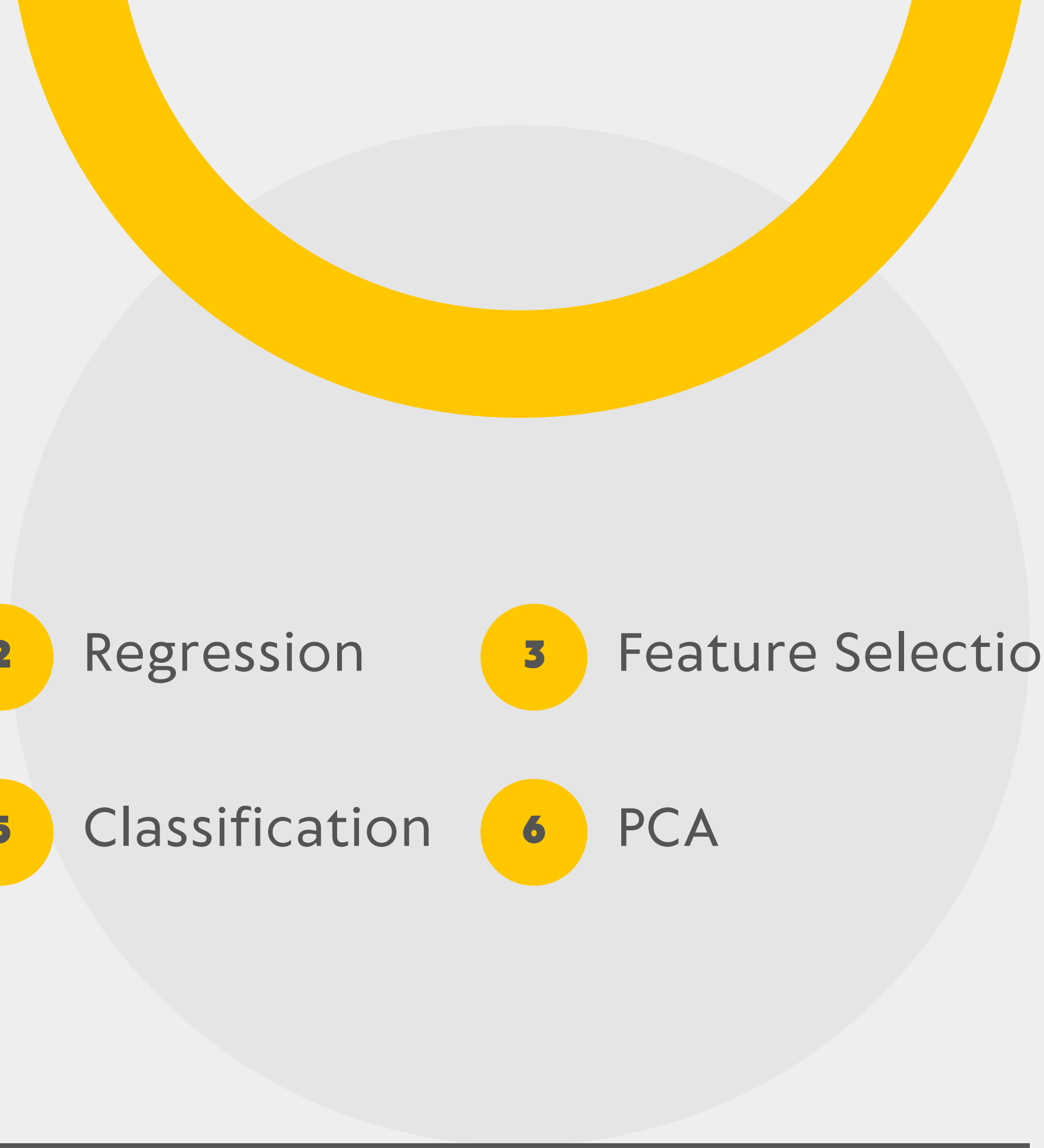


- **ACM476 TERM PROJECT
PHASE 4**

Melbourne Housing

Content

- 
- 1 Data Preprocessing and EDA
 - 2 Regression
 - 3 Feature Selection
 - 4 Clustering
 - 5 Classification
 - 6 PCA
-

