Babu Banarasi Das University

Case Study on

Predictive Analysis of Car Price Segmentation Using SPSS

(BCADSN15301)

Project Overview

Submitted By-

MD AYAN KHAN

MD HASHIR KHAN

PRIYANSHU GUPTA

(BCA DS&AI 33)

Submitted To-

MR. ROBIN TYAGI SIR

Project Definition and Goals

Definition: This is a Classification Modeling project using the CHAID Decision Tree algorithm to predict a car's price category (High vs. Standard) based on its features.

Purpose: To identify which car attributes (like Mileage, Engine Size, and Year) are the most significant drivers of high market value, achieving a reliable price prediction model.

Outcomes/Learning:

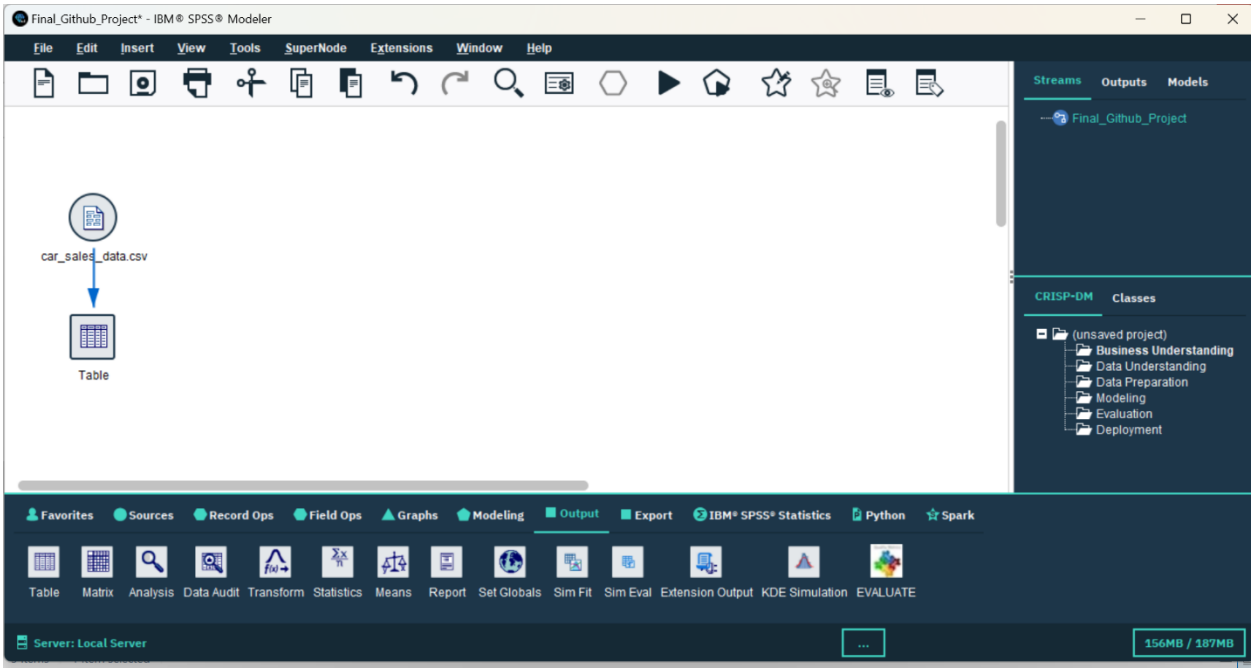
Data Transformation: Learned to use the Derive node to convert a continuous variable (Price) into a categorical Flag (Price_Category) suitable for classification models.

Model Building: Applied the CHAID node to automatically identify and build decision rules based on the strongest predictor variables in the dataset.

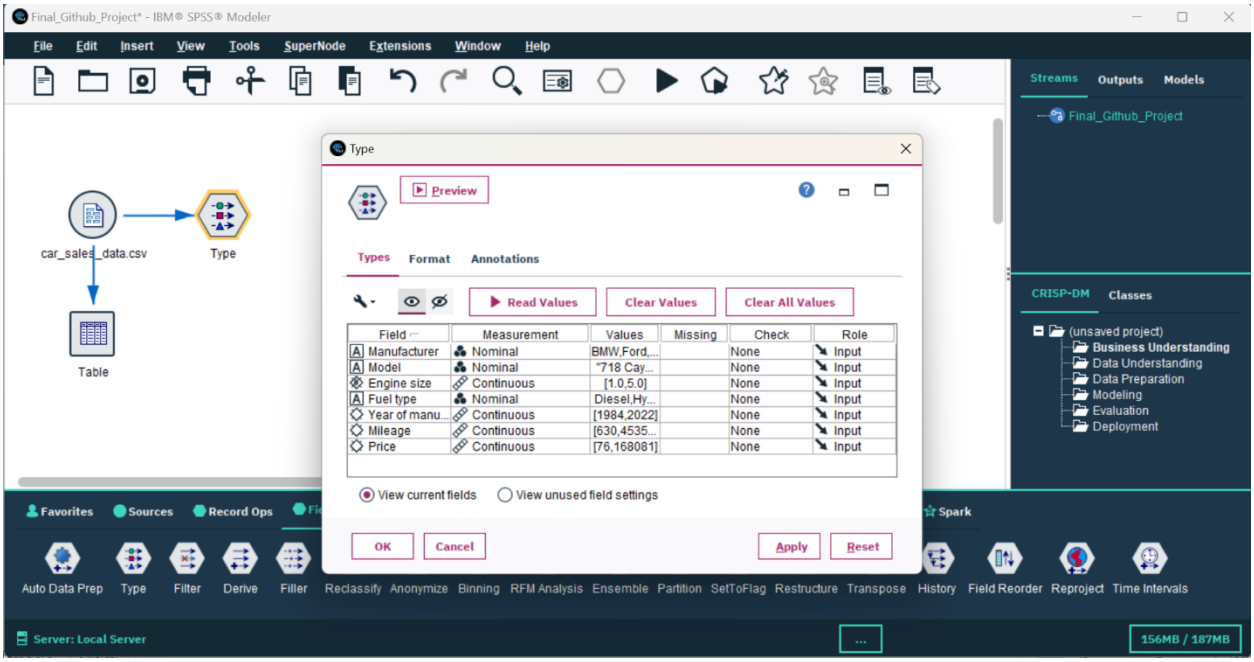
Required Tool: IBM SPSS MODELER TOOL

Working: - In this Project we will use 'car_sales_data.csv' dataset and perfom Car Price Classification and Segmentation Analysis. The following steps are going to be executed in IBM SPSS MODELER:

Step 1: From Sources we will select Var. File node and import our dataset (car_sales_data.csv). After that select Table node from output and then connect the dataset with the table to preview the output.



