Project Report

Full Name: Komnas Kafasis

Introduction

This is a project on the university course Business Intelligence and Big Data Analysis. For this project I had to find a dataset, which had to be cleansed and be inserted in a data warehouse. I also had to create a cube and metrics. Moreover, I also had to use Power BI to create various data visualizations. Lastly, the data from the data warehouse were used to create 2 data mining models.

Tools Used

The tools used in this project are the following:

- Python
- SQL Server Management Studio 19
- SQL Server
- Visual Studio 2022
- Microsoft Analysis Services Projects extension (Visual Studio)
- SQL Integration Services Projects extension (Visual Studio)
- Power BI
- RapidMiner Studio

Dataset Presentation

The dataset was found on kaggle.com here and its name is Car Sales Report. It's a dataset about car sales.

In short, bellow is the description of each column of the dataset.

Car_id: Unique identifier for each car in the dataset.

Date: Date of the car sale transaction.

Customer Name: Name of the customer purchasing the car.

Gender: Gender of the customer (e.g., Male, Female).

Annual Income: Annual income of the customer.

Dealer_Name: Name of the car dealer associated with the sale.

Company: Company or brand of the car.

Model: Model name of the car.

Engine: Specifications of the car's engine.

Transmission: Type of transmission in the car (e.g., Automatic, Manual).

Color: Color of the car's exterior.

Price: Listed price of the car for sale.

Dealer_No: Dealer identification number associated with the sale.

Body Style: Style or design of the car's body (e.g., Sedan, SUV).

Phone: Contact phone number associated with the car sale.

Dealer_Region: Geographic region or location of the car dealer.

Below is an image representation of the dataset.

Car_id	Date	Customer ...	Gender	Annual Inc...	Dealer_Na...	Company	Model
C_CND_000001	1/2/2022	Geraldine	Male	13500	Buddy Storbeck's Diesel Service Inc	Ford	Expedition
C_CND_000002	1/2/2022	Gia	Male	1480000	C & M Motors Inc	Dodge	Durango
C_CND_000003	1/2/2022	Gianna	Male	1035000	Capitol KIA	Cadillac	Eldorado
C_CND_000004	1/2/2022	Giselle	Male	13500	Chrysler of Tri-Cities	Toyota	Celica
C_CND_000005	1/2/2022	Grace	Male	1465000	Chrysler Plymouth	Acura	TL
C_CND_000006	1/2/2022	Guadalupe	Male	850000	Classic Chevy	Mitsubishi	Diamante
C_CND_000007	1/2/2022	Hailey	Male	1600000	Clay Johnson Auto Sales	Toyota	Corolla
C_CND_000008	1/2/2022	Graham	Male	13500	U-Haul CO	Mitsubishi	Galant
C_CND_000009	1/2/2022	Naomi	Male	815000	Rabun Used Car Sales	Chevrolet	Malibu
C_CND_000010	1/2/2022	Grayson	Female	13500	Rabun Used Car Sales	Ford	Escort
C_CND_000011	1/2/2022	Gregory	Male	13500	Race Car Help	Acura	RL
C_CND_000012	1/2/2022	Amar'E	Male	13500	Race Car Help	Nissan	Pathfinder
C_CND_000013	1/2/2022	Griffin	Male	885000	Saab-Belle Dodge	Mercury	Grand Marquis
C_CND_000014	1/2/2022	Harrison	Male	13500	Scrivener Performance Engineering	BMW	323i
C_CND_000015	1/2/2022	Zainab	Male	722000	Buddy Storbeck's Diesel Service Inc	Chrysler	Sebring Coupe
C_CND_000016	1/2/2022	Zara	Male	746000	C & M Motors Inc	Subaru	Forester
C_CND_000017	1/2/2022	Zoe	Female	535000	Capitol KIA	Hyundai	Accent
C_CND_000018	1/2/2022	Zoey	Female	570000	Chrysler of Tri-Cities	Cadillac	Eldorado
C_CND_000019	1/2/2022	Aaliyah	Male	685000	Chrysler Plymouth	Toyota	Land Cruiser
C_CND_000020	1/2/2022	Abigail	Male	455000	Classic Chevy	Honda	Accord
C_CND_000021	1/2/2022	Adrianna	Male	13500	Clay Johnson Auto Sales	Toyota	4Runner
C_CND_000022	1/2/2022	Joshua	Male	2500000	Classic Chevy	Infiniti	I30