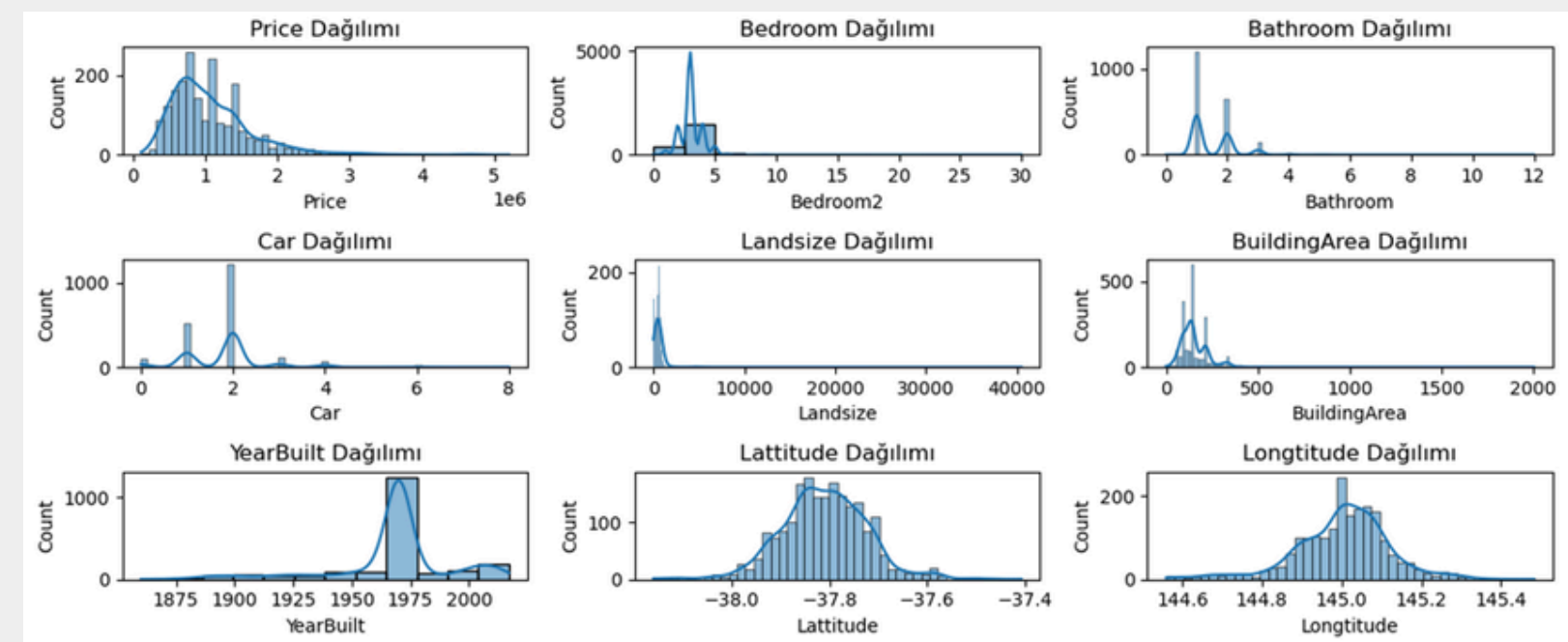
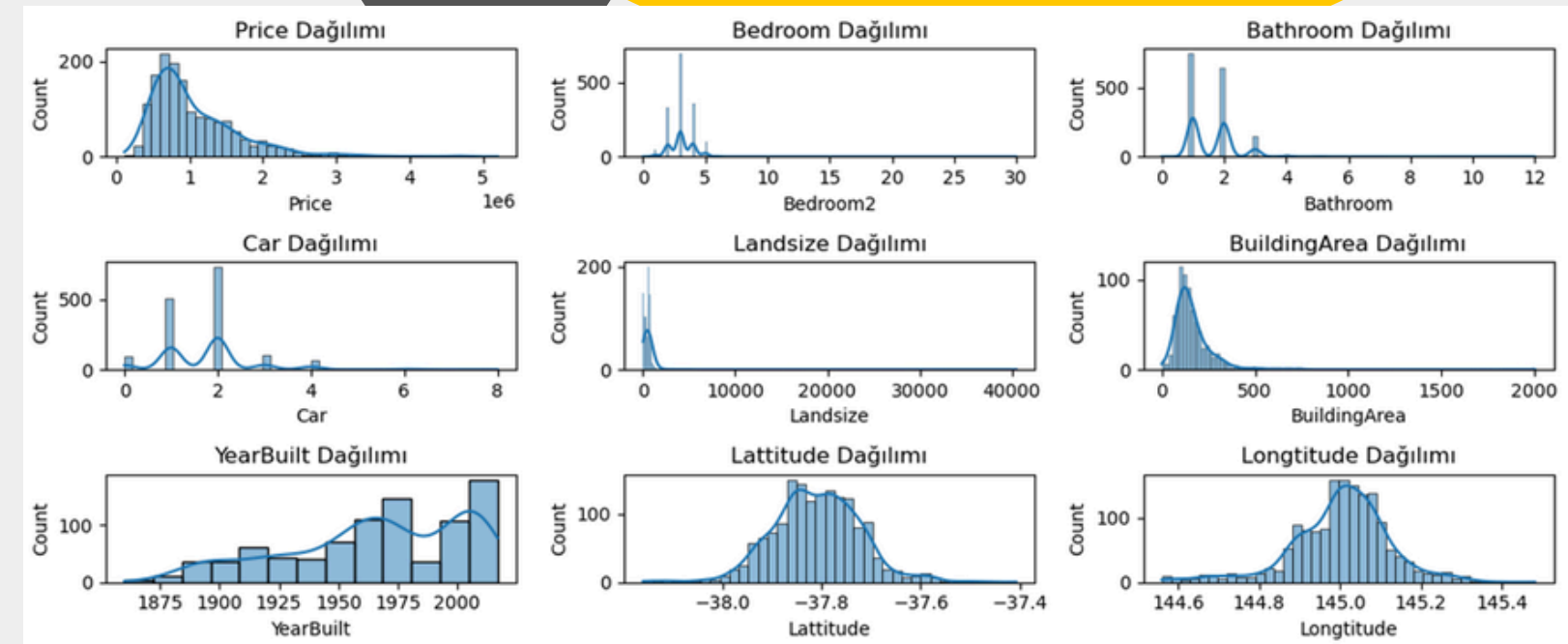
INTRODUCTION:

- The real estate market can be quite complex. Being able to predict housing prices accurately helps both buyers and sellers.
- The Melbourne Housing dataset, obtained from Kaggle, contains information about property sales, including attributes such as location, number of rooms, building area, and year built.
- We performed data preprocessing, exploratory data analysis (EDA), and implemented regression, clustering, classification, feature selection and principal component analysis techniques over the dataset.
- Goal, building predictive models and identify key factors influencing house prices.

- **Original Size:** 34,857 entries (rows)
- **Sample Used in Project:** 2,000 rows (randomly selected using random_state=5006)
- **Number of Features:** 21
- **Categorical Features:**
 - Suburb, Address, Type (house type), Method, SellerG, Date, CouncilArea, Regionname
- **Numerical Features:**
 - Rooms, Price, Distance, Postcode, Bedroom2, Bathroom, Car (number of parking spaces), Landsize, BuildingArea, YearBuilt, Latitude, Longitude, Propertycount

Data Preprocessing and EDA:

