

BSD 3101: PRINCIPLES OF DATA SCIENCE LAB WORK.

Lab 1 Documentation: Multiple Linear Regression Model on Boston Housing Dataset

Group Members

- 1. Lucy Gathoni Mugo 21/05185**
- 2. Bramwel Wanyoike 21/05119**
- 3. Kennedy Kamau 21/08536**
- 4. Newton Maina 21/04800**
- 5. Dennis Ooki 21/05691**
- 6. James Maina 21/08481**
- 7. Abigail Vivian 21/05257**
- 8. Joy Precious 21/05215**
- 9. Ahmed Fatma 21/07234**
- 10. Fabian Ndung'u 21/04883**

1. Introduction

This document outlines the steps followed to create and validate a multiple linear regression model to predict median home values (MEDV) based on various predictors from the Boston Housing dataset using RapidMiner Studio. The dataset contains 506 instances and 14 features, with MEDV being the target variable.

2. Data Preparation and Model Building

- **Step 1: Importing the Data**

The Boston Housing dataset was retrieved from the companion website using the *Retrieve Operator*. This dataset contains 14 features, including crime rate (CRIM), number of rooms (RM), and others as shown below:

Features:

- CRIM (per capita crime rate by town)
- ZN (proportion of residential land zoned for large lots)
- INDUS (proportion of non-retail business acres per town)
- CHAS (Charles River dummy variable)
- NOX (nitric oxide concentration)
- RM (average number of rooms per dwelling)
- AGE (proportion of owner-occupied units built before 1940)
- DIS (distance to employment centers)
- RAD (index of accessibility to radial highways)
- TAX (full-value property tax rate)
- PTRATIO (pupil-teacher ratio)
- B (proportion of Black population by town)

- LSTAT (lower status of the population)
- MEDV (median value of owner-occupied homes, the target variable)

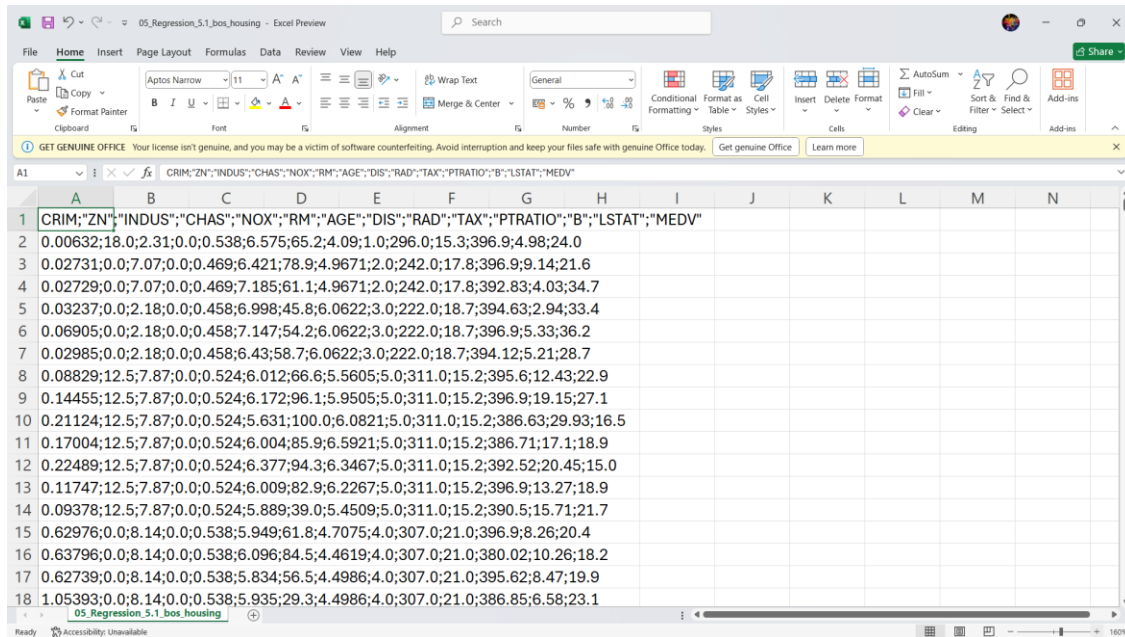


Figure 1: Boston Housing Dataset

• Step 2: Data Shuffling and Splitting the Data

The *Shuffle Operator* was used to randomize the data to ensure that both the training and test sets are statistically similar.

The data was split into a **training set** (rows 1–450) and a **test set** (rows 451–506) using the *Filter Examples Range Operator*.

- Training set: 450 instances
- Test set: 56 instances

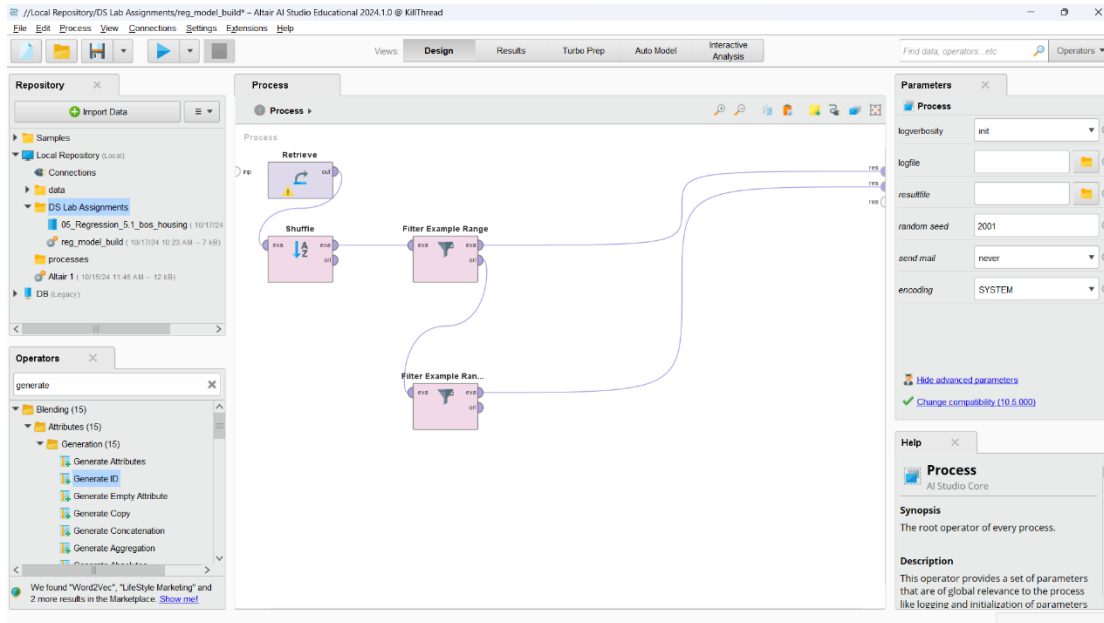


Figure 2: Shuffling and Splitting Data

- **Step 3: Setting the Target Variable and Split Validation**

Using the *Set Role Operator*, the target variable MEDV was assigned the "label" role. A *Split Validation Operator* was used to further split the training set into 70% training and 30% validation sets for cross-validation. The *relative* split setting was used, with a random seed of 1992 to ensure reproducibility.

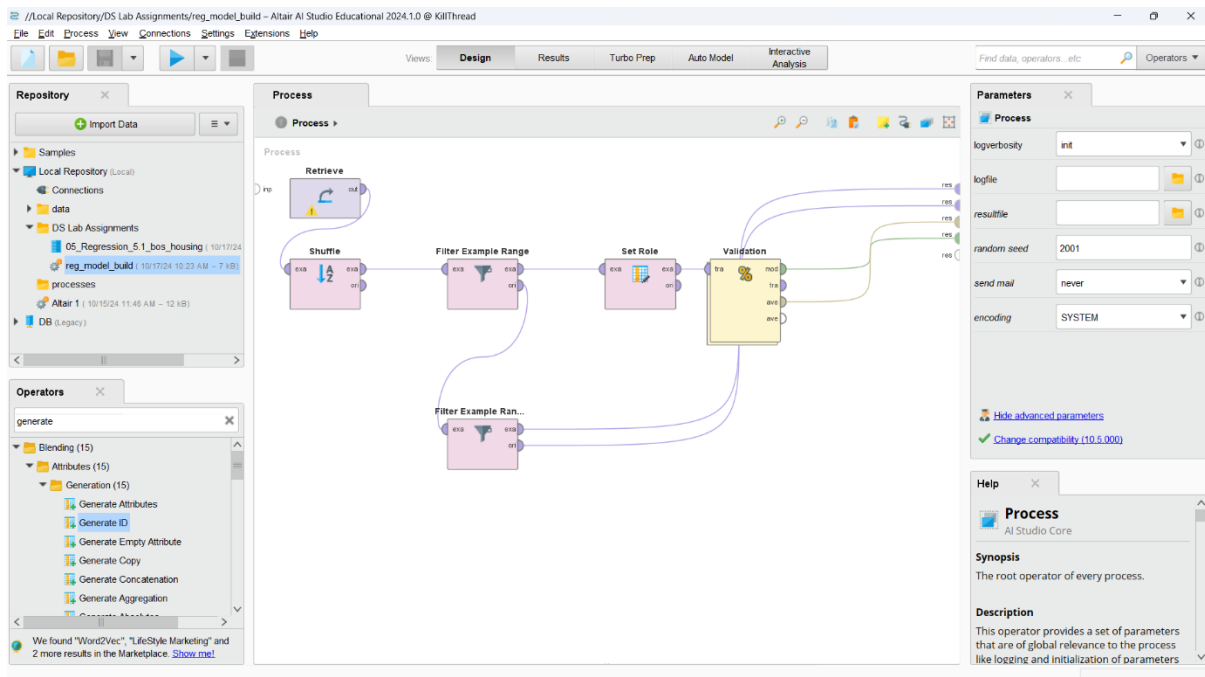


Figure 3: Target variable and Split Validation Ops

- **Step 4: Model Training**

A **Linear Regression Operator** was applied to the training data. The model learned the relationship between MEDV and the other features in the dataset. In the nested layout of the Split Validation Operator, the following operators were added:

- **Performance (Regression):** This operator evaluates the model's performance on both training and validation datasets.
 - **Parameters:** Adjusted to calculate RMSE (Root Mean Square Error) and R^2 (coefficient of determination), correlation, and squared correlation.
- **Linear Regression:** The core operator that builds the regression model.
 - **Parameters:** Default settings were used, but the model was tested by adjusting through selecting relevant features.