Step 2: Now we will use Type Node from Field Ops and connect it to car_sales_data.csv node to Read Values meaning it reads the data and automatically determines the measurement level and values for each field.

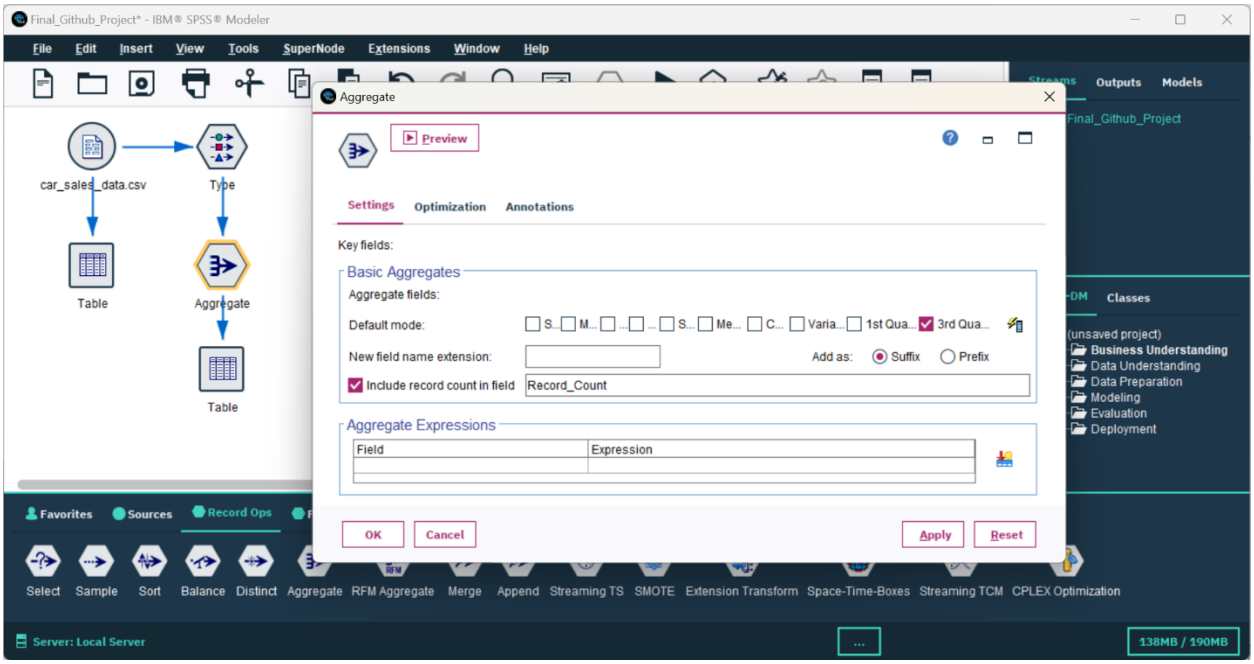


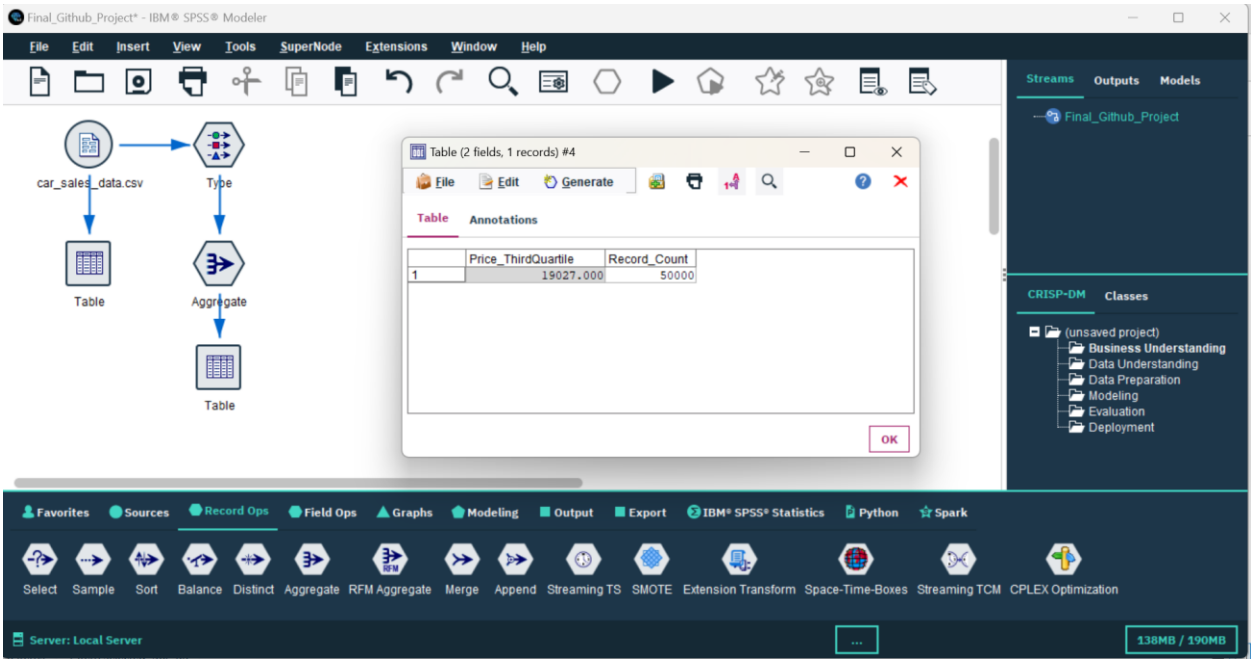
Step 3: We will Add the Aggregate Node (from Record Ops) and calculate the 3rd Quartile (75th Percentile) of the car price.

• The output shows a 3rd Quartile value of $19,027.

• This means 75% of all cars in the dataset sold for this price or less.