Car_id	Engine	Transmissi...	Color	Price ($)	Dealer_No	Body Style	Phone	Dealer_Re...
C_CND_000001	DoubleÂ Overhead Camshaft	Auto	Black	26000	06457-3834	SUV	8264678	Middletown
C_CND_000002	DoubleÂ Overhead Camshaft	Auto	Black	19000	60504-7114	SUV	6848189	Aurora
C_CND_000003	Overhead Camshaft	Manual	Red	31500	38701-8047	Passenger	7298798	Greenville
C_CND_000004	Overhead Camshaft	Manual	Pale White	14000	99301-3882	SUV	6257557	Pasco
C_CND_000005	DoubleÂ Overhead Camshaft	Auto	Red	24500	53546-9427	Hatchback	7081483	Janesville
C_CND_000006	Overhead Camshaft	Manual	Pale White	12000	85257-3102	Hatchback	7315216	Scottsdale
C_CND_000007	Overhead Camshaft	Manual	Pale White	14000	78758-7841	Passenger	7727879	Austin
C_CND_000008	DoubleÂ Overhead Camshaft	Auto	Pale White	42000	78758-7841	Passenger	6206512	Austin
C_CND_000009	Overhead Camshaft	Manual	Pale White	82000	85257-3102	Hardtop	7194857	Pasco
C_CND_000010	DoubleÂ Overhead Camshaft	Auto	Pale White	15000	85257-3102	Passenger	7836892	Scottsdale
C_CND_000011	Overhead Camshaft	Manual	Pale White	31000	78758-7841	SUV	7995489	Austin
C_CND_000012	DoubleÂ Overhead Camshaft	Auto	Pale White	46000	78758-7841	Hardtop	7288103	Pasco
C_CND_000013	DoubleÂ Overhead Camshaft	Auto	Black	9000	60504-7114	SUV	6842408	Aurora
C_CND_000014	DoubleÂ Overhead Camshaft	Auto	Pale White	15000	38701-8047	Hatchback	7558767	Greenville
C_CND_000015	Overhead Camshaft	Manual	Pale White	26000	06457-3834	Sedan	7677191	Middletown
C_CND_000016	Overhead Camshaft	Manual	Pale White	17000	60504-7114	Hatchback	8431908	Aurora
C_CND_000017	Overhead Camshaft	Manual	Black	18000	38701-8047	Hatchback	7814646	Greenville
C_CND_000018	DoubleÂ Overhead Camshaft	Auto	Pale White	31000	99301-3882	Passenger	7456650	Pasco
C_CND_000019	DoubleÂ Overhead Camshaft	Auto	Pale White	33000	53546-9427	SUV	7627010	Janesville
C_CND_000020	DoubleÂ Overhead Camshaft	Auto	Pale White	21000	85257-3102	Sedan	6736704	Scottsdale
C_CND_000021	Overhead Camshaft	Manual	Black	25000	78758-7841	Sedan	7889827	Austin
C_CND_000022	DoubleÂ Overhea	Auto	Black	21000	85257-3102	Hardtop	6183219	Austin

Data Cleansing

There are four problems that arise when trying to import the data. First the Price column has a ($) sign which signifies that the price is in dollars and isn't useful, but its solvable by a simple deletion. Secondly, when we import the dataset we want the Date column to be of datatype Date. The Date column isn't in the correct format of sql Date type in which there's '-' instead of '/'. Lastly there's an encoding problem in the Engine column where there's an Â character instead of a space character.