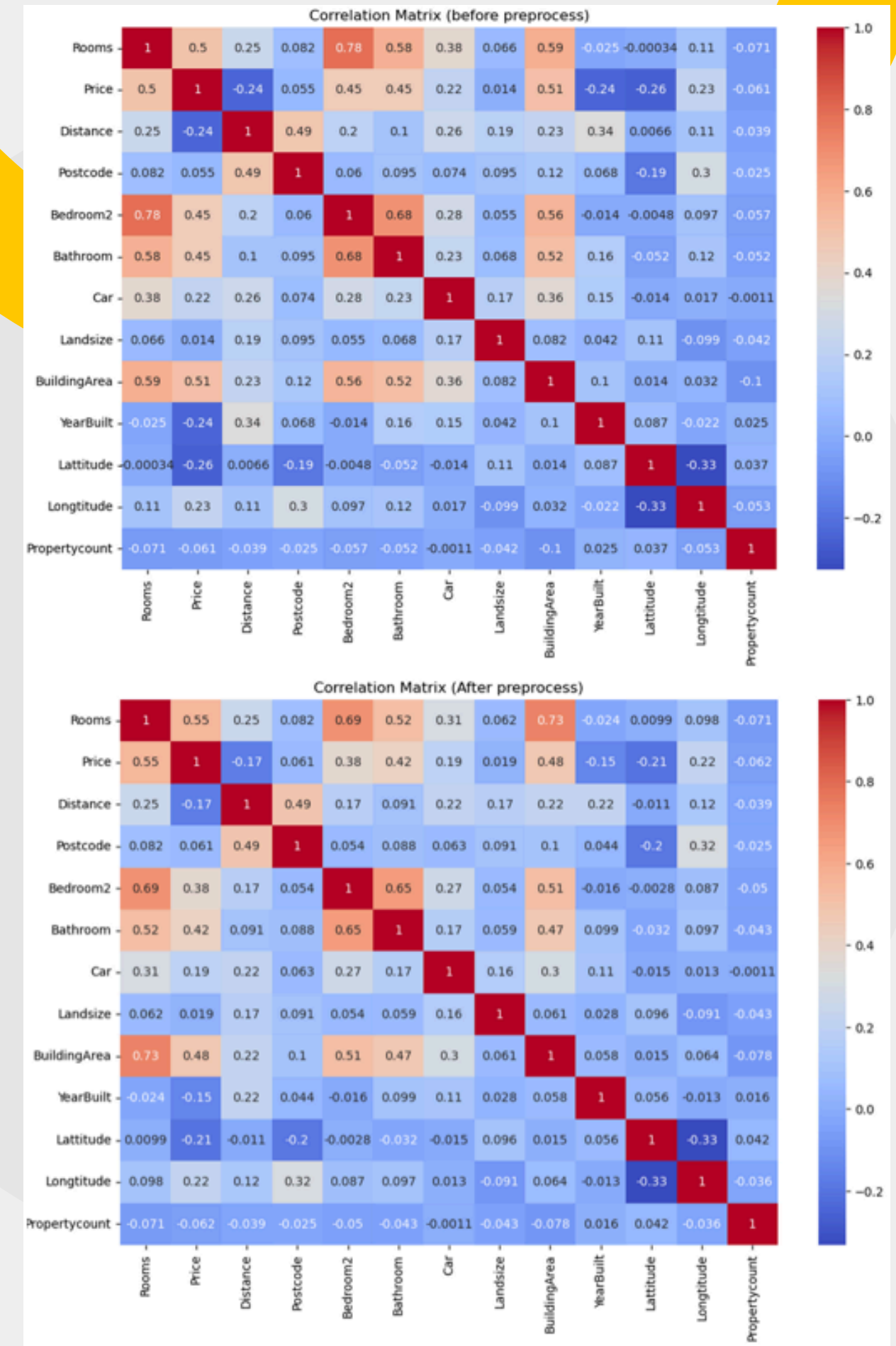
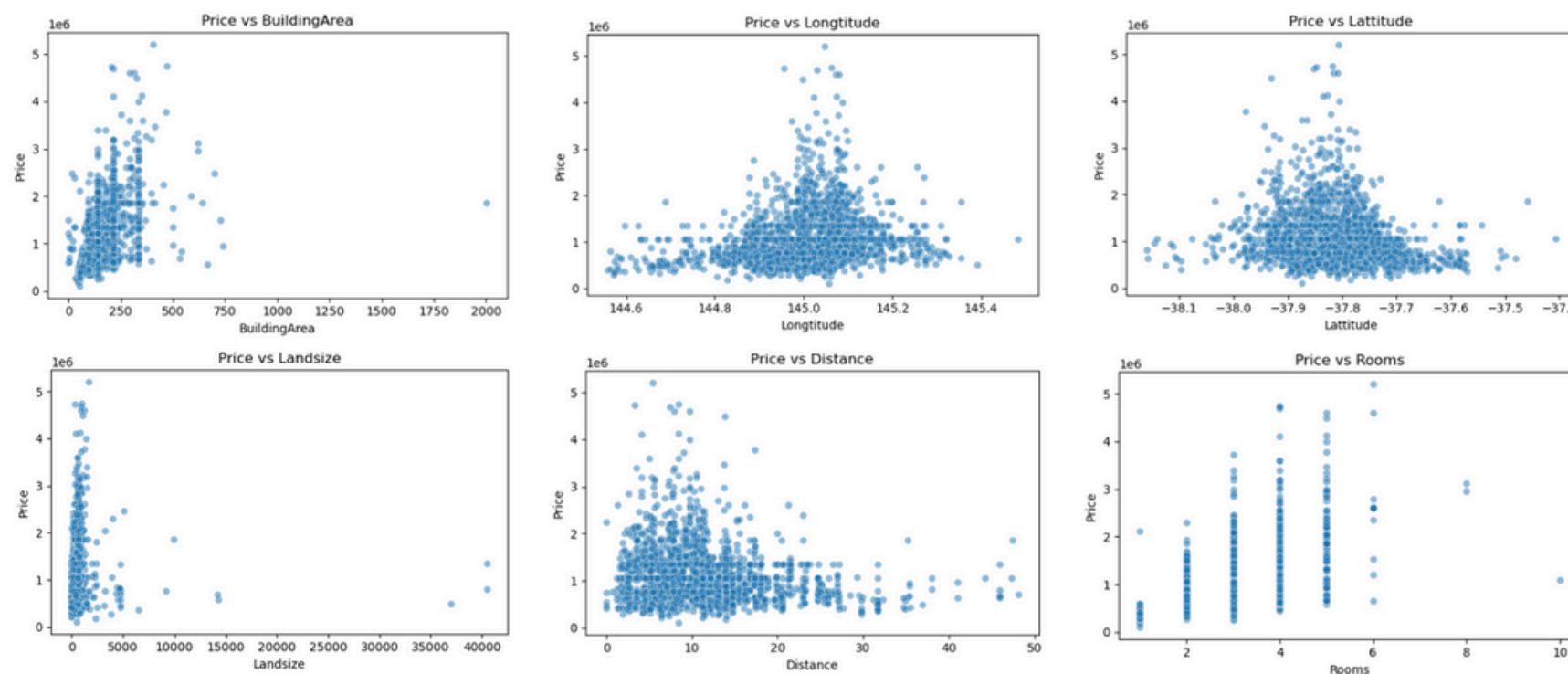
- Missing values were handled using averages, medians, or the most frequent values based on relevant property attributes.
 - Data type conversion applied
 - These steps ensure a clean dataset for analysis and model development.
-
- Price: Averaged based on the number of rooms.
 - Bedroom2, Bathroom, Car: Filled with the most frequent value.
 - Landsize: Averaged within the same suburb.
 - BuildingArea: Averaged based on the number of rooms.
 - YearBuilt: Filled with the median value.
 - Latitude & Longitude: Averaged within the same suburb.