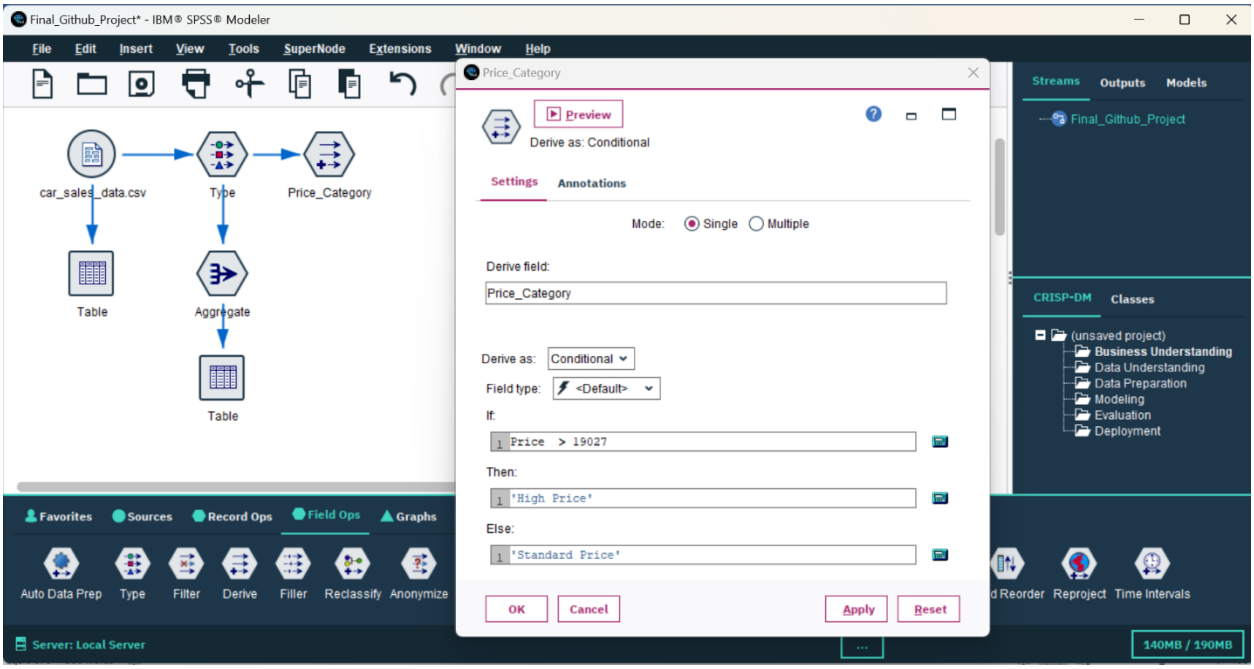
• We use this value to set the cutoff, defining the ‘High Price’ category as the top 25% of the market value.

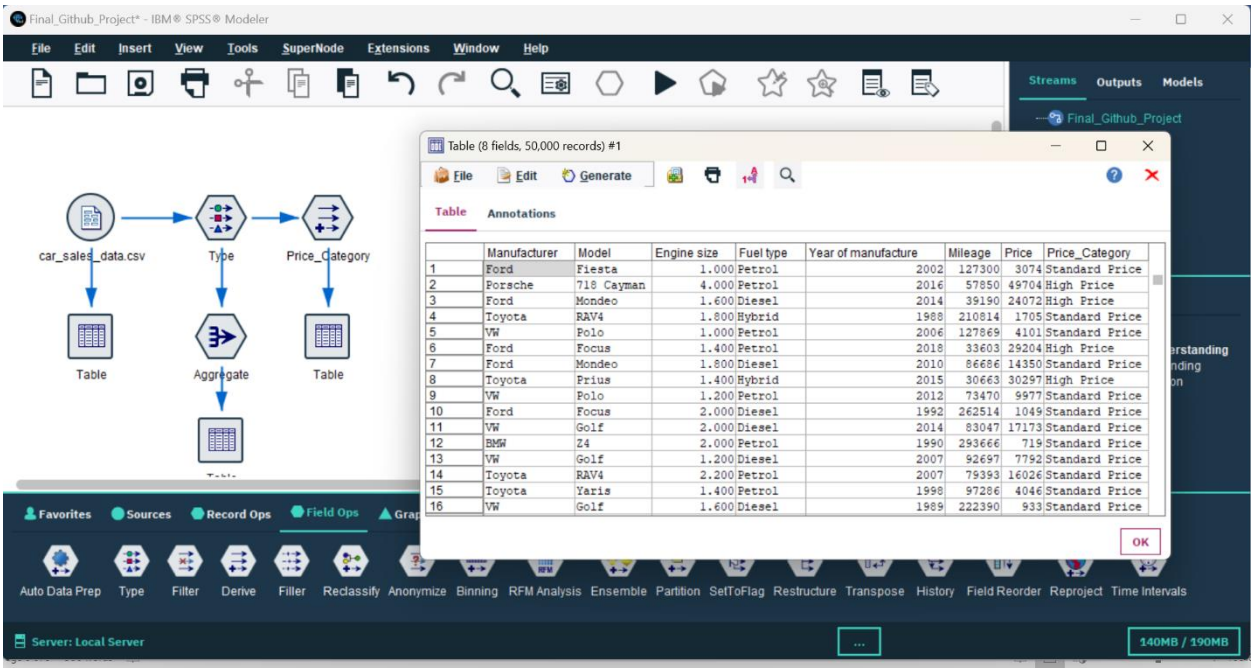

Final_Github_Project* - IBM® SPSS® Modeler
File Edit Insert View Tools SuperNode Extensions Window Help
car_sales_data.csv
Type
Table
Aggregate
Table
Aggregate
Preview
Settings Optimization Annotations
Key fields:
Basic Aggregates
Aggregate fields:
Default mode:
New field name extension:
Add as: Suffix Prefix
Include record count in field
Record_Count
Aggregate Expressions
Field
Expression
OK Cancel Apply Reset
Streams Outputs Models
Final_Github_Project
Classes
(unsaved project)
Business Understanding
Data Understanding
Data Preparation
Modeling
Evaluation
Deployment
Favorites Sources Record Ops
Select Sample Sort Balance Distinct Aggregate RFM Aggregate Merge Append Streaming TS SMOTE Extension Transform Space-Time-Boxes Streaming TCM CPLEX Optimization
Server: Local Server
138MB / 190MB