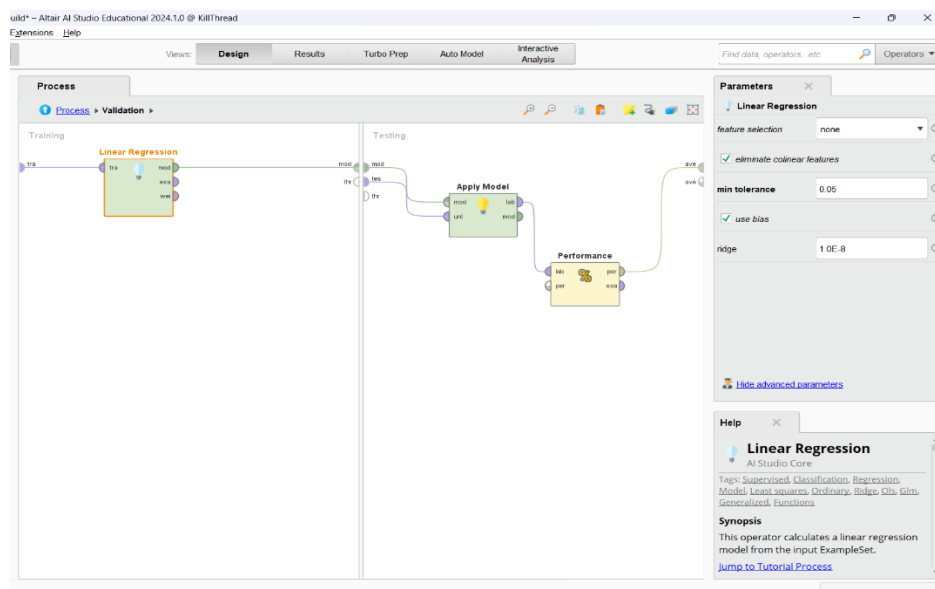


Figure 4: Linear Reg Ops Parameter Modification

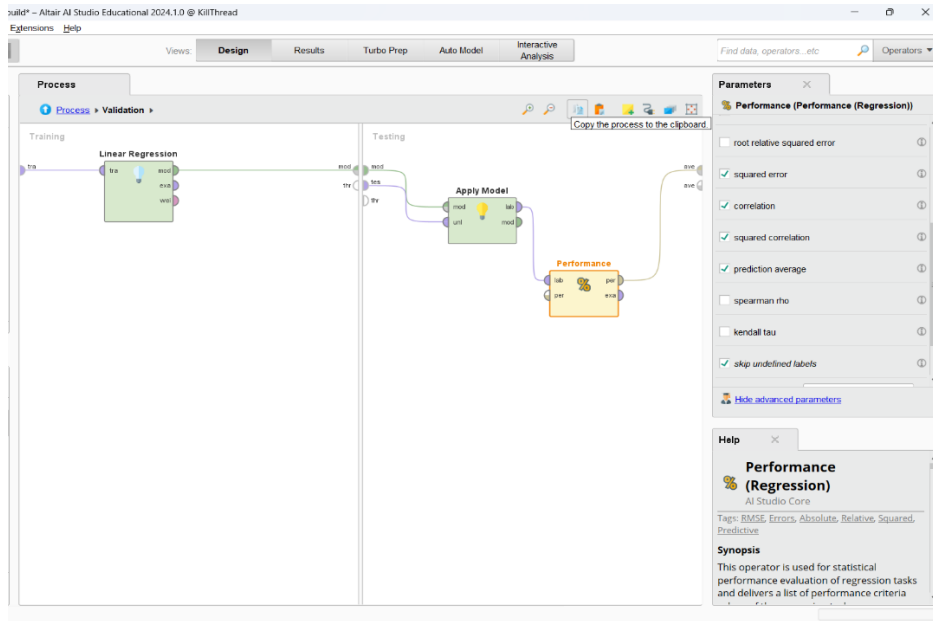


Figure 5: Performance Regression Ops Parameter Modification

- **Model Execution:** Running the model provided two default views namely:
 - Description View (Fig 6)
 - Data view (Fig 7 and 8)

A comparison was also done on the parameters in the Linear regression operator: feature change from none to a greedy parameter which emitted least significant factors being INDUS and AGE.

A performance tabulation is also provided below for the trained model whose **correlation was at 0.891**

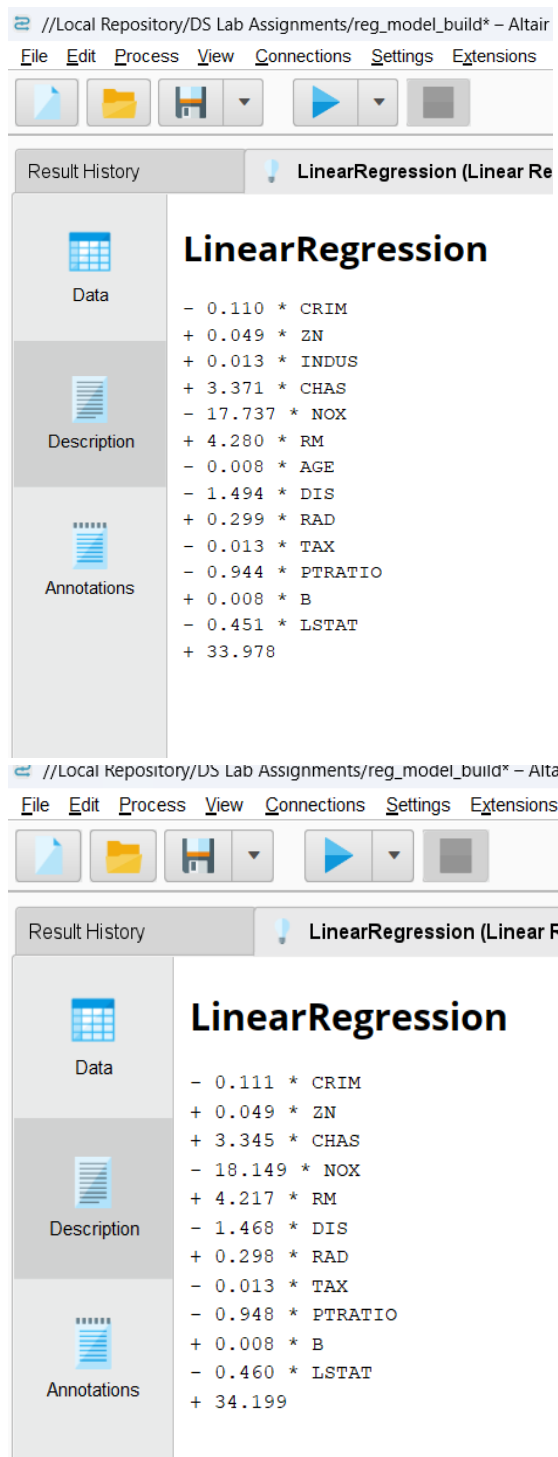


Figure 7 and 8 : Model with no feature and greedy feature comparison

Using All Features (No Feature Selection):

- Performance Metrics:**
 The model's performance was assessed with all features included, demonstrating potential overfitting due to irrelevant features.

Using Greedy Feature Selection (Excluding INDUS and AGE):

- Performance Metrics:**
 The model's RMSE and R^2 values were evaluated, showing improved performance metrics compared to the full feature set.

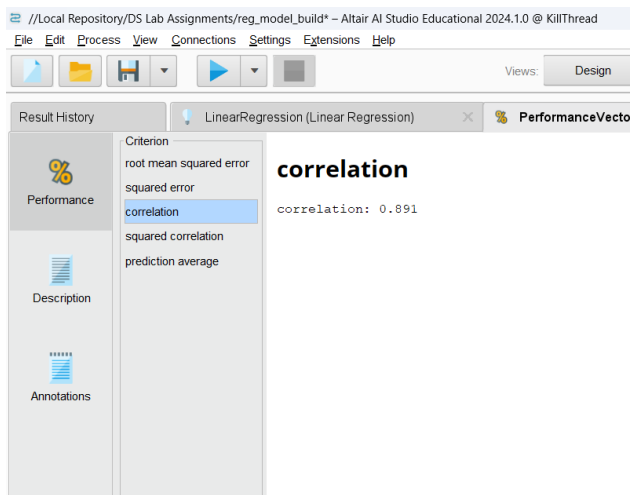


Figure 9: Performance Vector Display

5. Applying the Model to Unseen Test Data

After training and validating the model, it was applied to the unseen test dataset (rows 451–506).

