Case and Data Description

The main objective is to build and experiment with regression pipelines in Azure Machine Learning Studio.

The task is to create a predictive model in Azure ML Studio to predict the price of a house based on its features using different regression techniques. The supervised dataset contains seven variables:

- House Sale Identifier – Categorical
- House Age - Numeric
- SqFt - Numeric
- Num Bathroom - Numeric
- Num Bedrooms - Numeric
- Average Income - Numeric
- Sales Price – Numeric (Response Variable)

Deploying different regression techniques such as decision tree regression, decision forest regression, and linear regression can provide valuable insights into predicting the price of a house based on its features. Accurate predictions can be provided for individuals or businesses in the real estate industry by evaluating the performance of each model and selecting the best one.

Approach

Step 1: Exploratory Data Analysis.

Using Summarize Data component in Azure ML pipeline design we can conduct a preliminary EDA. The Result Dataset from the Summarize Data component shows details such as count, number missing, sample skewness, min, max, mean, frequency, etc. The preliminary EDA identifies that the House Age and Average Income column have 80 and 20 missing values respectively. Five columns showed sample skewness above the acceptable range of 0.5.

Column Name	Sample Skewness	Proposed Transformation
Number of bedrooms	0.63	sqrt(x)
Sqft	0.8066	sqrt(x)
Average Income	0.9	sqrt(x)
Sales Price	2.67	ln(x)
Number of Bathrooms	10.8	log10(x)

Step 2: Data Cleaning and Preparation.

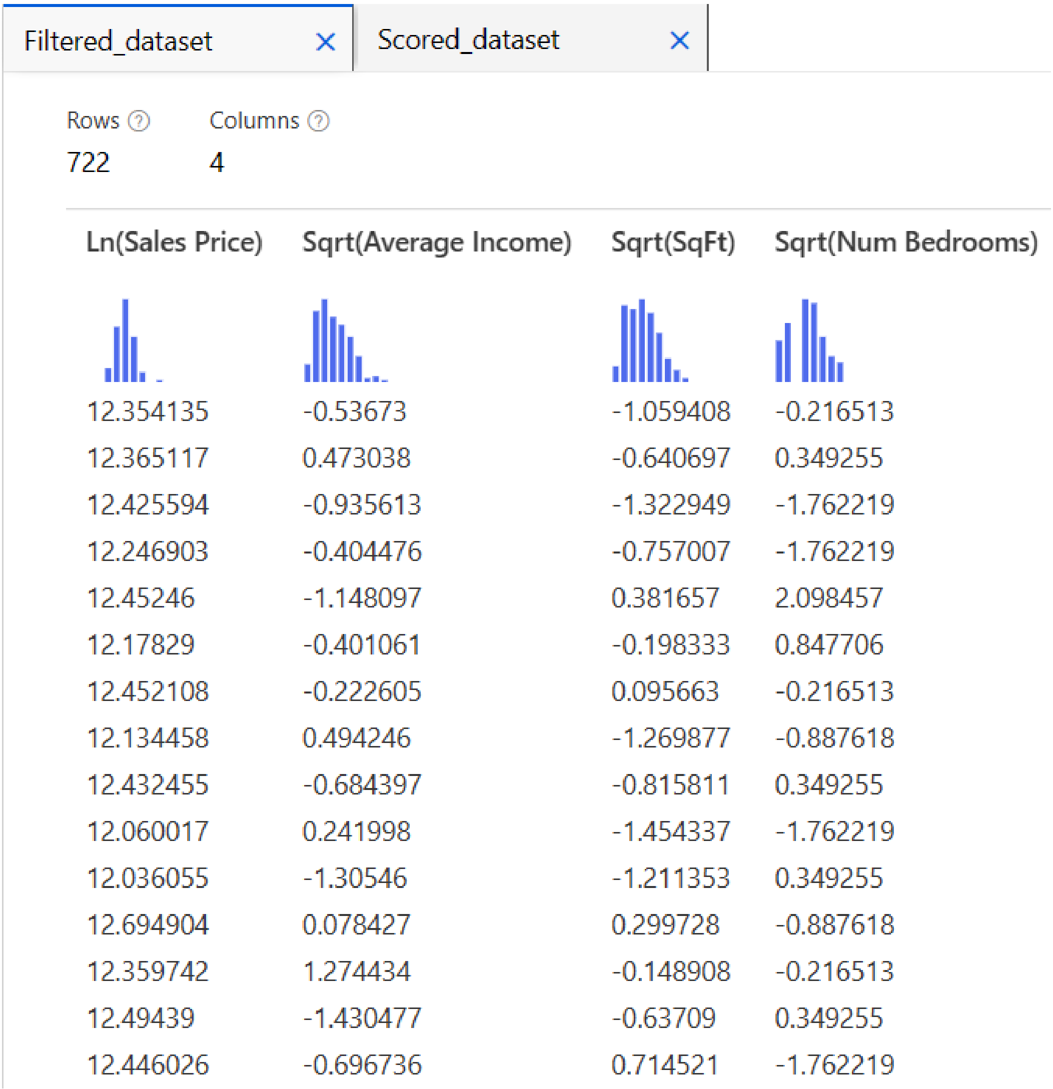
A sample skewness of 10.5 is peculiarly interesting, so upon further investigation, it is relieved that the Number of bathrooms column has 10 rows have values >200 while the rest have values ranging from 1-12. These outliers might be a typo (200 is entered in place of 2). So before applying column transformations to deal with the sample skewness, these outliers should be dealt with. With the intention to keep the pipeline simple, these outliers are marked missing using Clip_Values component.