Data Preprocessing and EDA:

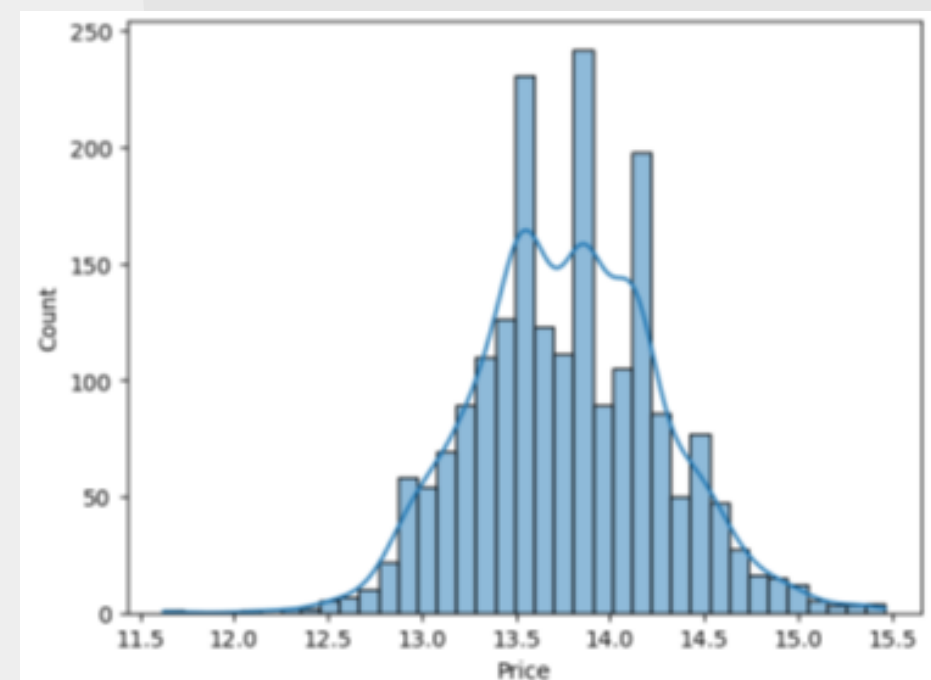
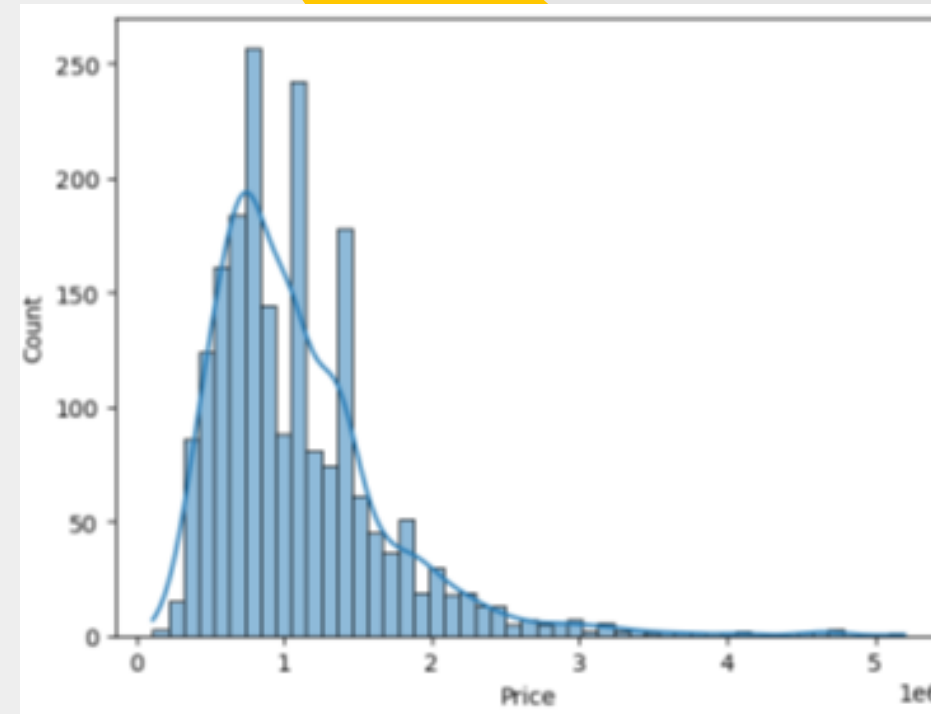
After cleaning the data, we analyzed feature relationships and removed some columns that didn't help with predictions:

- Address: Unique for each row, too detailed, and not useful for price prediction.
- SellerG: Seller names change, making future predictions inconsistent.
- Date: No strong link to price, just extra noise.
- Propertycount: The number of listings in an area had little effect on price.
- Postcode: Weak price impact, and better alternatives like CouncilArea exist.