Final_Github_Project* - IBM® SPSS® Modeler
File Edit Insert View Tools SuperNode Extensions Window Help
car_sales_data.csv
Type
Table
Aggregate
Table
Table (2 fields, 1 records) #4
File Edit Generate
Table Annotations
Price_ThirdQuartile
Record_Count
1
19027.000
50000
OK
Streams Outputs Models
Final_Github_Project
CRISP-DM Classes
(unsaved project)
Business Understanding
Data Understanding
Data Preparation
Modeling
Evaluation
Deployment
Favorites Sources Record Ops Field Ops Graphs Modeling Output Export IBM® SPSS® Statistics Python Spark
Select Sample Sort Balance Distinct Aggregate RFM Aggregate Merge Append Streaming TS SMOTE Extension Transform Space-Time-Boxes Streaming TCM CPLEX Optimization
Server: Local Server
138MB / 190MB

Step 4: Now we will take the Derive Node from Field Ops and connect it to type node and in the Derive Node we will do “Derive as: Conditional and Field Type: Default” ***.If*** Price > 19027 ***Then*** 'High Price' ***Else*** 'Standard Price'

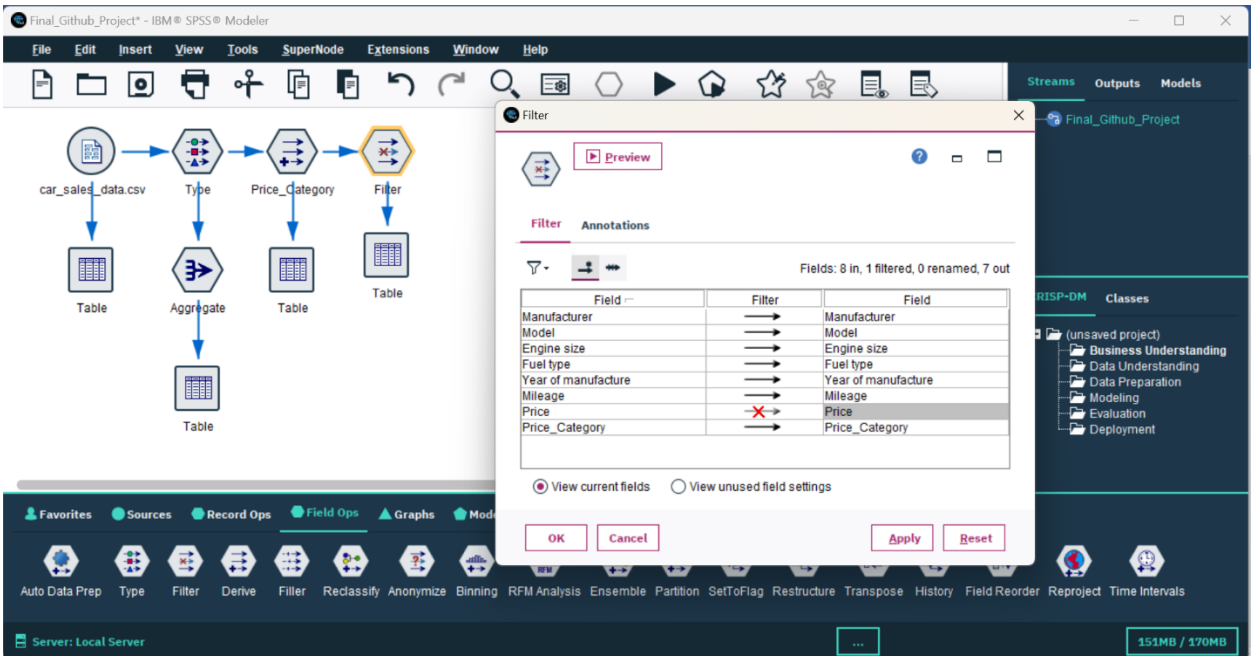
Apply and then connect a table and run it to see the output

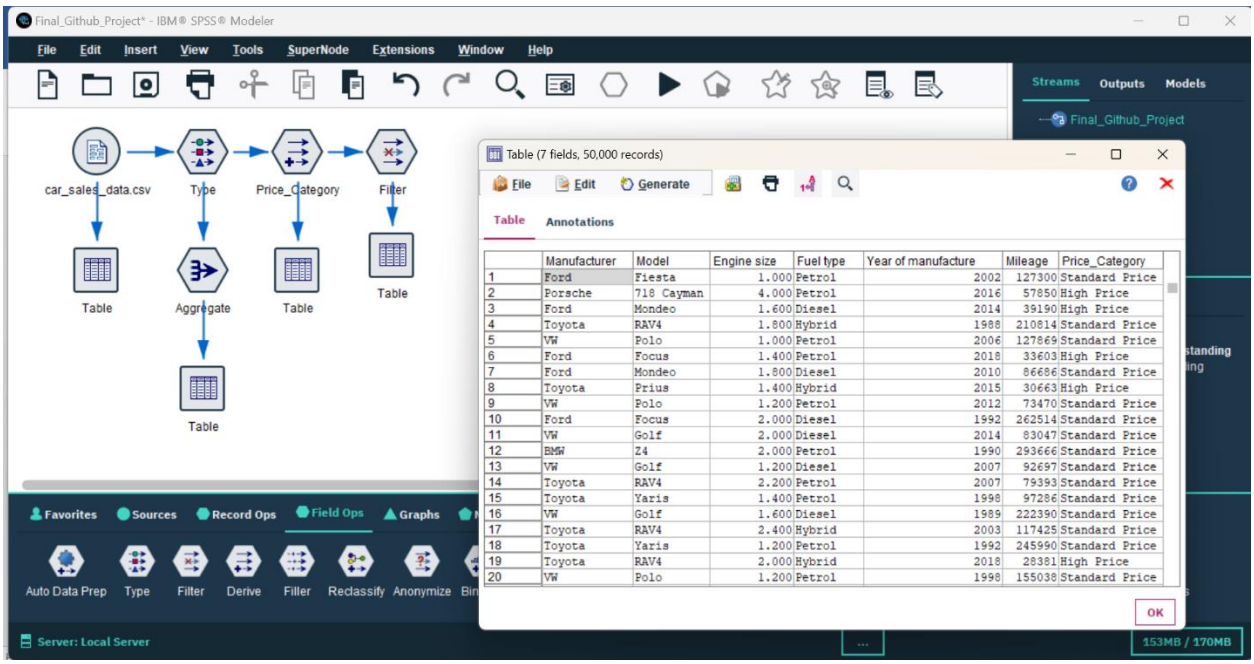




Step 5: Now we will use Filter Node from Field Ops and connect it to the Derive Node (Price_Category). In the Filter Node we will Cross (or deselect) the Price field.