A common way to deal with missing values in numeric columns is to replace the missing values with the mean of the column. But doing so before splitting the data into training and testing will cause information leakage into the training set. This information leakage happens as the mean is generated using the values from testing data set and this influence the prediction model. Hence to avoid this information leakage and have a simpler pipeline the rows with the missing values are removed using the Clean_Missing_Data component. The outliers (Number of bathrooms > 12) were removed, this reduced the sample skewness of the column to 0.64. To reduce the skewness, Square root transformation was applied on columns with Skewness between 0.5 - 1 and ln function transformation was applied on columns with Skewness >1 using the Apply_Math_Operations component.

Step 3: Feature Selection and Splitting the data.

Using Apply_Math_Operations creates a new column with the transformed data. Hence, we need to remove the original rows to avoid correlation. The Select_Coulmns_In_Dataset component allows to select the required columns. After removing the skewed columns, the dataset can be split into training and testing with a 80:20 ratio. The training dataset is then normalized to bring the numeric columns to the same scale using Normalize_data component. To apply the same normalization transformation on the testing dataset, the Apply_Transformation component is used.

When all attributes are utilized in a model, it often results in poor generalization performance on unseen data. To avoid this, Filter_Based_Feature_Selection is used to select the most important features using Pearson correlation on the transformed response variable, Sales Price. This columns transformation can be transferred to the testing dataset using the Select_Columns_Transform and the Apply_Transform component. Filter based feature selection selected the transformed SqFt, Num Bedrooms, and Average Income as the three most important response variables.