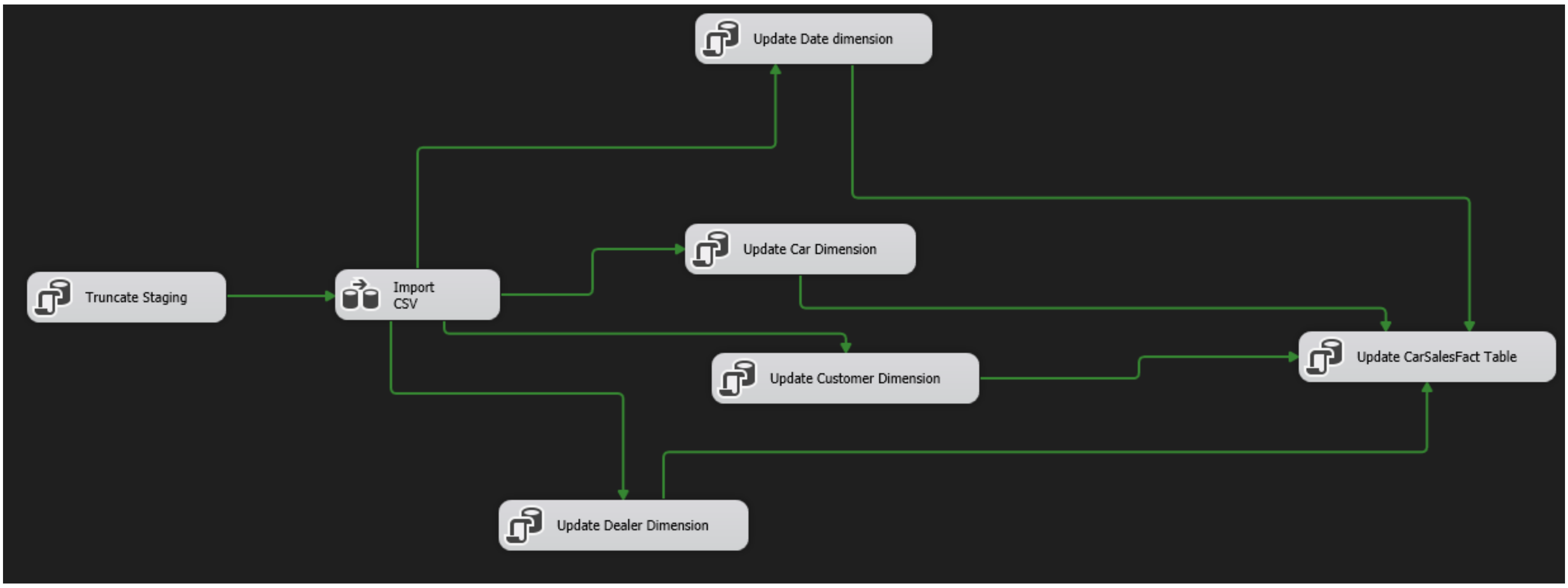
To solve these problems the python script bellow was created.

```python
import pandas as pd

# Read the CSV file
df = pd.read_csv('Car Sales.csv', encoding='utf-8')

# Specify the column you want to correct
column_to_correct = 'Engine'

# Replace "Double Overhead" with "DoubleÂ Overhead"
df[column_to_correct] = df[column_to_correct].str.replace('Double Overhead', 'DoubleÂ Overhead')

date_column = 'Date'

# Convert dates from / to -
df[date_column] = pd.to_datetime(df[date_column].str.replace('/', '-'))


# Save the corrected data back to the CSV file
df.to_csv('Car Sales.csv', index=False, encoding='utf-8')
```

The script uses the pandas library to read the csv file using a dataframe. First it tries to replace the Â character with space, which it succeeds in but in the end when the dataset is inserted into the database it is recognized as a B. This is the best I could do. After that it replaces the '/' character with the '-' character for proper date formatting. In the end it saves the updated dataframe to the csv file.

ETL PROCESS

As for the ETL process, we have to set it up so that we can schedule it and make it an automated process. For this reason, I set up an Integration Services project on Visual Studio 2022 using the SQL Integration Services Projects extension. I set up a control flow as seen on the image below.