1. Predictions on Test Data:

- The trained model was used to predict MEDV values for the test dataset.

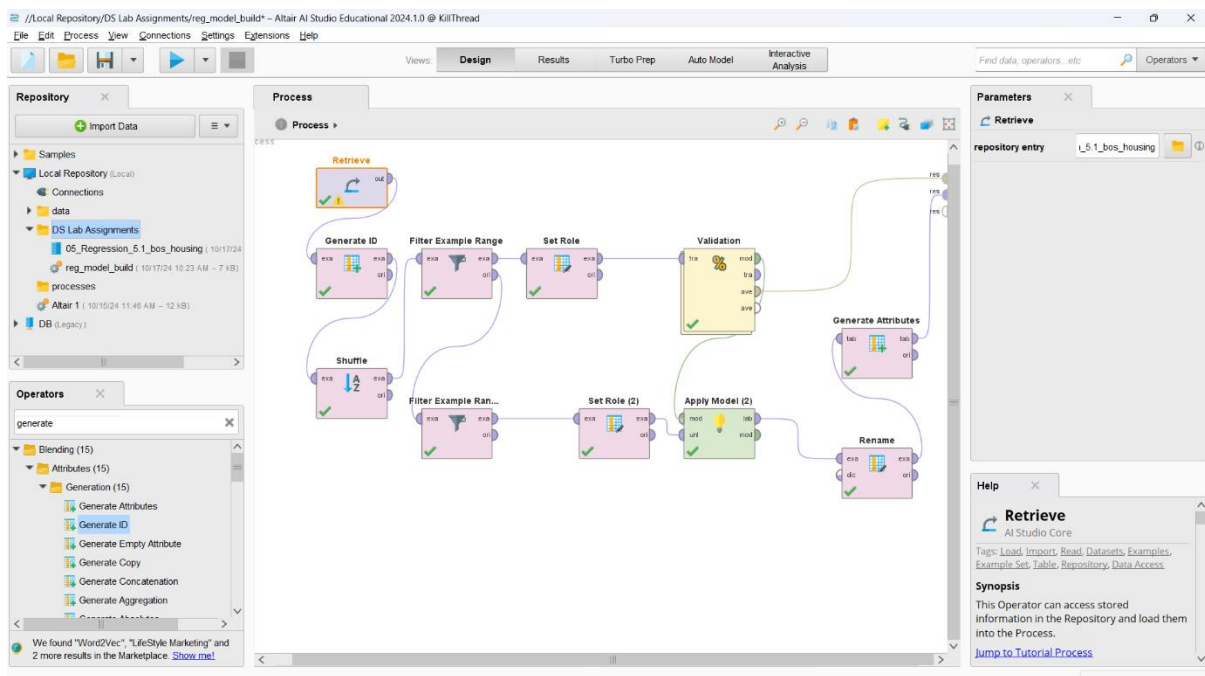


Figure 10: Model Application Design

2. Evaluation of Model Performance on Test Data:

- The **Performance (Regression)** operator was employed to evaluate the predictions, calculating RMSE and R^2 values specific to the test set.

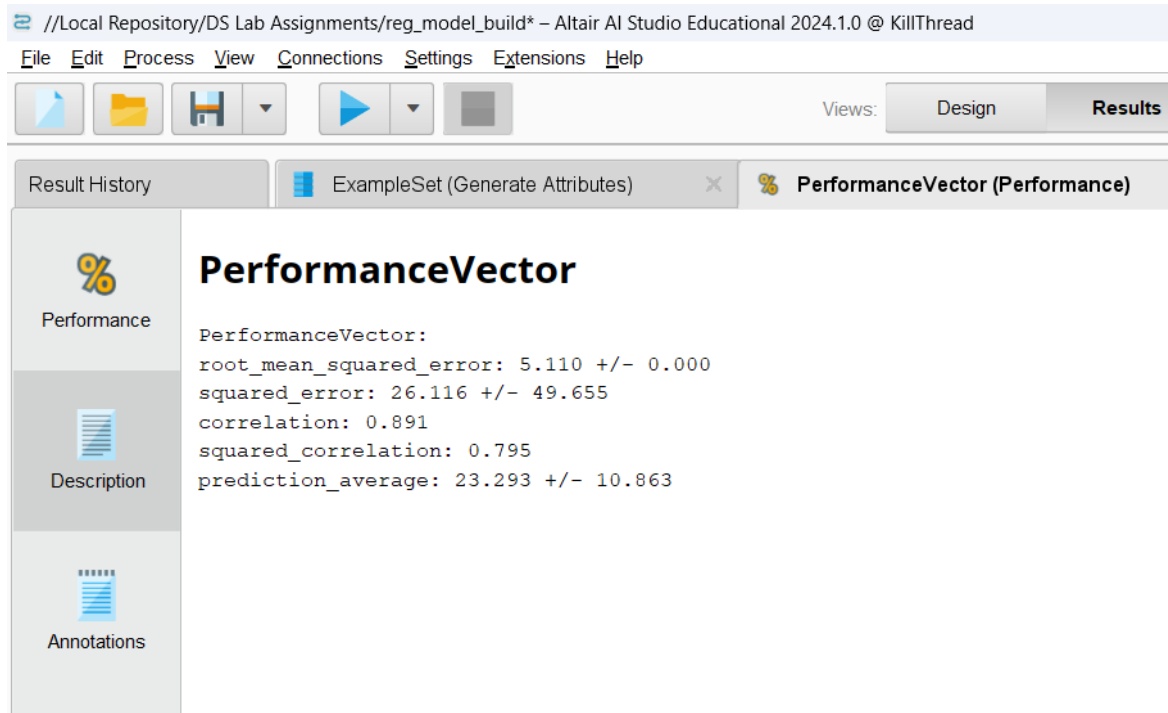


Figure 11: Performance Vector Analysis

Summary of Results on Test Data

- **RMSE:** The error metric for assessing how well the model predicted MEDV on the unseen test data.
- **R^2 :** Provided an indication of how much variance in the test data was explained by the model.

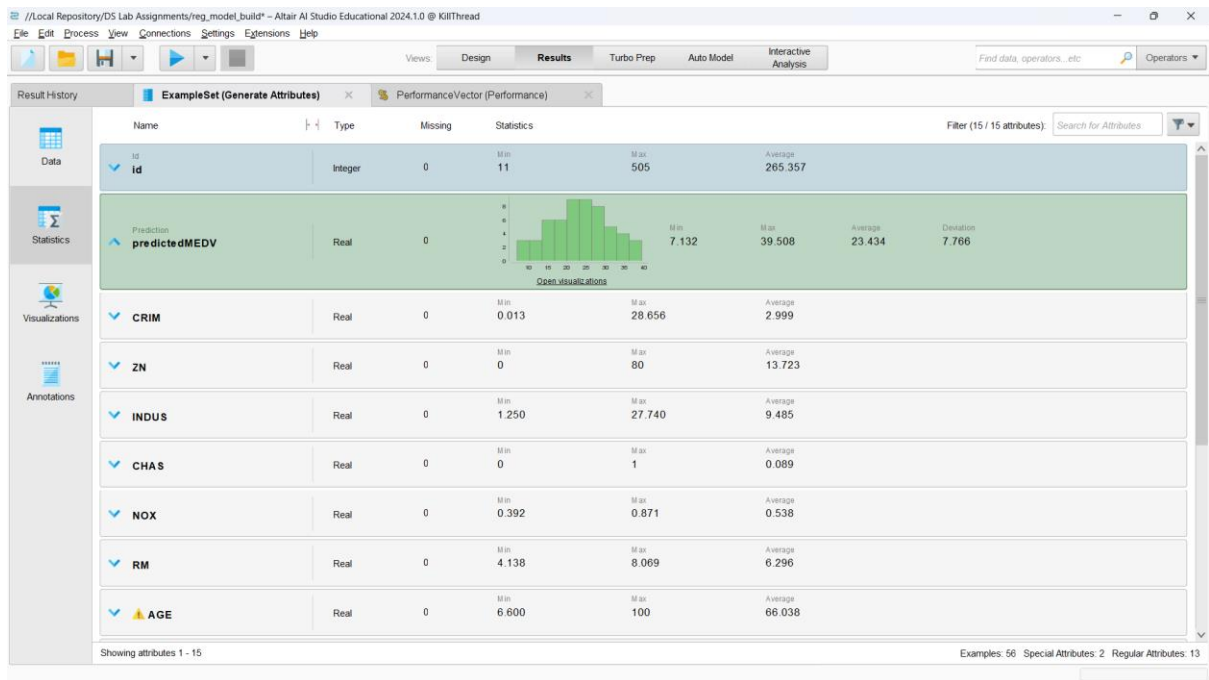


Figure 12: Summary results on test data

Step 6. Visualizations

Three visualizations were created to help interpret the results of the linear regression model applied to the Boston Housing dataset.