And then connect a table and run.



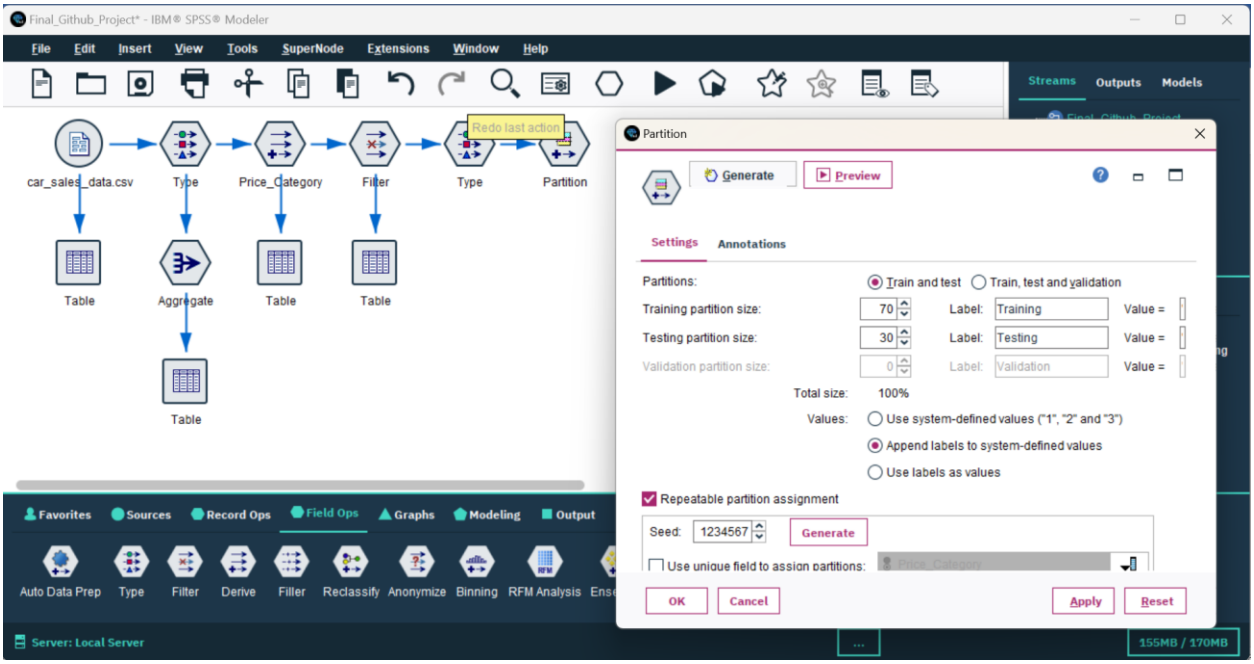


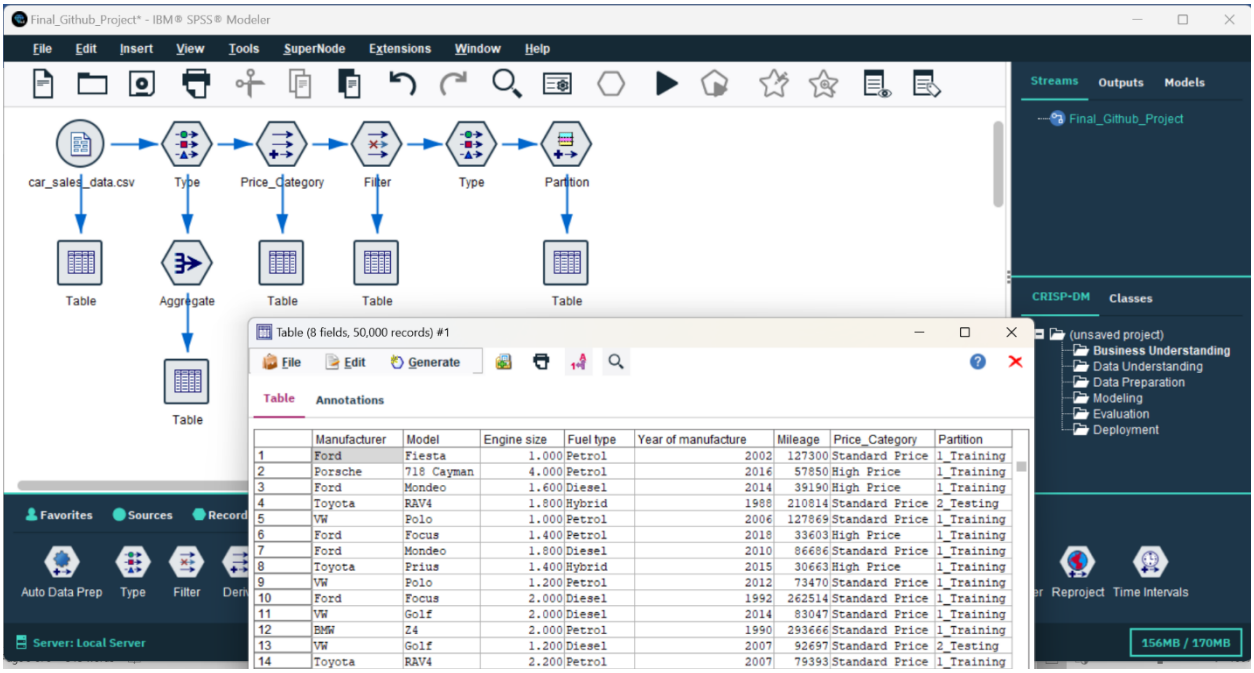
Step 6: We will now add 2nd Type Node from Field Ops and connect it to our previous filter node .Now in the Type Node we will change the role of 'Price_Category' as Target and every other as Input.

Step 7: Now we will add Partition node from Field Ops and connect it to the previous or 2nd Type Node. In the Partition Node we will split the data into a 70% Training Set and a 30% Testing Set.

(This Division is basic for correctly validating the model's accuracy on data it has not processed yet.)

And then Connect a Table to see the output.