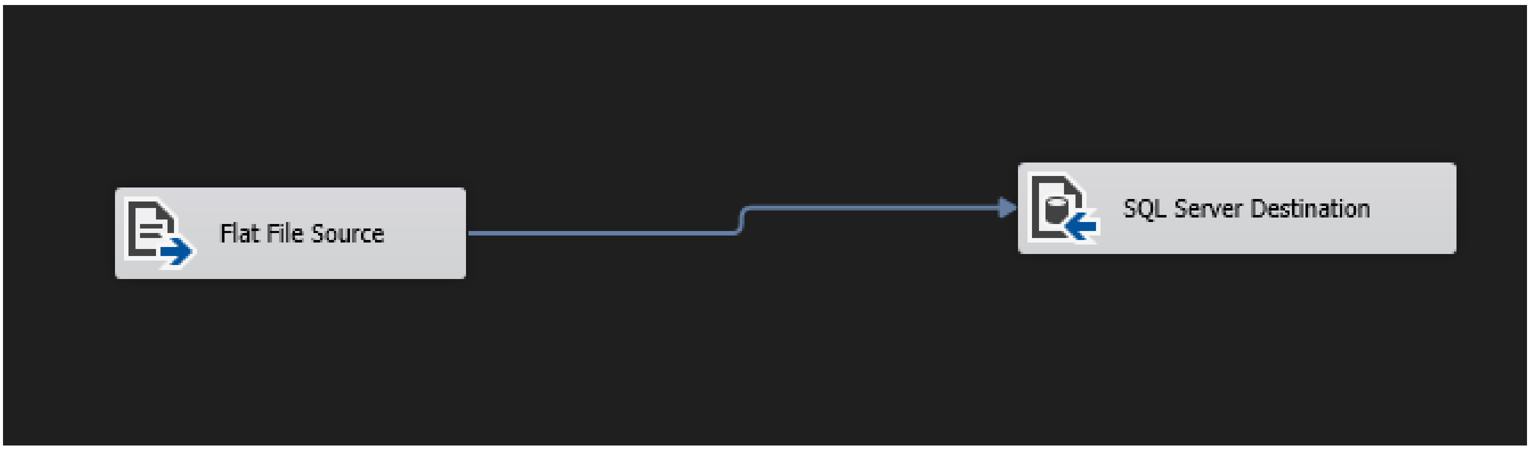


First the staging table, which holds from the csv, gets truncated, so all data are deleted. This ensures that we’re working with the latest version of the data. If this step didn’t exist, then the staging table would store the same data multiple times. Then the CSV data is imported from the csv file to the staging table in the server. After that the dimension tables are updated with the new data from the staging table. In the end the Fact table is also updated with the new data.

First, I set up the “Import CSV” task on the control flow and set up its data flow as seen below.

So, as for the Import CSV task, first a flat file (in this case a csv file) is extracted and moved to the SQL destination Server. For this I also set up a connection manager and declared the column data types.

As for the SQL Server Destination task I connected to the SQL Server and created a table called staging to write the data from the dataset.