- **Visualization 1: Scatter Plot of RM (Average Rooms) vs. Predicted MEDV**

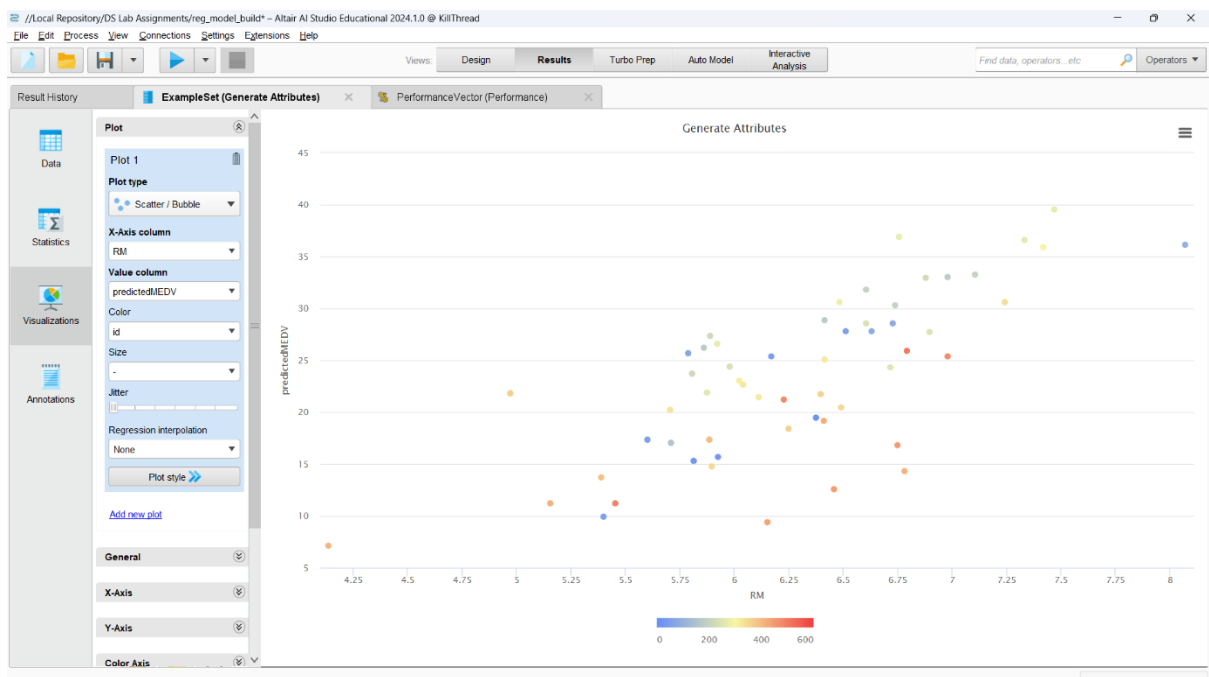


Figure 13: Scatter plot RM vs Predicted MEDV

This scatter plot shows the relationship between the number of rooms per dwelling (RM) and the predicted median home values (PredictedMEDV) from the model.

Insight: As the number of rooms increases, the predicted median home value also tends to increase, indicating a positive correlation between room count and home value.

- **Visualization 2: Histogram of MEDV**

This histogram displays the distribution of the actual MEDV values in the dataset.

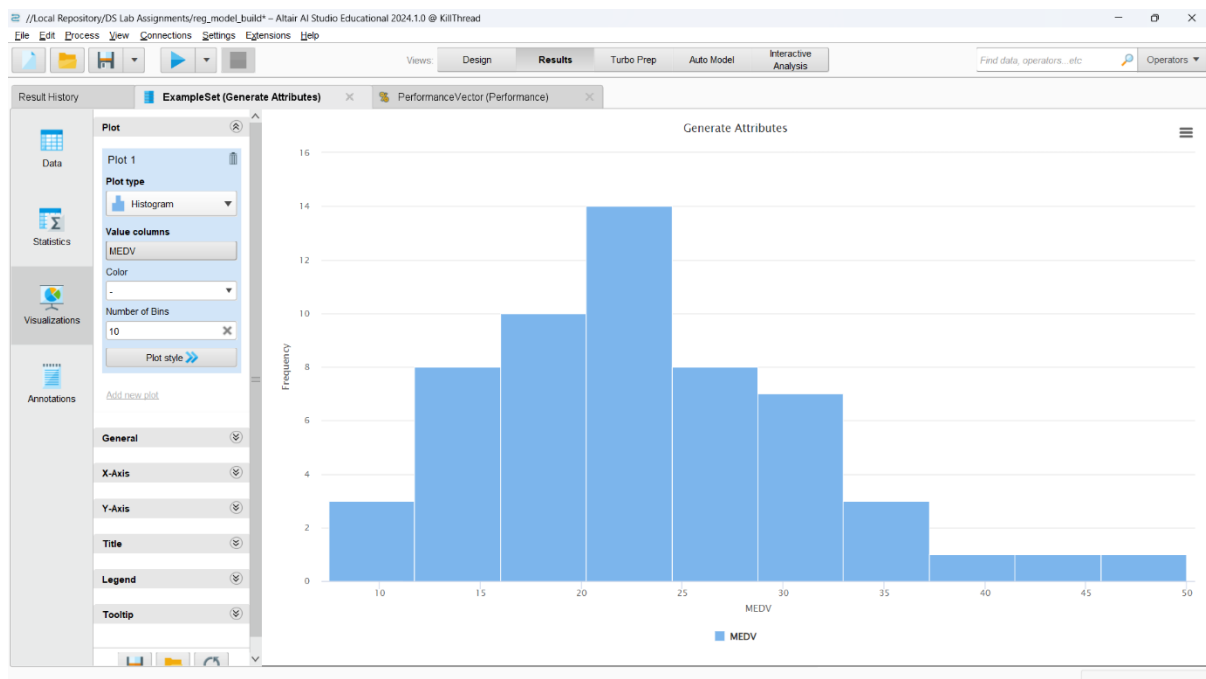


Figure 14: Histogram on Actual MEDV

Insight: The distribution of home values shows a right skew, with a majority of homes valued between \$15,000 and \$25,000. This skewness indicates that most homes in the dataset are moderately priced, but there are some outliers with significantly higher values.

- **Visualization 3: Line Graph Plot (Actual MEDV vs. Predicted MEDV)**

This line plot visualizes the difference between the actual and predicted values of MEDV.

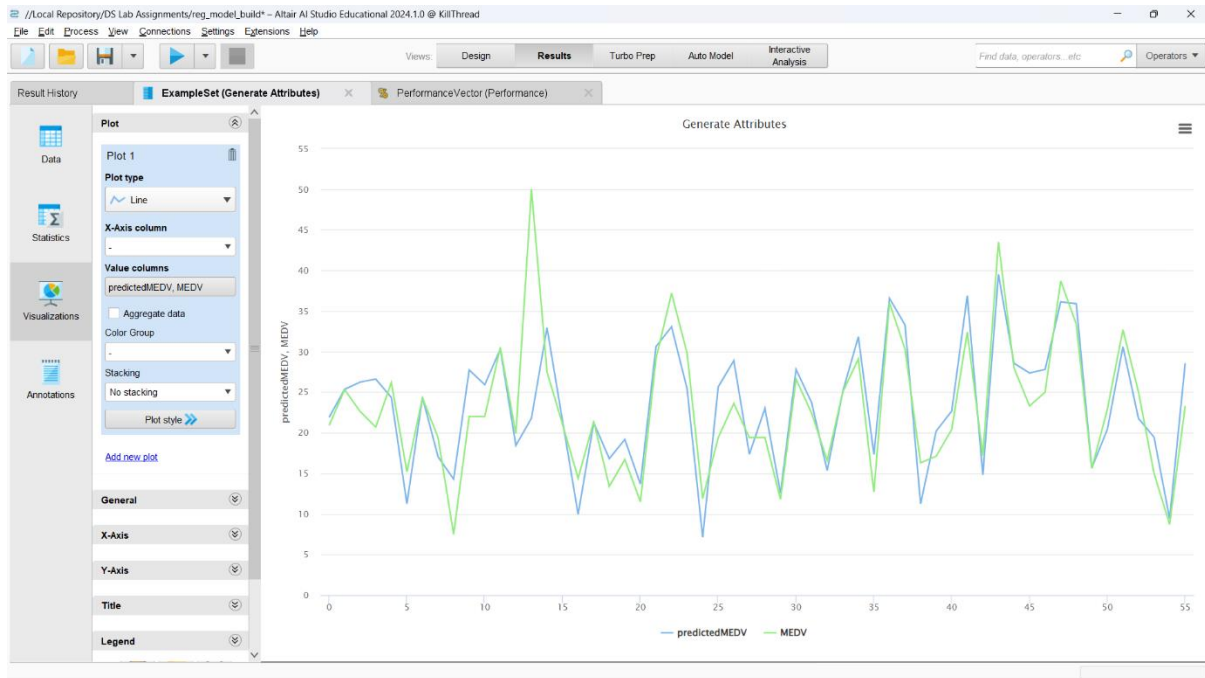


Figure 15: Actual MEDV vs Predicted MEDV

7. Conclusion

The multiple linear regression model was successfully applied to the Boston Housing dataset to predict median home values (MEDV). The model performed reasonably well, as shown by the line graph plot, but some patterns in the line plot suggest that further improvements could be made, potentially by considering non-linear relationships or adding more complex features.