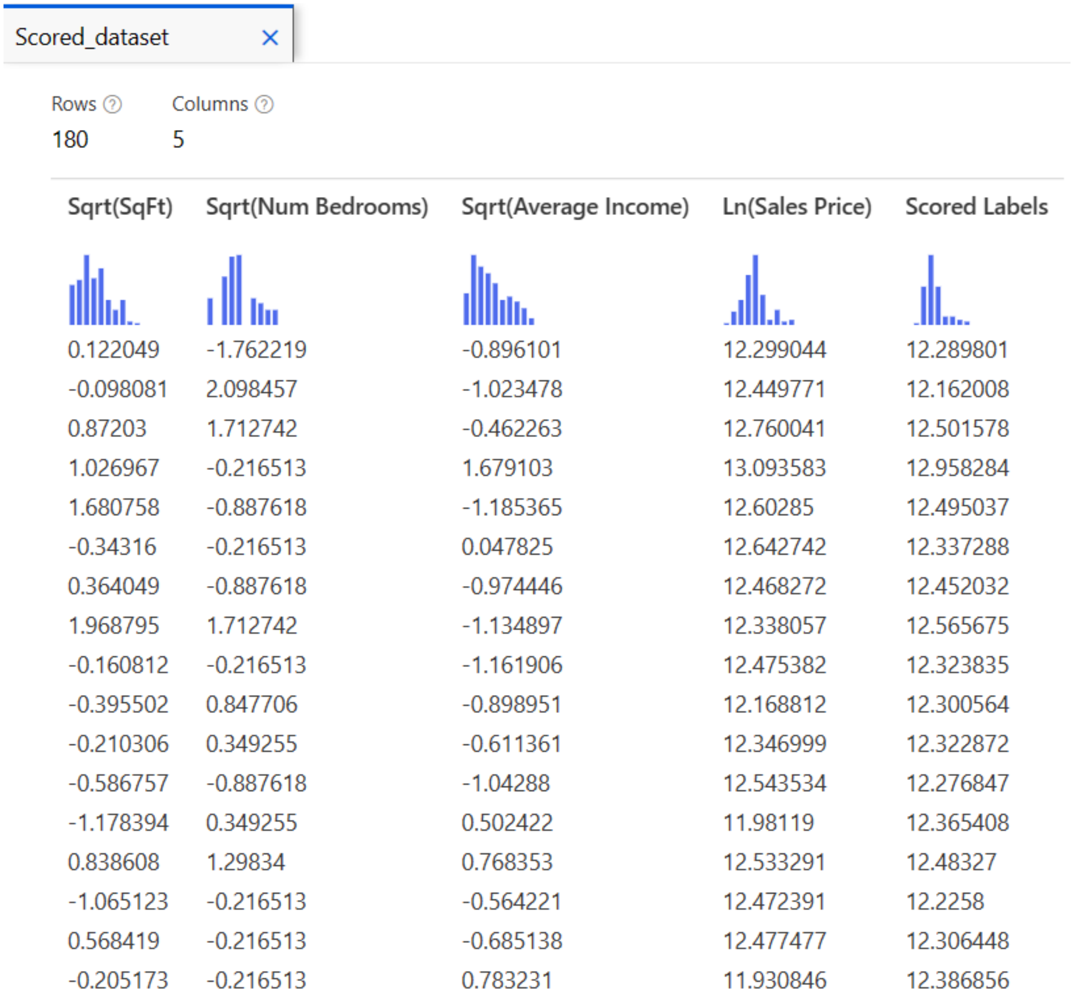
Rows ⓘ 722

Columns ⓘ 4

Ln(Sales Price)	Sqrt(Average Income)	Sqrt(SqFt)	Sqrt(Num Bedrooms)
12.354135	-0.53673	-1.059408	-0.216513
12.365117	0.473038	-0.640697	0.349255
12.425594	-0.935613	-1.322949	-1.762219
12.246903	-0.404476	-0.757007	-1.762219
12.45246	-1.148097	0.381657	2.098457
12.17829	-0.401061	-0.198333	0.847706
12.452108	-0.222605	0.095663	-0.216513
12.134458	0.494246	-1.269877	-0.887618
12.432455	-0.684397	-0.815811	0.349255
12.060017	0.241998	-1.454337	-1.762219
12.036055	-1.30546	-1.211353	0.349255
12.694904	0.078427	0.299728	-0.887618
12.359742	1.274434	-0.148908	-0.216513
12.49439	-1.430477	-0.63709	0.349255
12.446026	-0.696736	0.714521	-1.762219

Step 4: Regression Modeling

Next, decision tree regression, decision forest regression, and linear regression models are experimented to predict the price of a house. Decision tree regression is a machine learning algorithm that uses a decision tree to model the relationship between the independent and dependent variables. Decision forest regression, on the other hand, uses multiple decision trees to model the relationship between the variables. Linear regression, meanwhile, models the linear relationship between the independent and dependent variables. Each model component is followed by Train_Model, Score_model and Evaluate_Model component respectively. (Please refer to the pipeline image for more information)



Scored_dataset ×

Rows 180

Columns 5

Sqrt(SqFt)	Sqrt(Num Bedrooms)	Sqrt(Average Income)	Ln(Sales Price)	Scored Labels
0.122049	-1.762219	-0.896101	12.299044	12.289801
-0.098081	2.098457	-1.023478	12.449771	12.162008
0.87203	1.712742	-0.462263	12.760041	12.501578
1.026967	-0.216513	1.679103	13.093583	12.958284
1.680758	-0.887618	-1.185365	12.60285	12.495037
-0.34316	-0.216513	0.047825	12.642742	12.337288
0.364049	-0.887618	-0.974446	12.468272	12.452032
1.968795	1.712742	-1.134897	12.338057	12.565675
-0.160812	-0.216513	-1.161906	12.475382	12.323835
-0.395502	0.847706	-0.898951	12.168812	12.300564
-0.210306	0.349255	-0.611361	12.346999	12.322872
-0.586757	-0.887618	-1.04288	12.543534	12.276847
-1.178394	0.349255	0.502422	11.98119	12.365408
0.838608	1.29834	0.768353	12.533291	12.48327
-1.065123	-0.216513	-0.564221	12.472391	12.2258
0.568419	-0.216513	-0.685138	12.477477	12.306448
-0.205173	-0.216513	0.783231	11.930846	12.386856

Step 5: Results and Evaluation.