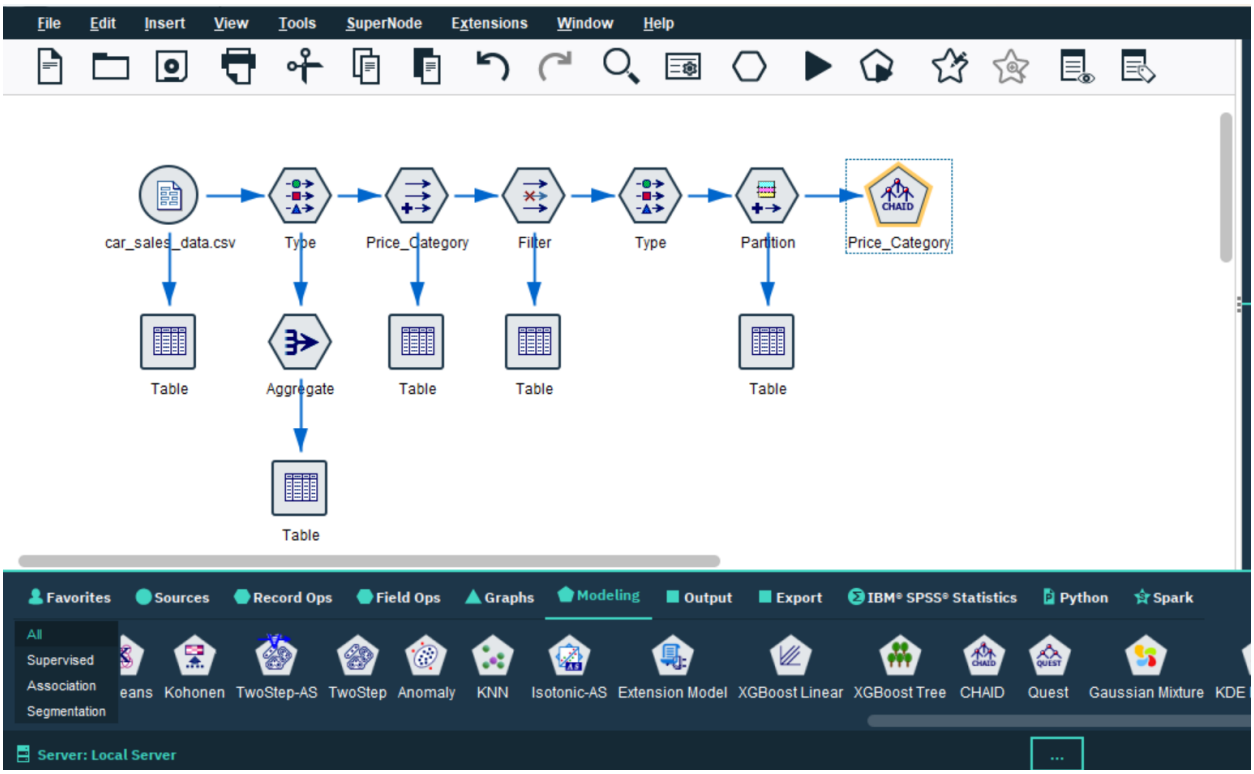


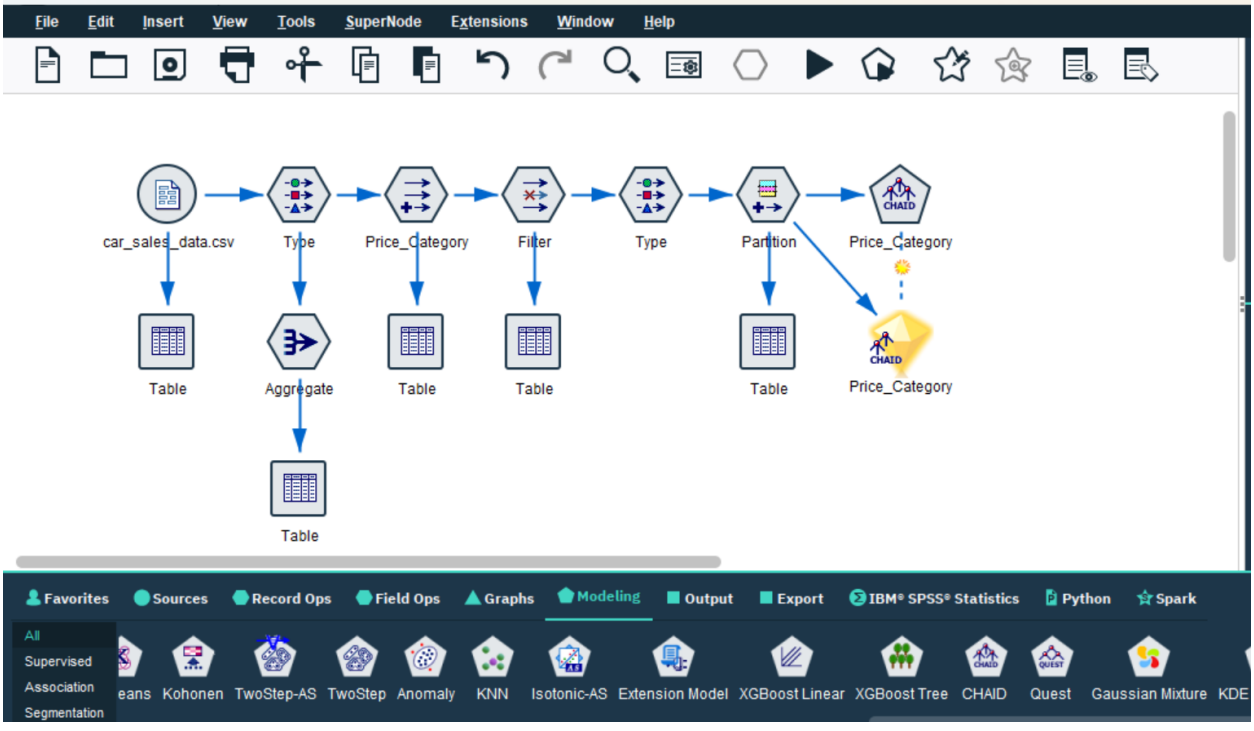
	Manufacturer	Model	Engine size	Fuel type	Year of manufacture	Mileage	Price_Category	Partition
1	Ford	Fiesta	1.000	Petrol	2002	127300	Standard Price	1_Training
2	Porsche	718 Cayman	4.000	Petrol	2016	57850	High Price	1_Training
3	Ford	Mondeo	1.600	Diesel	2014	39190	High Price	1_Training
4	Toyota	RAV4	1.800	Hybrid	1988	210814	Standard Price	2_Testing
5	VW	Polo	1.000	Petrol	2006	127869	Standard Price	1_Training
6	Ford	Focus	1.400	Petrol	2018	33603	High Price	1_Training
7	Ford	Mondeo	1.800	Diesel	2010	86686	Standard Price	1_Training
8	Toyota	Prius	1.400	Hybrid	2015	30663	High Price	1_Training
9	VW	Polo	1.200	Petrol	2012	73470	Standard Price	1_Training
10	Ford	Focus	2.000	Diesel	1992	262514	Standard Price	1_Training
11	VW	Golf	2.000	Diesel	2014	83047	Standard Price	1_Training
12	BMW	Z4	2.000	Petrol	1990	293666	Standard Price	1_Training
13	VW	Golf	1.200	Diesel	2007	92697	Standard Price	2_Testing
14	Toyota	RAV4	2.200	Petrol	2007	79393	Standard Price	1_Training