8. Question 3 - Documentation: Linear Regression Model on Salary Prediction

1. Introduction

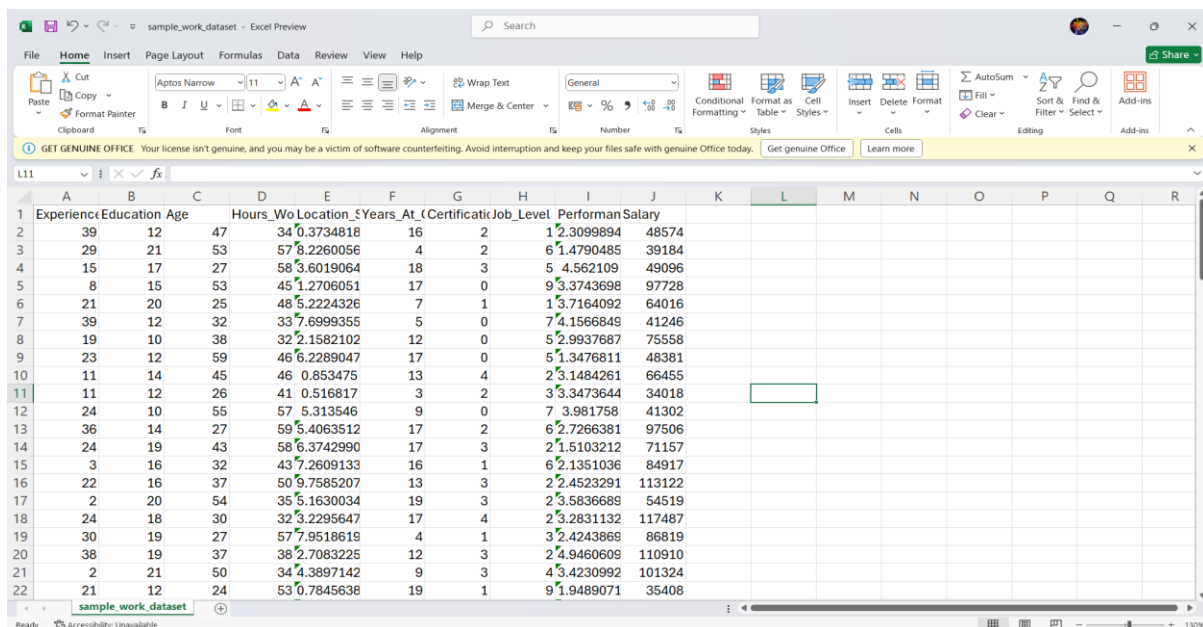
This documentation details the steps taken to build and analyze a linear regression model to predict employee salaries (Salary) based on various factors from a dataset. The dataset contains more than 100 rows and 10 columns. Our goal is to predict the Salary using linear regression and evaluate the model's accuracy.

2. Model Building Steps

2.1 Data Import

The dataset was loaded into RapidMiner using the Retrieve operator. The dataset contains various features such as:

- Experience (Years of experience)
- Age
- Job Role
- Education Level
- Location
- Salary (Target variable)



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	Experience	Education	Age	Hours_We	Location	Years_At	(Certificati	Job_Level	Performan	Salary								
2	39	12	47	34	0.3734818	16	2	1	2.3099894	48574								
3	29	21	53	57	8.2260056	4	2	6	1.4790485	39184								
4	15	17	27	58	3.6019064	18	3	5	4.562109	49096								
5	8	15	53	45	1.2706051	17	0	9	3.3743698	97728								
6	21	20	25	48	5.2224326	7	1	1	3.7164092	64016								
7	39	12	32	33	7.6999355	5	0	7	4.1566849	41246								
8	19	10	38	32	2.1582102	12	0	5	2.9937687	75558								
9	23	12	59	46	6.2289047	17	0	5	1.3476811	48381								
10	11	14	45	46	0.853475	13	4	2	3.1484261	66455								
11	11	12	26	41	0.516817	3	2	3	3.3473644	34018								
12	24	10	55	57	5.313546	9	0	7	3.981758	41302								
13	36	14	27	59	5.4063512	17	2	6	2.7266381	97506								
14	24	19	43	58	6.3742990	17	3	2	1.5103212	71157								
15	3	16	32	43	7.2609133	16	1	6	2.1351036	84917								
16	22	16	37	50	9.7585207	13	3	2	2.4523291	113122								
17	2	20	54	35	5.1630034	19	3	2	3.5836689	54519								
18	24	18	30	32	3.2295647	17	4	2	3.2831132	117487								
19	30	19	27	57	7.9518619	4	1	3	2.4243869	86819								
20	38	19	37	38	2.7083225	12	3	2	4.9460609	110910								
21	2	21	50	34	4.3897142	9	3	4	3.4230992	101324								
22	21	12	24	53	0.7845638	19	1	9	1.9489071	35408								

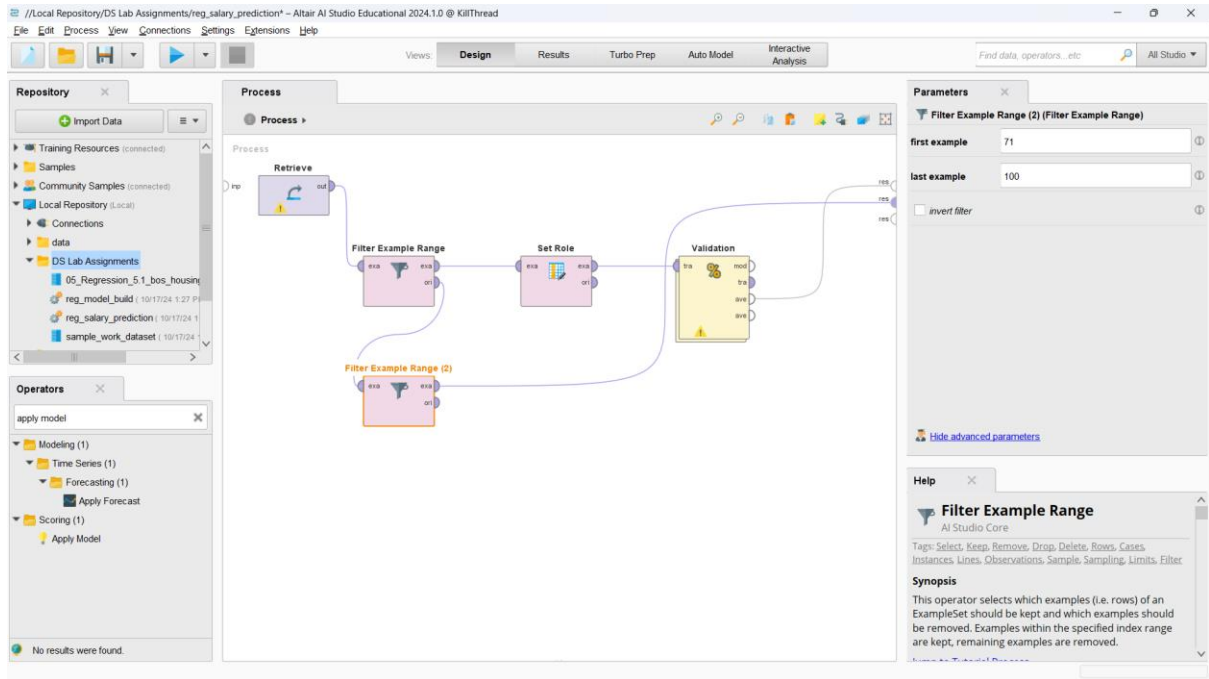
2.2 Set Role for Target (Label)

Using the Set Role operator, the Salary column was assigned the role of **label**. This indicates that the linear regression model will try to predict the values in this column using the other features.