Each model is evaluated using metrics such as mean squared error, R-squared, and root mean squared error. These measures can be obtained from the evaluate results option in the Evaluate_Model component. The evaluation results are shown below.

Model	Mean Absolute Error (MAE)	Root Mean Squared Error (RMSE):	Relative Squared Error (RSE):	Relative Absolute Error (RAE):	Coefficient of Determination (R-squared)
Linear Regression	0.1704	0.2229	0.492	0.7356	0.5073
Boosted Decision Tree	0.1709	0.2206	0.4821	0.7377	0.517
Decision Forest Regression	0.1446	0.1852	0.3399	0.624	0.66

Mean Absolute Error (MAE): MAE measures the average magnitude of the errors in the prediction. A lower MAE indicates better performance.

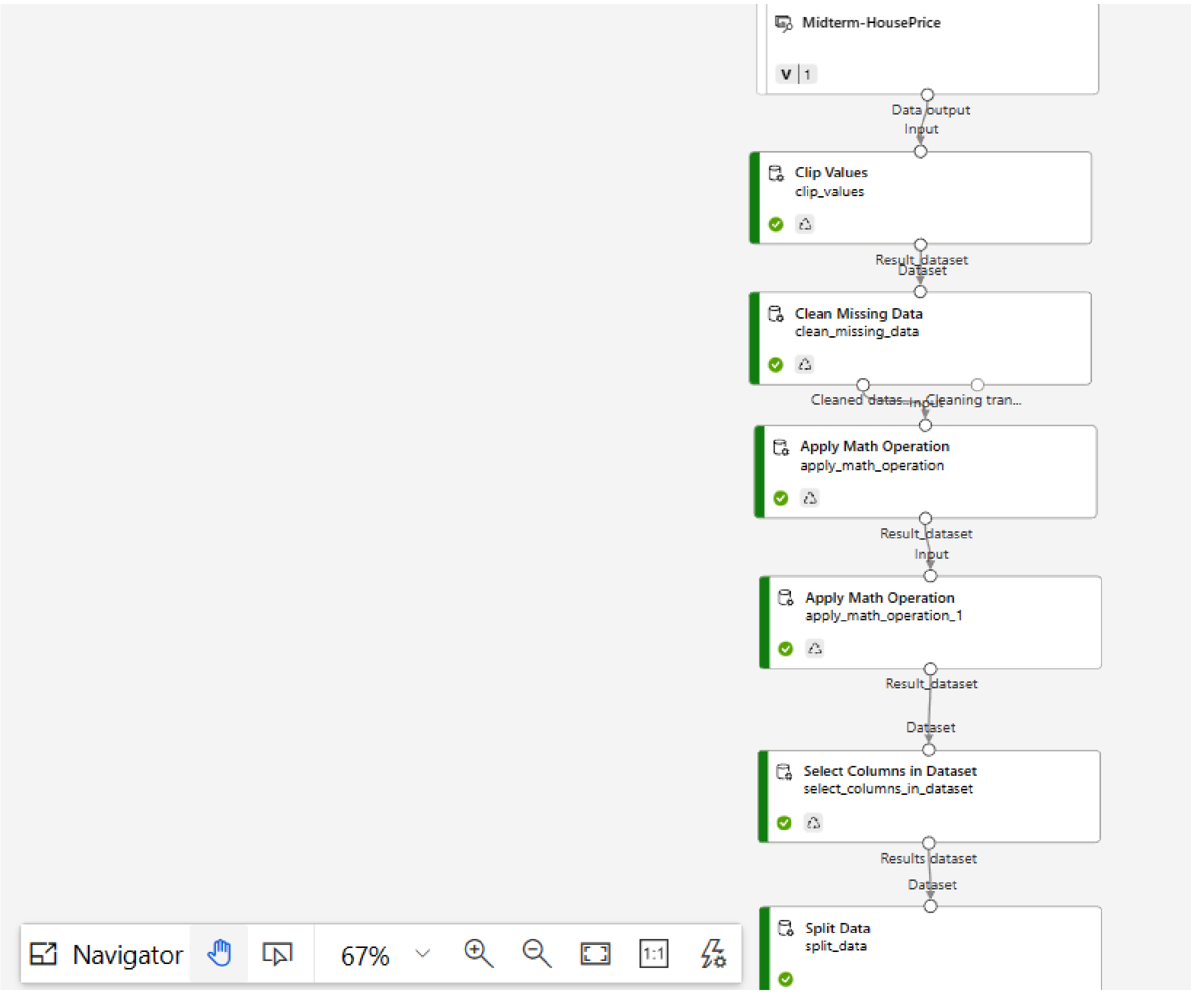
Root Mean Squared Error (RMSE): RMSE measures the square root of the average of the squared errors in the prediction. RMSE is more sensitive to large errors compared to MAE. A lower RMSE indicates better performance.