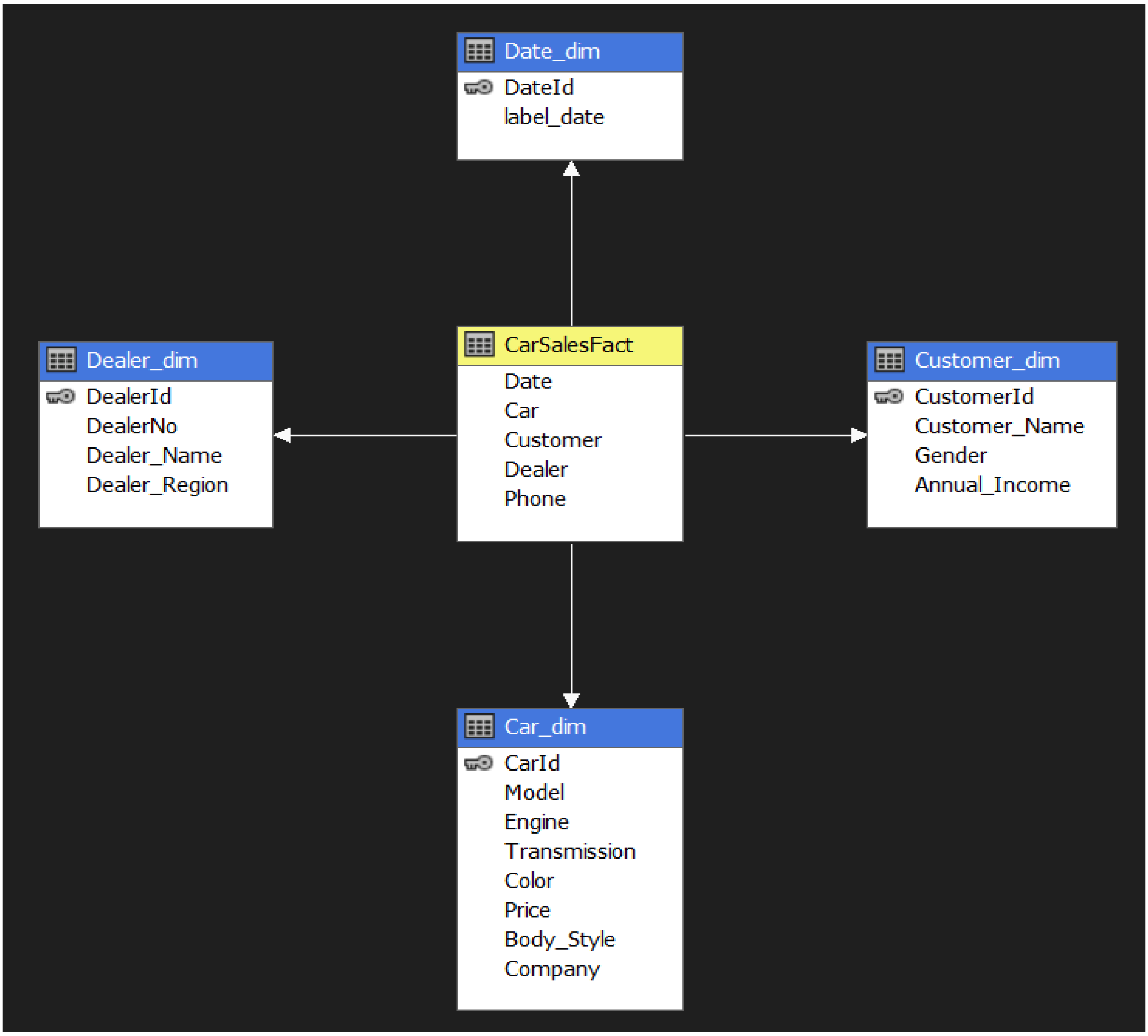
After that I created a Truncate Staging task to truncate the stating table before importing the csv file. For this task I inserted the SQL Server connection and the following SQL statement to be executed:

```
TRUNCATE TABLE [CarSales].[dbo].[staging]
```

Star Schema

For the schema of the data warehouse i used a star schema that will look like on the image below.

The dimension tables have an id column and one or more attributes. Each dimension table holds unique values for each record with unique ids on the id column. For example the date dimension holds the unique dates of sales in the label column and unique ids on the id column. The id columns are primary keys of the dimension tables.



The CarSalesFact is the fact table and contains the ids of the dimensions. So I created the dimensions and the fact table and declared the primary and foreign keys.

For the update of the dimension tables on the control flow on SSIS i used the following SQL statements.

These SQL statements update the dimension tables using the staging table. They also check to insert only the data that hasn’t been inserted into the dimension table before.

```
-- Update Car Dimension
INSERT INTO Car_dim (CarId, Model, Engine, Transmission, Color, Price, Body_Style, Company)
SELECT DISTINCT
    s.Car_id,
    s.Model,
    s.Engine,
    s.Transmission,
    s.Color,
    s.Price,
    s.[Body Style],
    s.Company
FROM
    staging s
WHERE NOT EXISTS (
    SELECT 1
    FROM Car_dim cd
    WHERE cd.CarId = s.Car_id
);


-- Update Dealer Dimension
INSERT INTO Dealer_dim (DealerNo, Dealer_Name, Dealer_Region)
SELECT DISTINCT
    s.Dealer_No,
    s.Dealer_Name,
    s.Dealer_Region
FROM
    staging s
WHERE NOT EXISTS (
    SELECT 1
    FROM Dealer_dim dd
    WHERE dd.[DealerNo] = s.Dealer_No AND dd.[Dealer_Name] = s.[Dealer_Name] AND dd.[Dealer_Region] = s.[Dealer_Region]
);



-- Update Customer Dimension
INSERT INTO Customer_dim (Customer_Name, Gender, Annual_Income)
SELECT DISTINCT
    s.[Customer Name],
    s.Gender,
    s.[Annual Income]
FROM
    staging s
WHERE NOT EXISTS (
    SELECT 1
    FROM Customer_dim cd
    WHERE cd.Customer_Name = s.[Customer Name] AND cd.Annual_Income = s.[Annual Income]
);

-- Update Date Dimension
INSERT INTO Date_dim (label_date)
SELECT DISTINCT
    s.Date
FROM
    staging s
WHERE NOT EXISTS (
    SELECT 1
    FROM Date_Dim dd
    WHERE dd.label_date = s.Date
);
```