2.3 Splitting the Data

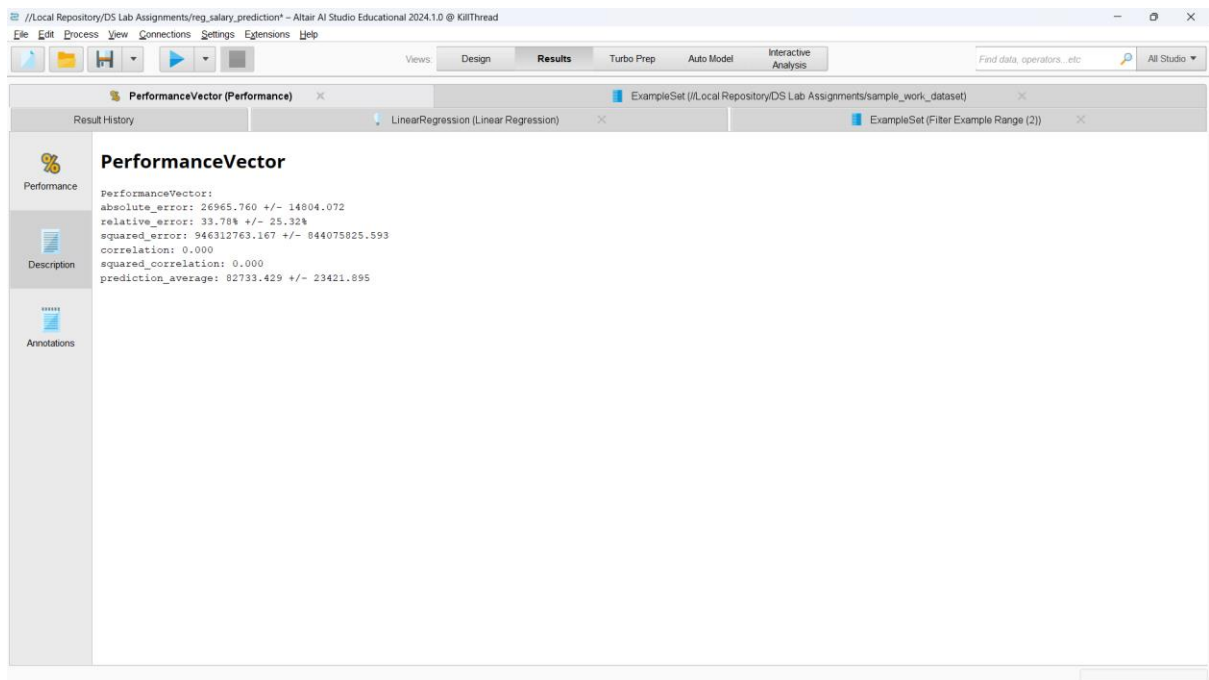
We used two Filter Example Range operators to split the dataset into training and test sets:

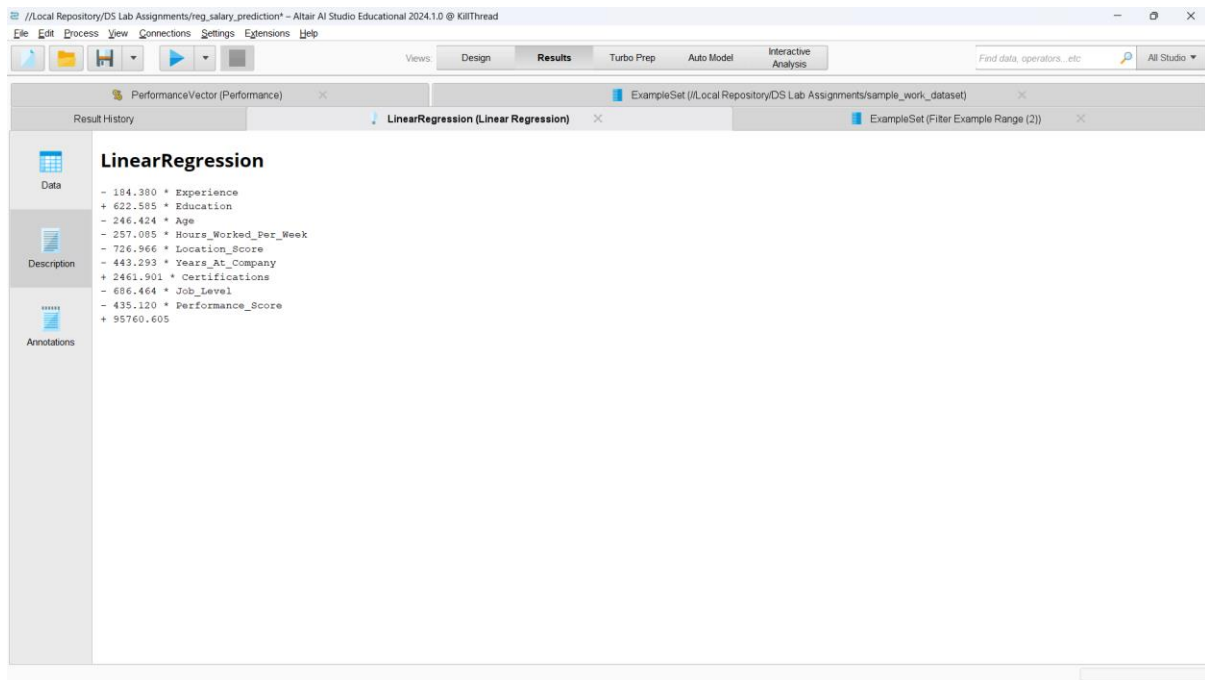
- **Training Set:** The first 70 rows of the dataset were used for model training.
- **Test Set:** Rows 71–100 were used for testing and evaluation.



2.4 Training the Linear Regression Model

The Linear Regression operator was used to train the model on the training data. The goal was to find a relationship between the input features and the target variable (Salary), allowing the model to predict salaries for unseen data.

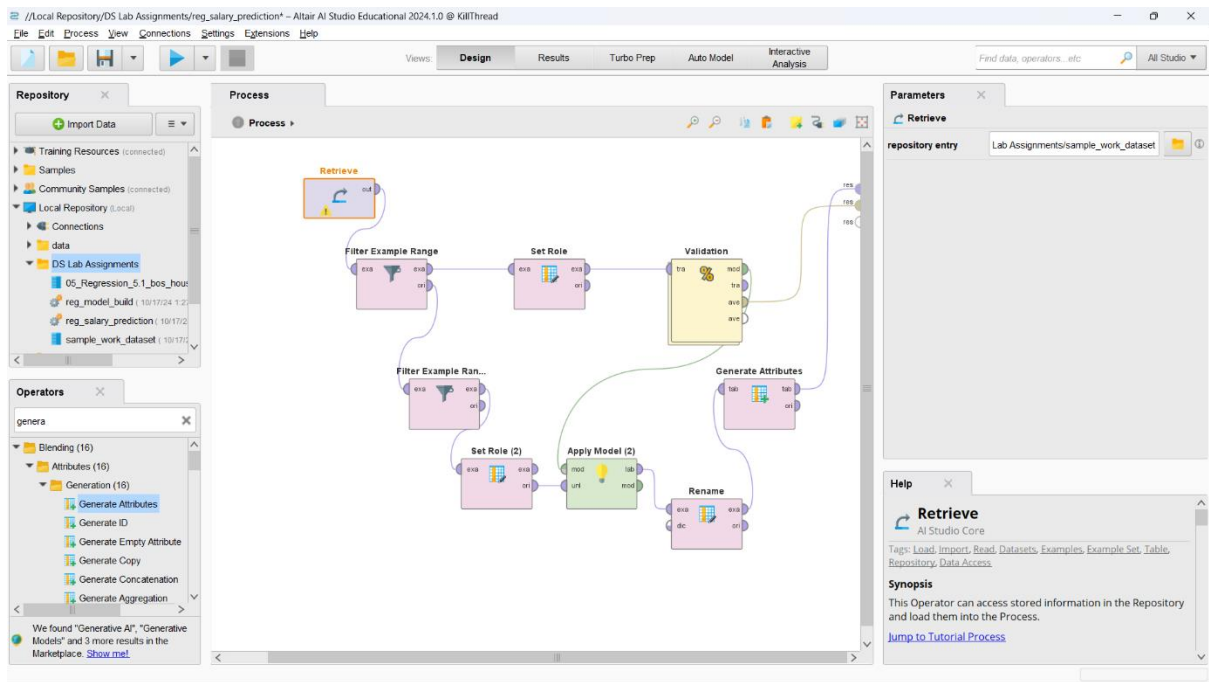




Attribute	Coefficient	Std. Error	Std. Coefficient	Tolerance	t-Stat	p-Value	Code
Experience	-184.380	316.233	-0.074	1.000	-0.583	0.562	
Education	622.585	1037.530	0.081	0.987	0.600	0.551	
Age	-246.424	310.352	-0.102	1.000	-0.794	0.430	
Hours_Worked_Per_Week	-257.085	374.352	-0.088	0.994	-0.687	0.495	
Location_Score	-726.966	1279.658	-0.075	0.999	-0.568	0.572	
Years_At_Company	-443.293	692.914	-0.084	0.999	-0.640	0.525	
Certifications	2461.901	2385.533	0.135	0.985	1.032	0.306	
Job_Level	-686.464	1353.219	-0.066	1.000	-0.507	0.614	
Performance_Score	-435.120	3259.963	-0.018	0.940	-0.133	0.894	
(Intercept)	95760.605	28898.370	?	?	3.314	0.002	***