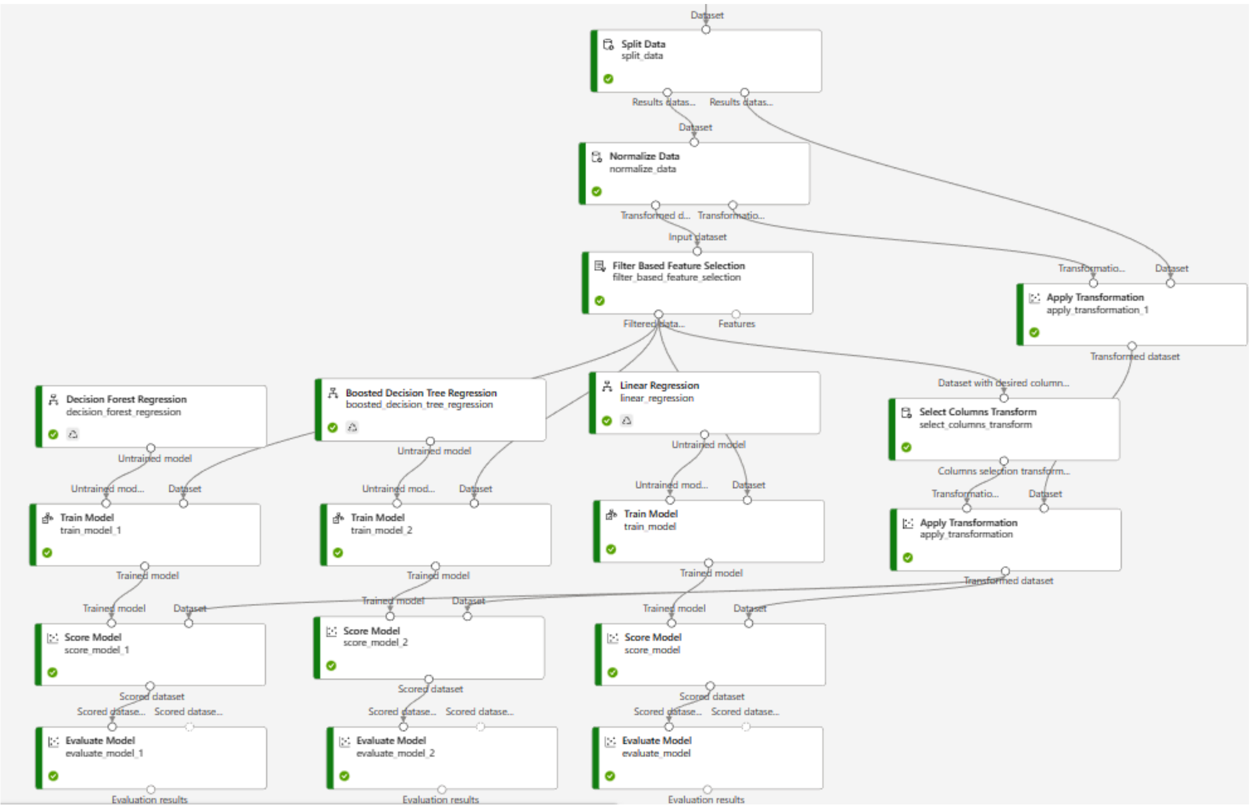
Relative Squared Error (RSE): RSE measures the proportion of the total variation in the response variable that is explained by the model. A lower RSE indicates better performance.

Relative Absolute Error (RAE): RAE measures the proportion of the total variation in the response variable that is not explained by the model. A lower RAE indicates better performance.

Coefficient of Determination (R-squared): R-squared measures the proportion of the total variation in the response variable that is explained by the model. A higher R-squared indicates better performance.

Final Model





Version Control

This project uses the student version of Azure Machine Learning Platform 2023.

Annexure

Normalization is not required for Decision Forest Regression and Decision Tree Regression, however, applying normalization gave better results and a cleaner pipeline.