The CarSalesFact table is the fact table and contains the ids for the Date, Car, Customer and Dealer from the staging table, as well as the Phone column for the phone associated with the sale. The Date, Car, Customer and Dealer columns are all foreign keys.

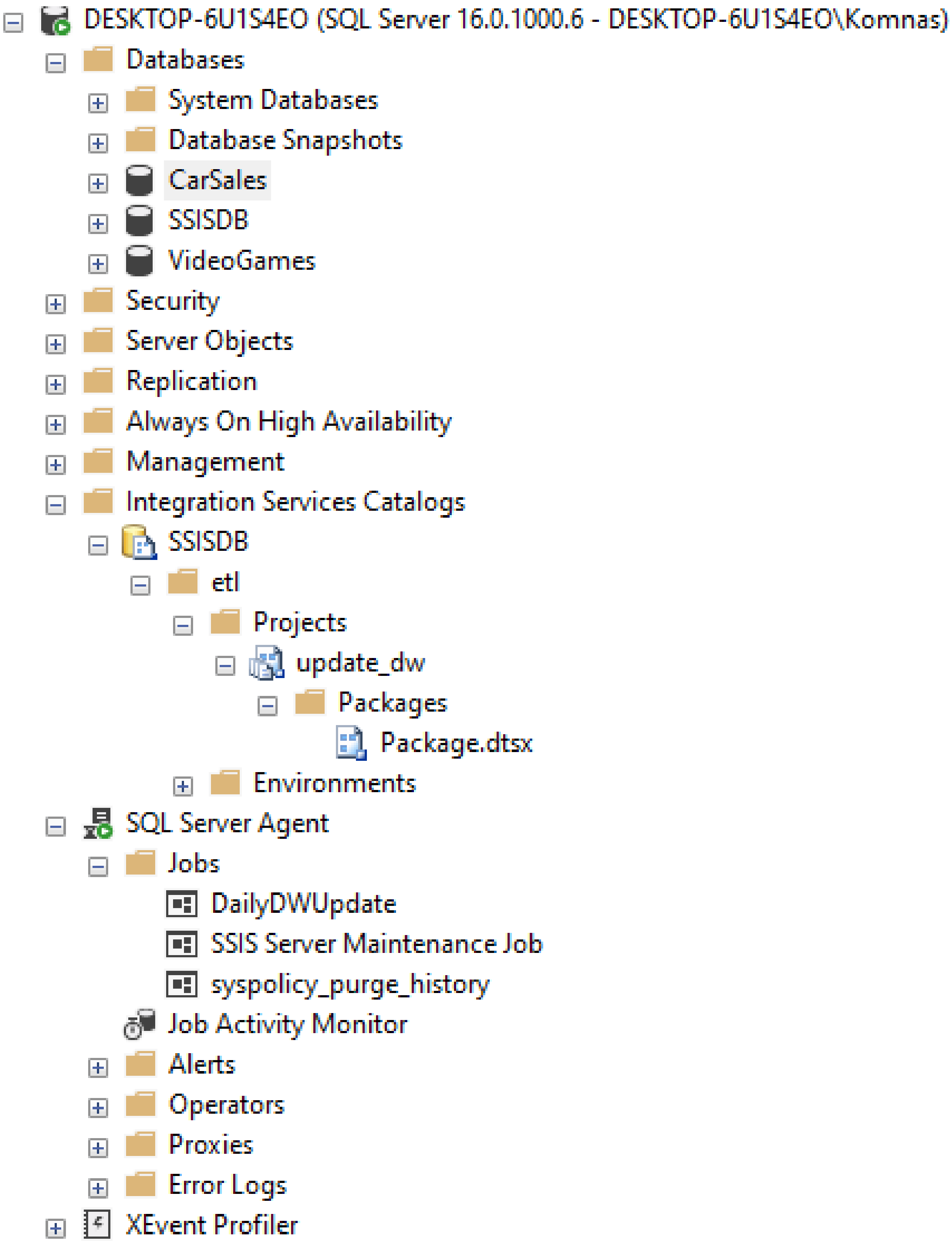
To update the fact table I used the following SQL statement.