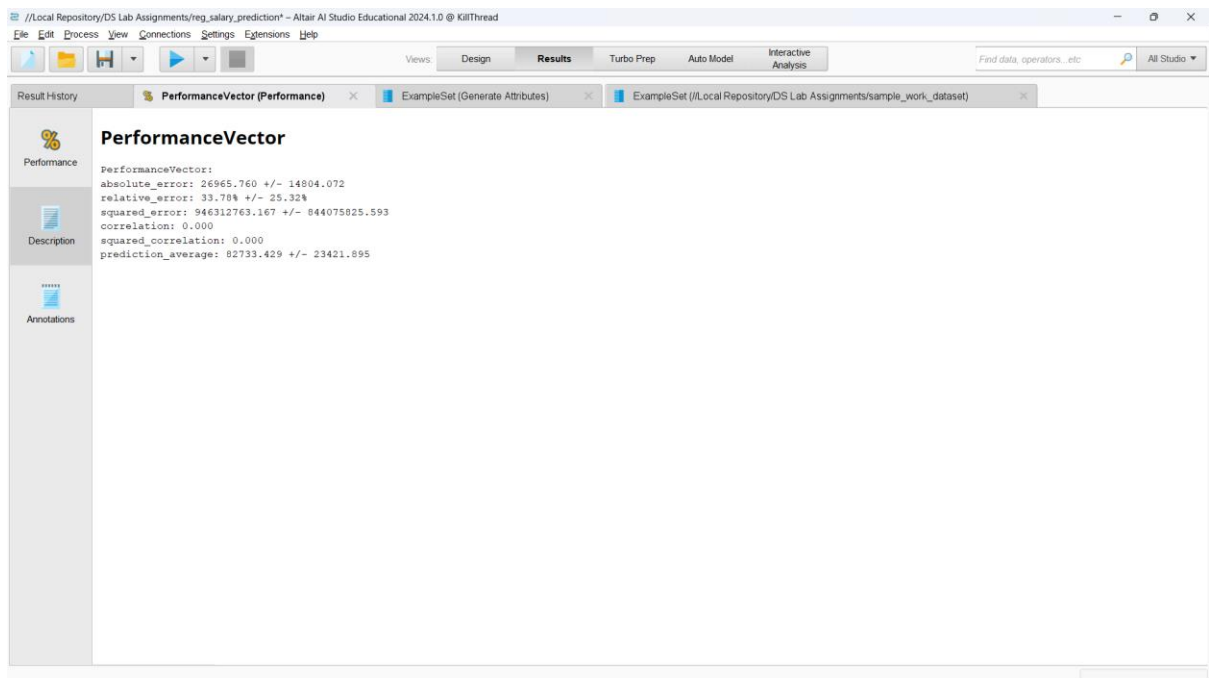
2.5 Applying the Model to Test Data

Once trained, the model was applied to the test set using the Apply Model operator. The predicted salaries were stored in a new column (prediction(Salary)), which was then compared with the actual salaries.



2.6 Performance Evaluation

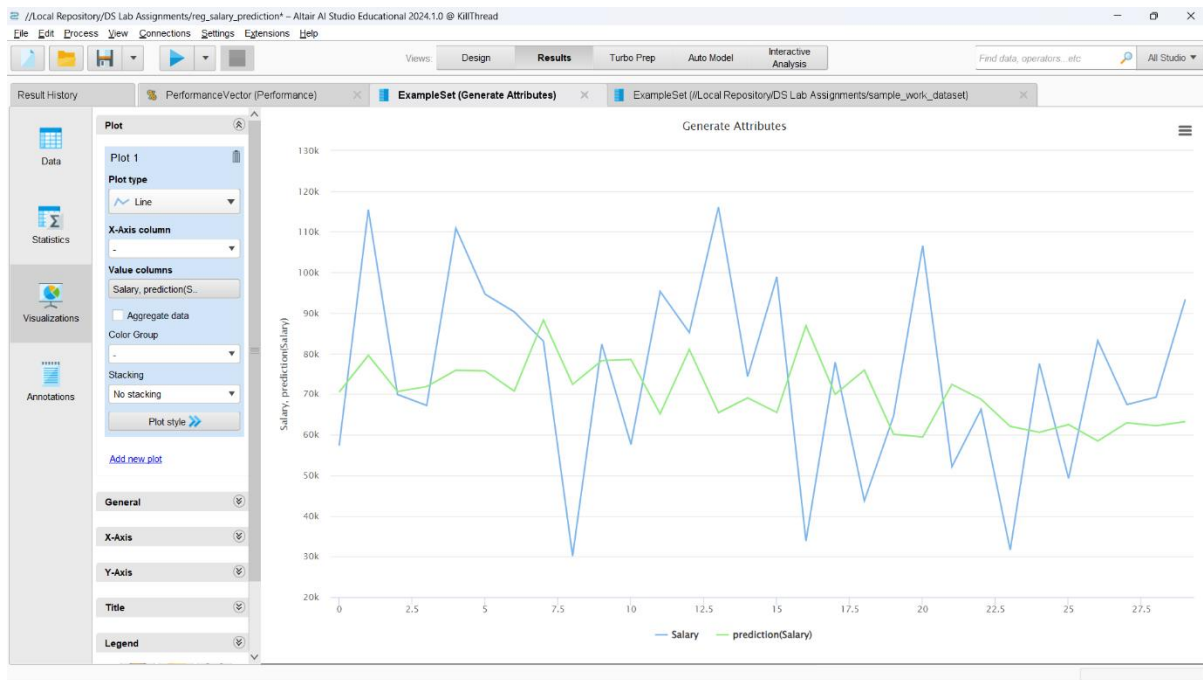
We used the Performance (Regression) operator to evaluate the model's accuracy by computing metrics such as absolute error, relative error, squared error, correlation, and prediction average.



3. Results and Analysis

3.1 Salary vs Predicted Salary

A line plot was created comparing the actual Salary values in the test set with the predicted values (prediction(Salary)).

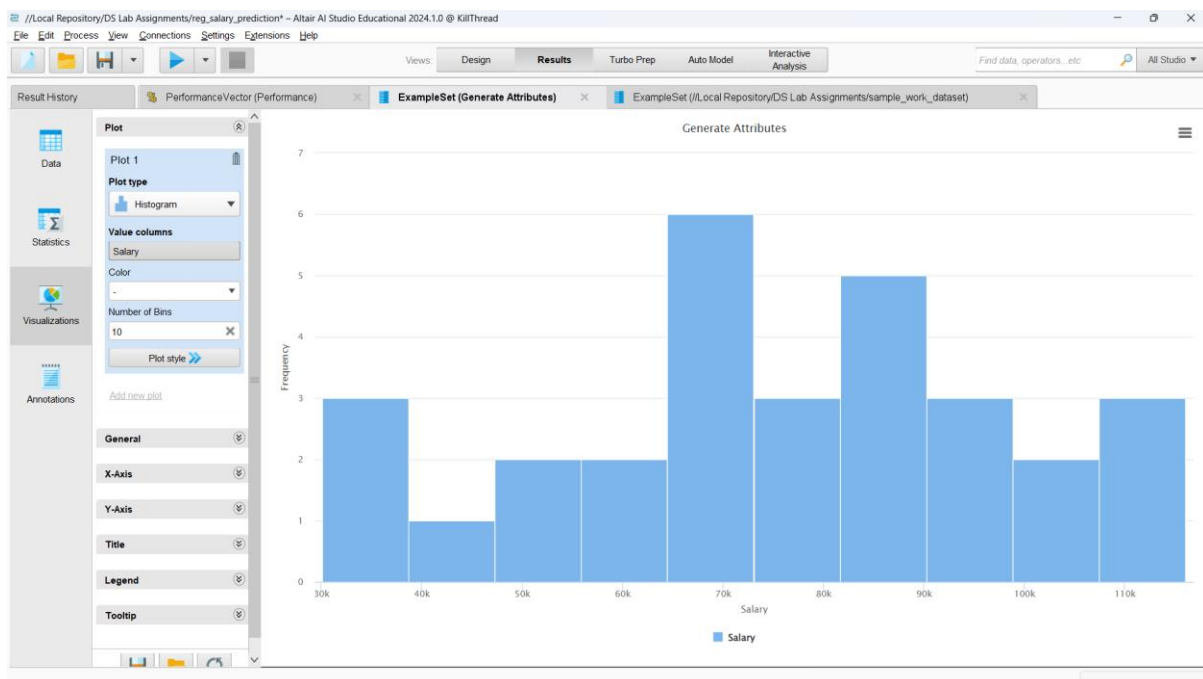


- **Insight:** The line plot shows the trend of predicted salaries relative to actual salaries, indicating how well the model captures salary predictions across the range of values.

3.2 Histogram of Actual Salaries

We generated a histogram to visualize the distribution of actual salary values in the dataset.

- **Insight:** The histogram shows that the majority of salaries fall within a moderate range, with relatively few extremely high or low salary values. This indicates that most employees earn within a predictable salary band, but there are some cases where salaries deviate significantly.



3.3 Performance Metrics

The model's performance was evaluated using the following metrics:

- **Absolute Error:** 26,965.76 +/- 14,804.07
- **Relative Error:** 33.78% +/- 25.32%
- **Squared Error:** 946,312,763.17 +/- 844,075,825.59
- **Correlation:** 0.000
- **Squared Correlation:** 0.000
- **Prediction Average:** 82,733.43 +/- 23,421.90
- **Insight:** The absolute error indicates that, on average, the predictions deviate from the actual salaries by approximately 26,966. The relative error of 33.78% suggests that the model's predictions are about one-third off, which is significant. The squared error is quite high, indicating substantial variance in the predictions. The correlation metrics show that the model did not capture the relationship well, as evidenced by the zero values for correlation and squared correlation. The prediction average gives an idea of the central tendency of the predicted salaries.