Regression:

- **Target variable:** Price
- **independent variables:** Suburb, Rooms, Type, Method, Distance, Bedroom2, Bathroom, Car, Landsize, CouncilArea, Lattitude, Longitude, Regionname
- **Distribution analysis:** The Price variable was found to be right-skewed.
- **Applied log transformation:** Used `np.log1p` to normalize the Price values.
- **Applied regression models:**
 - Linear Regression
 - K-Nearest Neighbors (K=15)
 - Decision Tree (random_state=42)
 - Random Forest (random_state=42)
- **Evaluation metrics used:**
 - Mean Squared Error (MSE)
 - Root Mean Squared Error (RMSE)
 - R^2 Score
 - Mean Error Score



Model: Linear Regression
MSE: 0.07
RMSE: 0.26
R2 Score: 0.70
Mean error Score: 0.20

Model: K-Nearest Neighbors
MSE: 0.09
RMSE: 0.30
R2 Score: 0.62
Mean error Score: 0.22

Model: Decision Tree
MSE: 0.13
RMSE: 0.36
R2 Score: 0.42
Mean error Score: 0.27

Model: Random Forest
MSE: 0.06
RMSE: 0.25
R2 Score: 0.73
Mean error Score: 0.19

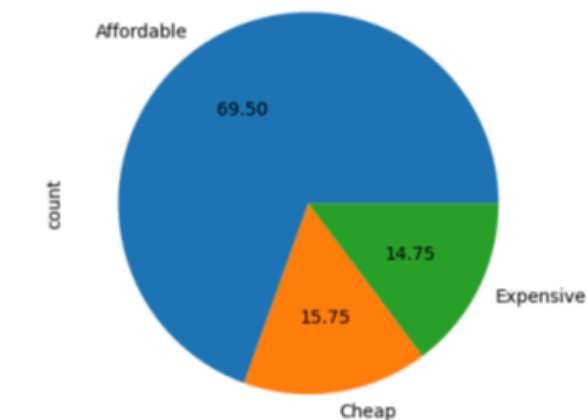
Feature Selection:

- **Target variable:** Price_Category (created using statistical binning of log-transformed (normalized) Price)
- **independent variables:** Suburb, Rooms, Type, Method, Distance, Bedroom2, Bathroom, Car, Landsize, CouncilArea, Lattitude, Longitude, Regionname
- **Created Price_Category using statistical binning:**
 - Cheap: min to (mean-std)
 - Affordable: (mean-std) to (mean+std)
 - Expensive: (mean+std) to max
- **Preprocessing:**
 - Rooms, Car, Bedroom2, and Bathroom treated as categorical features.
 - Applied standard scaling to numerical features.
 - Applied one-hot encoding to categorical features.
 - Applied label encoding to Price_Category
- **Class imbalance handling:** After creating the Price Category feature, we noticed that our data was imbalanced, so we applied random under-sampling to balance it.

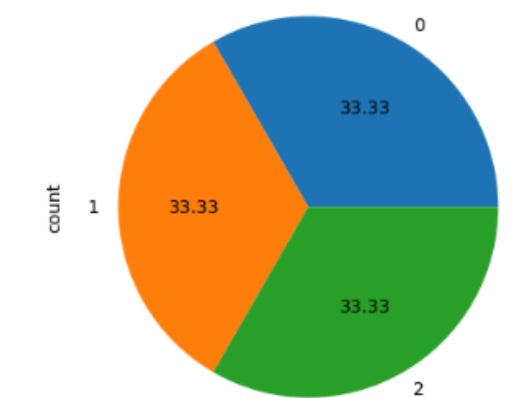
Price Category Frequency

```
Price_Category
Affordable    1390
Cheap         315
Expensive     295
Name: count, dtype: int64
```

Before random under sampling



After random under sampling



Top 5 Features:

```
BuildingArea
Distance
Longitude
Latitude
Bathroom_1
```

```
All features - CV accuracy: 0.78
Top 5 features - CV accuracy: 0.73
```

Top 5 Features:

```
BuildingArea
Longitude
Latitude
Distance
Landsize
```

```
All features - CV accuracy: 0.78
Top 5 (RF feature importances) - CV accuracy: 0.75
```

Clustering:

In the clustering phase, After standardizing the numerical features, hierarchical clustering revealed three distinct property groups with notable differences in size, location, and structure.

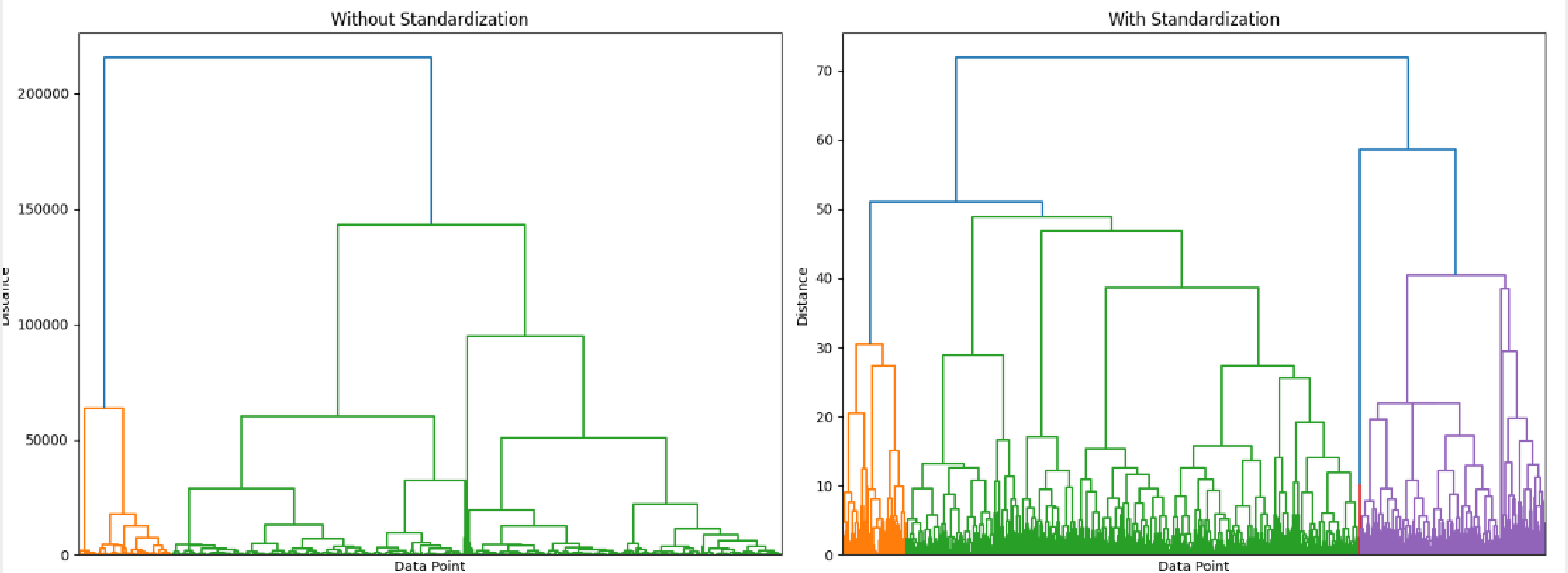
Standardization clearly improved cluster quality by preventing large-scale features from overpowering others. Clusters showed meaningful separation:

- Cluster 1: Properties located closer to the center, with smaller building/land areas and fewer rooms.
- Cluster 2: Houses located further from the center, characterized by larger areas and more rooms, representing spacious suburban properties.
- Cluster 3: Also centrally located like Cluster 1, but with higher room counts and larger size, suggesting a more premium inner-city segment

ROOMS	INT64
DISTANCE	FLOAT64
BEDROOM2	INT64
BATHROOM	INT64
CAR	INT64
LANDSIZE	INT64
BUILDINGAREA	INT64
YEARBUILT	INT64
LATTITUDE	FLOAT64
LONGTITUDE	FLOAT64
PROPERTYCOUNT	INT64

Average characteristics of each cluster:						
	Rooms	Distance	Bedroom2	Bathroom	Car	Landsize \
Cluster						
1	2.692726	11.214004	2.800136	1.278722	1.700884	550.957852
2	3.333333	33.500000	3.333333	2.000000	4.333333	39312.333333
3	4.076046	11.216920	3.933460	2.140684	2.058935	595.365019
	BuildingArea	YearBuilt	Lattitude	Longtitude	Propertycount	
Cluster						
1	125.702243	1965.342624	-37.807155	144.992298	7823.066621	
2	184.333333	1976.666667	-37.609163	144.684613	2861.333333	
3	235.615970	1975.904943	-37.822490	145.031807	6903.233840	

These groupings provide valuable insights into housing segmentation, useful for targeted analysis and decision-making.



Classification:

House prices were categorized into Low, Mid, and High classes. Log transformation ensured a balanced class distribution, helping prevent model bias. Each class showed clear differences in property characteristics:

Low-price houses: Smaller in size, fewer rooms, mostly located farther from the center (Northern Metropolitan).

Mid-price houses: Medium-sized properties, common in Southern Metropolitan regions.

High-price houses: Larger properties with more rooms, closer to the city center, also mostly in Southern Metropolitan.

	PriceCategory	Count
0	Low	709
1	Mid	633
2	High	658

PriceCategory	PriceRange	AvgPrice	AvgRooms	AvgBedrooms	AvgBathrooms
Low	112.000 - 763.562	594.037	2,44	2,67	1,23
Mid	763.562 - 1.200.000	971.315	3,07	3,09	1,45
High	1.200.000 - 5.200.000	1.738.934	3,71	3,57	1,85
PriceCategory	AvgBuildingArea (M)	AvgLandsize (M)	AvgDistance (KMs)	MostCommonType	MostCommonRegion
Low	114,33	601,78	12,20	u - unit, duplex;	Northern Metropolitan
Mid	147,47	594,54	11,02	h - house, villa;	Southern Metropolitan
High	205,14	666,50	9,58	h - house, villa;	Southern Metropolitan

Model Comparison:

To predict price categories, four models were tested. Random Forest consistently performed best, with:

- Highest accuracy (0.72)
- Strong precision, recall, and F1-scores across all classes
- Most consistent results in 10-fold cross-validation (average accuracy: 0.729)

Thus, Random Forest was selected as the final model due to its robust and balanced classification performance.

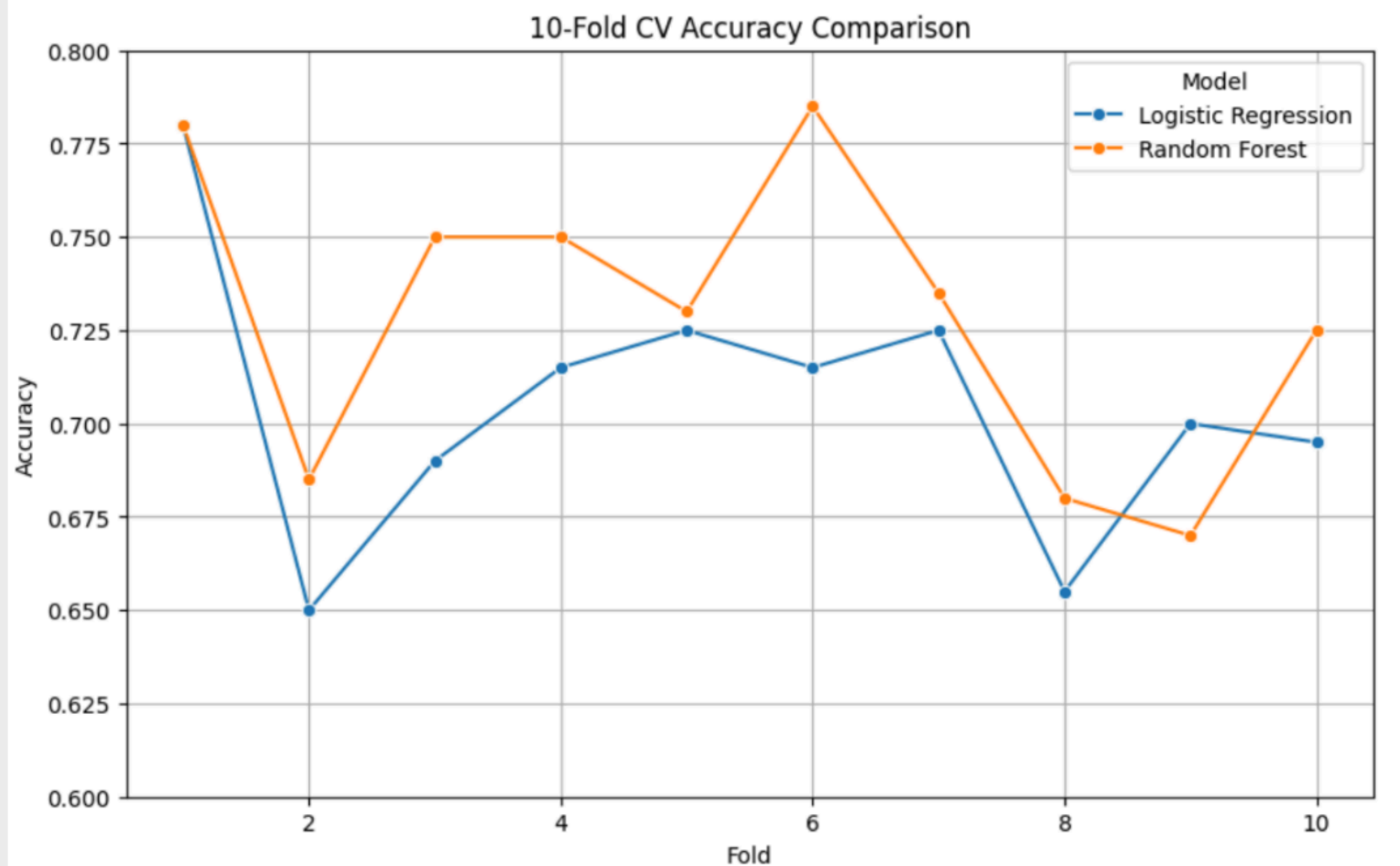
Logistic Regression Test Performance				
	precision	recall	f1-score	support
0	0.72	0.77	0.74	132
1	0.83	0.77	0.80	142
2	0.57	0.57	0.57	126
accuracy			0.71	400
macro avg	0.71	0.70	0.70	400
weighted avg	0.71	0.71	0.71	400

