models
- Objective model selection using metrics
- Understanding overfitting and generalization

This task strengthened practical ML skills and aligned with industry-level practices.

---

10. Future Improvements

- Hyperparameter tuning using GridSearchCV
- Cross-validation for robust evaluation
- Feature engineering with interaction terms
- Trying ensemble models like Random Forest or Gradient Boosting
- Deploying the final model using a web application

---

11. Tools & Technologies Used

- Python
- Pandas & NumPy
- Scikit-learn
- Matplotlib
- Jupyter Notebook