Step 8: Now from Modelling , we will add a CHAID Node and connect it to the Partion Node. In the CHAID Node we will make Price_Category as Target and else as Inputs.

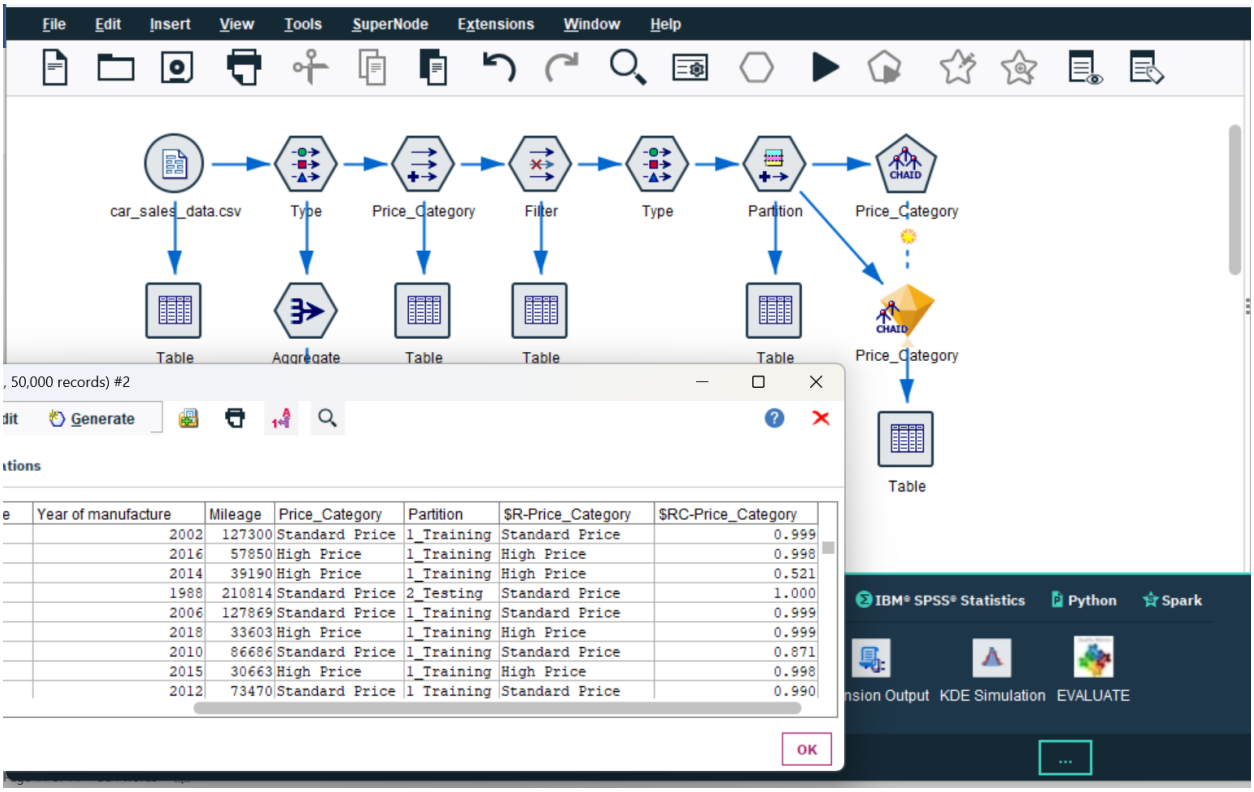
After Running it .It will automatically generate a CHAID NUGGET

(The Resulting Decision Tree Identified Mileage as the single most important factor for Predicting the price Category)