KNN Classifier Test Performance				
	precision	recall	f1-score	support
0	0.69	0.74	0.72	132
1	0.75	0.81	0.78	142
2	0.55	0.45	0.50	126
accuracy			0.68	400
macro avg	0.66	0.67	0.66	400
weighted avg	0.67	0.68	0.67	400

Decision Tree Test Performance				
	precision	recall	f1-score	support
0	0.71	0.75	0.73	132
1	0.80	0.75	0.78	142
2	0.59	0.60	0.59	126
accuracy			0.70	400
macro avg	0.70	0.70	0.70	400
weighted avg	0.71	0.70	0.70	400

Random Forest Test Performance				
	precision	recall	f1-score	support
0	0.72	0.80	0.76	132
1	0.82	0.81	0.81	142
2	0.62	0.55	0.58	126
accuracy			0.72	400
macro avg	0.72	0.72	0.72	400
weighted avg	0.72	0.72	0.72	400

10-Fold CV Accuracy Comparison



✓ Random Forest:

- Accuracy (Test): **0.72**
- Cross-validation accuracy ranged between **67%** and **78%**, which is consistent with test performance.

✓ Logistic Regression:

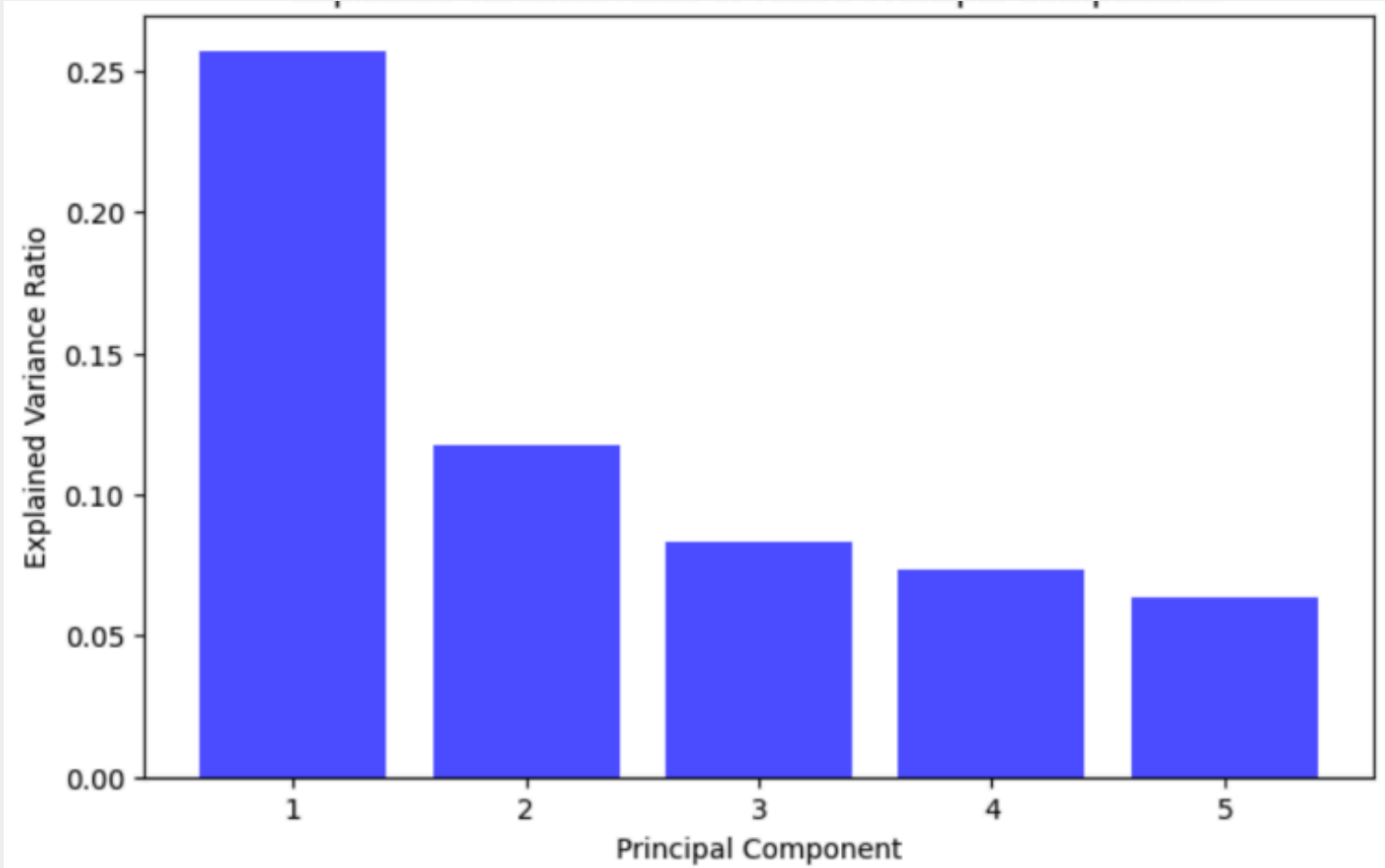
- Accuracy (Test): **0.70**
- Cross-validation accuracy ranged between 64% and 76%, showing noticeable fluctuations.