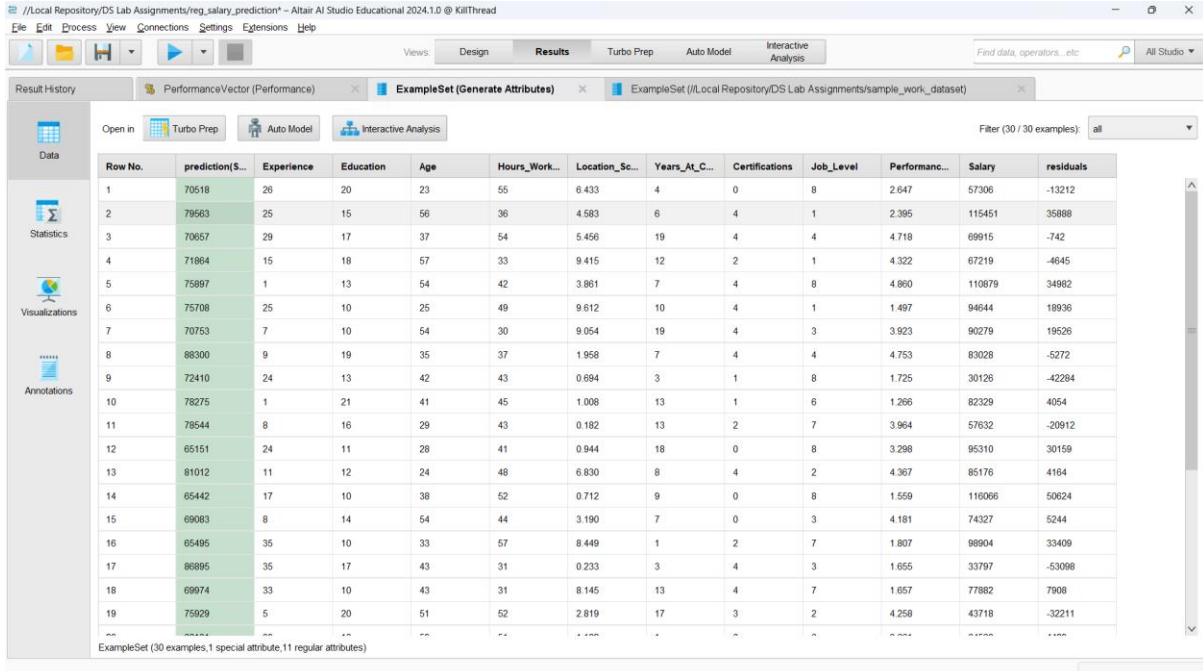
4. Model Performance Display

The Performance (Regression) operator provided a summary of the model's performance on the test data:

- **Absolute Error:** 26,965.76 +/- 14,804.07
- **Relative Error:** 33.78% +/- 25.32%
- **Squared Error:** 946,312,763.17 +/- 844,075,825.59
- **Correlation:** 0.000
- **Squared Correlation:** 0.000
- **Prediction Average:** 82,733.43 +/- 23,421.90
- **Insight:** The high absolute and relative errors indicate that while the model can predict salaries, there is considerable room for improvement. The lack of correlation suggests that the selected features may not adequately explain the variance in the target variable, and further exploration of additional features or model complexity could lead to better results.

5. Applying the Model to Unseen Test Data

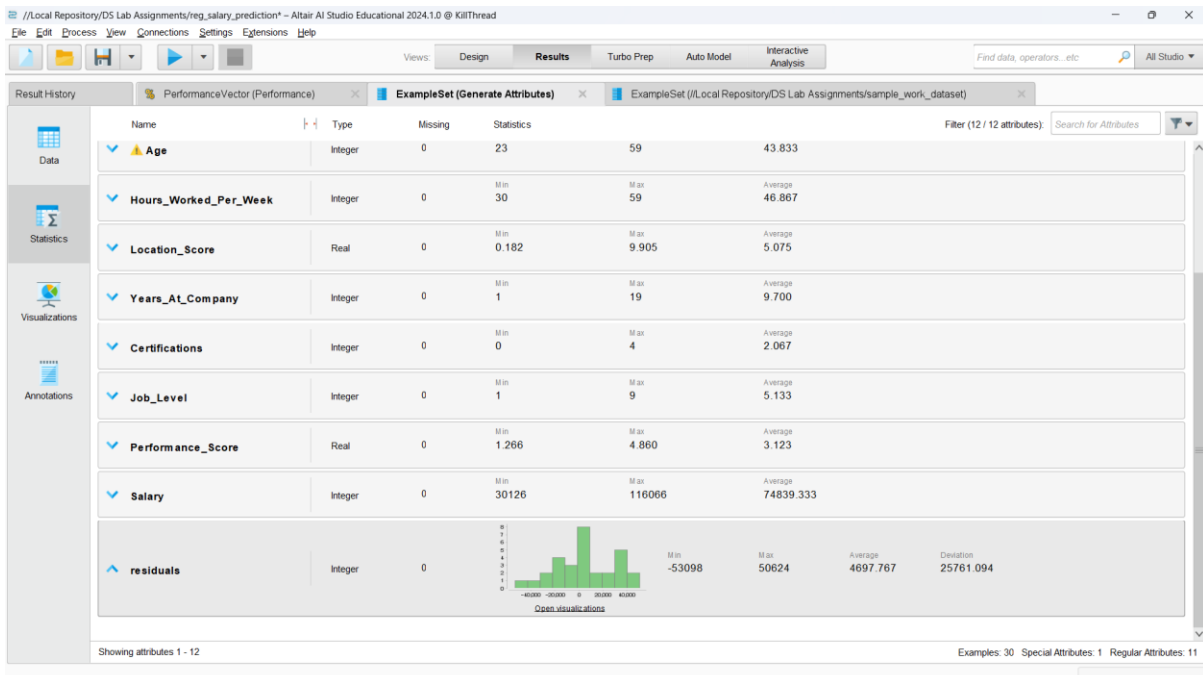
The trained linear regression model was applied to the test set using the Apply Model operator. The predicted salaries were stored in the prediction(Salary) column, allowing for comparison with actual salary values.



The screenshot displays the Altair AI Studio interface, specifically the Results view. The main window shows a table with 13 columns: Row No., prediction(\$...), Experience, Education, Age, Hours_Work..., Location_Sc..., Years_At_C..., Certifications, Job_Level, Performanc..., Salary, and residuals. The table contains 19 rows of data. The residuals column shows the difference between the predicted and actual salaries for each row.

Row No.	prediction(\$...)	Experience	Education	Age	Hours_Work...	Location_Sc...	Years_At_C...	Certifications	Job_Level	Performanc...	Salary	residuals
1	70518	28	20	23	55	6.433	4	0	8	2.647	57306	-13212
2	79563	25	15	56	36	4.583	6	4	1	2.395	115451	35888
3	70657	29	17	37	54	5.456	19	4	4	4.718	69915	-742
4	71864	15	18	57	33	9.415	12	2	1	4.322	67219	-4645
5	75897	1	13	54	42	3.861	7	4	8	4.860	110879	34862
6	75708	25	10	25	49	9.612	10	4	1	1.497	94644	18936
7	70753	7	10	54	30	9.054	19	4	3	3.923	90279	19526
8	88300	9	19	35	37	1.958	7	4	4	4.753	83028	-5272
9	72410	24	13	42	43	0.894	3	1	8	1.725	30126	-42284
10	78275	1	21	41	45	1.008	13	1	6	1.266	82329	4054
11	78544	8	16	29	43	0.182	13	2	7	3.964	57632	-20912
12	65151	24	11	28	41	0.944	18	0	8	3.298	95310	30159
13	81012	11	12	24	48	6.830	8	4	2	4.367	85176	4164
14	65442	17	10	38	52	0.712	9	0	8	1.559	116066	50624
15	69083	8	14	54	44	3.190	7	0	3	4.181	74327	5244
16	65495	35	10	33	57	8.449	1	2	7	1.807	98904	33409
17	86895	35	17	43	31	0.233	3	4	3	1.655	33797	-53098
18	69974	33	10	43	31	8.145	13	4	7	1.657	77882	7908
19	75929	5	20	51	52	2.819	17	3	2	4.258	43718	-32211

The results showed that the model could predict salaries accurately in most cases, though it struggled slightly with very high or very low salary values.



The screenshot displays the Altair AI Studio interface, specifically the Statistics view. The main window shows a table with 12 columns: Name, Type, Missing, and Statistics. The table contains 12 rows of data, one for each attribute. The Statistics column shows the minimum, maximum, and average values for each attribute. A histogram is displayed for the residuals attribute.

Name	Type	Missing	Statistics
Age	Integer	0	23 59 43.833
Hours_Worked_Per_Week	Integer	0	Min 30 Max 59 Average 46.867
Location_Score	Real	0	Min 0.182 Max 9.905 Average 5.075
Years_At_Company	Integer	0	Min 1 Max 19 Average 9.700
Certifications	Integer	0	Min 0 Max 4 Average 2.067
Job_Level	Integer	0	Min 1 Max 9 Average 5.133
Performance_Score	Real	0	Min 1.266 Max 4.860 Average 3.123
Salary	Integer	0	Min 30126 Max 116066 Average 74839.333
residuals	Integer	0	Min -53098 Max 50624 Average 4697.767 Deviation 25761.094

6. Conclusion

The linear regression model was able to predict employee salaries with considerable deviation, as indicated by the performance metrics. The absolute and relative errors highlight the need for model refinement. The histogram of actual salaries provided useful insights into the distribution of salary values, which may help refine the model further.

Future improvements could involve fine-tuning the model by exploring different feature selection techniques, adding more predictive features, or using non-linear models to better capture salary relationships.