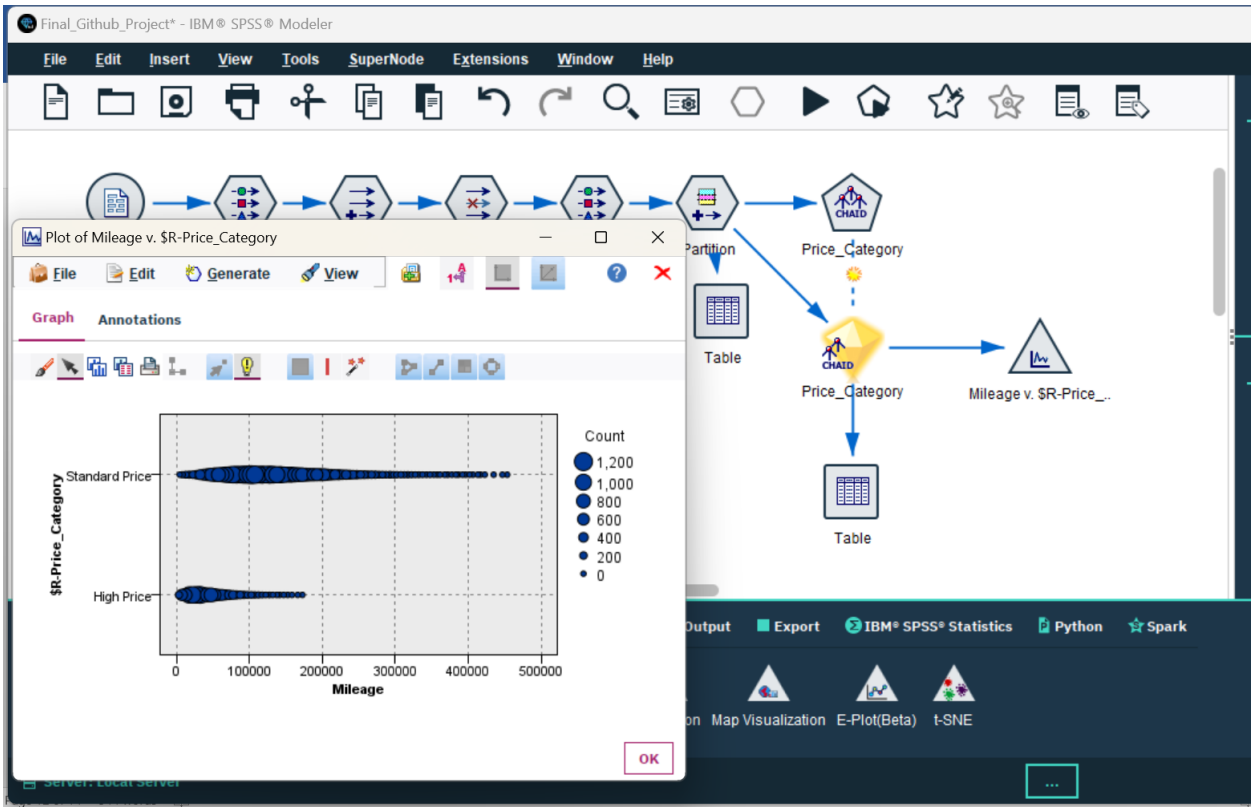


After connecting a Table to Output Node, We can see two new columns $R-Price_Category(Prediction) and $RC-Price_Category(Confidence)



STEP 9: We will connect a Plot Node from Graphs to the CHAID Nugget and in the plot node we will put Mileage in the X-Field and $R-Price_Category in Y-Field to visualize Predicted Category vs. Mileage.

The graph clearly showed that High Price predictions are concentrated in the low mileage range, validating the model's internal logic.



STEP 10: And at Last, We will add Analysis Node from Output and connect it to the CHAID Nugget and run it.

[Analysis Node (Confusion matrix) is to see the Classification Accuracy (the percentage of records the model correctly predicted in your testing partition)].

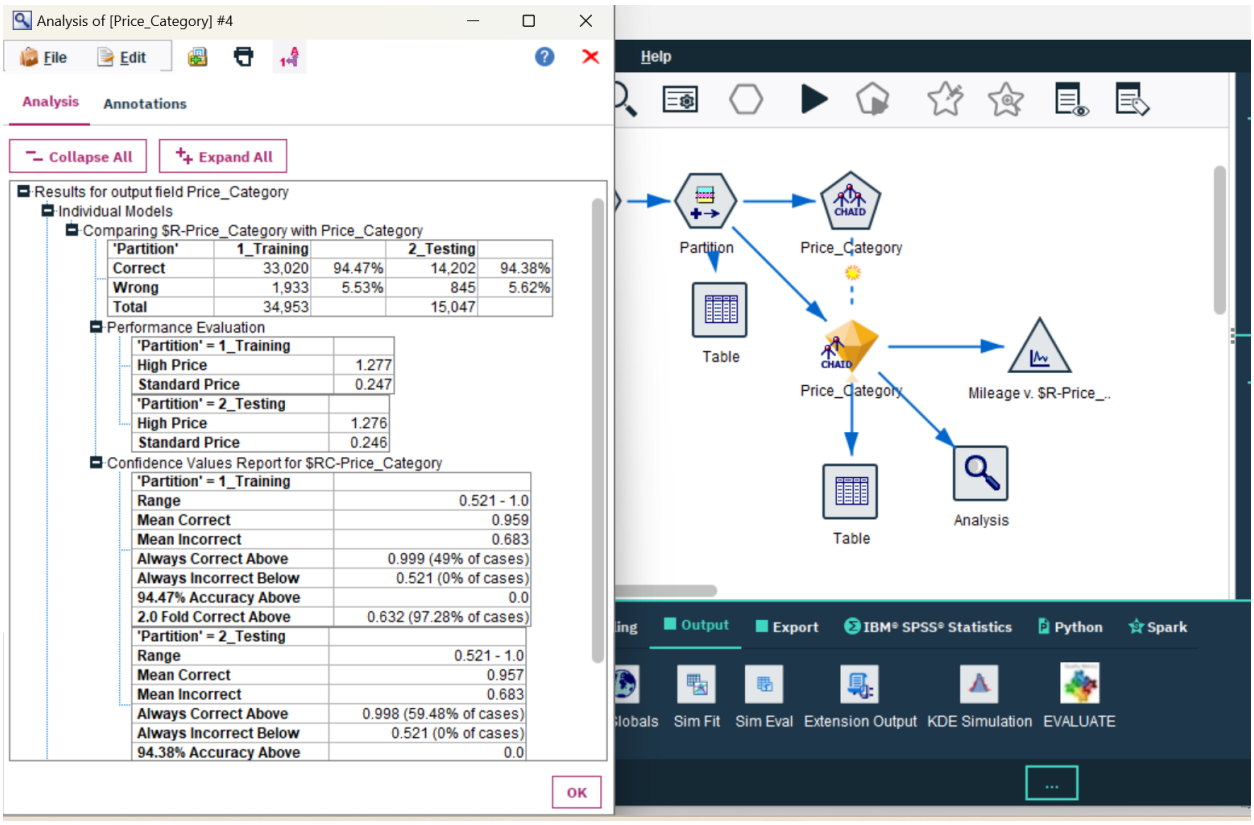
After running the Analysis Node we can see the result.

Overall Accuracy – 94.27% **-**The model correctly classified the Price_Category for over 94% of all test records, indicating excellent predictive performance.

High Price Accuracy – 86.30% - The model correctly identified 86.30% of the actual 'High Price' cars. (This is the Recall for the 'High Price' class).

Standard Price Accuracy – 96.80% - The model correctly identified 96.80% of the actual 'Standard Price' cars.

Total Errors – 877 records - Out of 15,300 test records, the model made 877 errors (506 False Negatives + 371 False Positives).



Conclusion

The CHAID model achieved high overall accuracy (94.27%). While it is slightly better at identifying 'Standard Price' cars (96.80\%), it still has a very strong 86.30% success rate in correctly identifying the target 'High Price' segment. This demonstrates that the features selected by the model are highly effective at differentiating high-value vehicles.