• Logistic Regression 10-Fold CV Accuracy Scores: [0.78 0.65 0.69 0.715 0.725 0.715 0.725 0.655 0.7 0.695]
Average Accuracy: 0.705

• Random Forest 10-Fold CV Accuracy Scores: [0.78 0.685 0.75 0.75 0.73 0.785 0.735 0.68 0.67 0.725]
Average Accuracy: 0.729

PCA:

We prepared our dataset for performing Principal Component Analysis (PCA) by applying feature removal, price transformation, standard scaling, one-hot encoding, and under-sampling. Then, we examined the first 5 principal components.



	PC1
Rooms	0,49
Bedroom2	0,48
BuildingArea	0,44
Bathroom	0,42
EVR	0,26

	PC2
Lattitude	0,56
Longtitude	-0,51
Distance	0,41
YearBuilt	0,29
EVR	0,12

	PC3
YearBuilt	0,68
Distance	0,47
Lattitude	-0,34
Longtitude	0,31
EVR	0,08

	PC4
Landsize	0,93
Lattitude	-0,20
Car	0,19
YearBuilt	-0,10
EVR	0,07

	PC5
Car	0,49
Longtitude	-0,45
Lattitude	-0,43
Landsize	-0,29
EVR	0,06

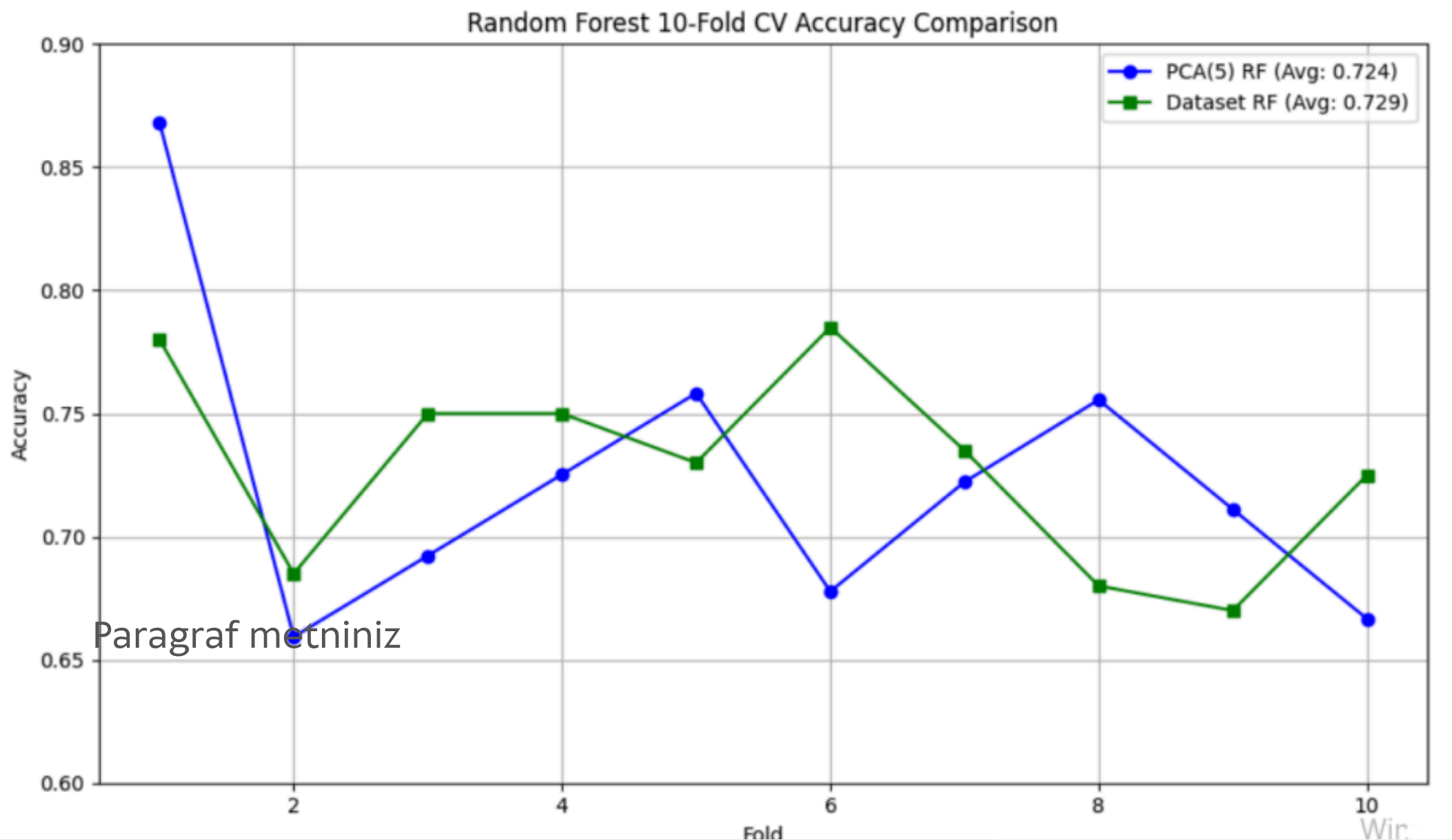
- PC1: 0.257
- PC2: 0.118
- PC3: 0.083
- PC4: 0.073
- PC5: 0.063

Total explained variance by first 5 components:
0.593

Model Comparison:

Then we compared The first 5 principal component Random Forest Model with The Random Forest models based by prepared dataset.

Although PCA components reduced dimensionality effectively, model comparison showed that using the original balanced dataset led to more reliable and consistent classification results, especially for the "Affordable" class.



By Balanced Dataset

	precision	recall	f1-score	support
expensive -> 0	0.72	0.80	0.76	132
cheap -> 1	0.82	0.81	0.81	142
affordable -> 2	0.62	0.55	0.58	126
accuracy			0.72	400
macro avg	0.72	0.72	0.72	400
weighted avg	0.72	0.72	0.72	400

By the first 5 principal components.

	precision	recall	f1-score	support
Affordable	0.59	0.52	0.55	295
Cheap	0.79	0.85	0.82	315
Expensive	0.77	0.79	0.78	295
accuracy			0.72	905
macro avg	0.71	0.72	0.72	905
weighted avg	0.72	0.72	0.72	905