This SQL statement inserts the ids of the Date, Car, Customer and Dealer from the respective dimensions, as well as the Phone from the staging table. It also ensures that only the data that hasn't been inserted into the dimension table before are inserted.

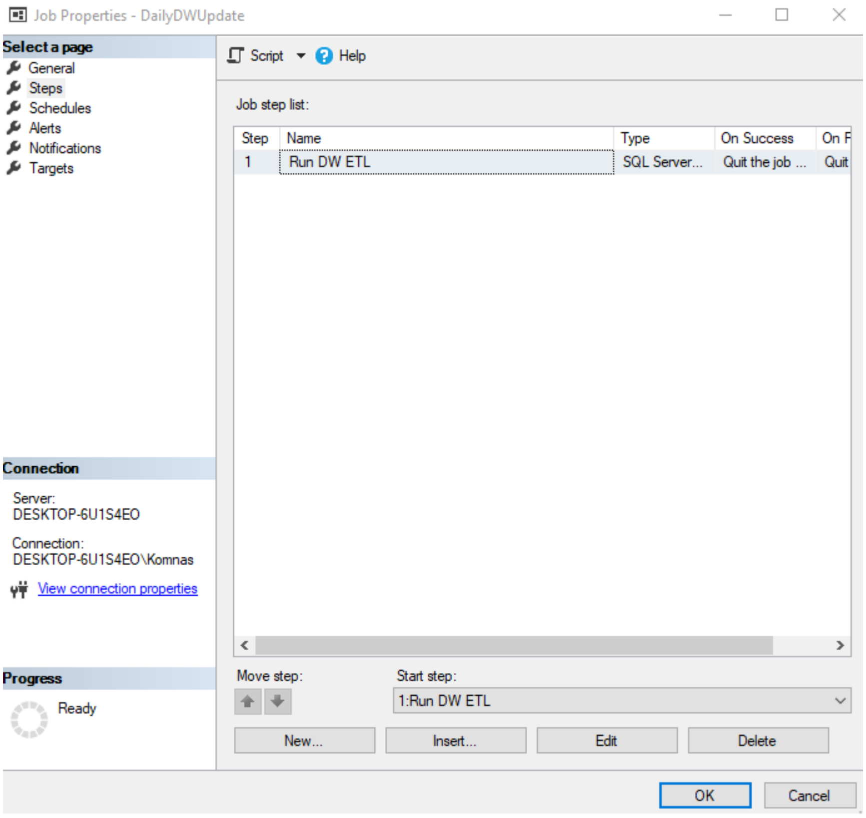
```sql
 -- Update CarSalesFact Table
INSERT INTO CarSalesFact
SELECT [Date_dim].DateId AS [Date],
[Car_dim].CarId AS [Car],
[Customer_dim].CustomerId AS [Customer],
[Dealer_dim].DealerId AS [Dealer],
[staging].[Phone]
FROM staging
INNER JOIN [Date_dim] on [staging].[Date] = [Date_dim].[label_date]
INNER JOIN [Car_dim]
on [staging].[Model] = [Car_dim].[Model] AND [staging].[Engine] = [Car_dim].[Engine] AND [staging].[Transmission] = [Car_dim].[Transmission]
AND [staging].[Color] = [Car_dim].[Color] AND [staging].[Price] = [Car_dim].[Price] AND [staging].[Body Style] = [Car_dim].[Body_Style]
AND [staging].[Company] = [Car_dim].[Company]
INNER JOIN [Customer_dim] on [staging].[Customer Name] = [Customer_dim].[Customer_Name] AND [staging].[Gender] = [Customer_dim].[Gender]
AND [staging].[Annual Income] = [Customer_dim].[Annual_Income]
INNER JOIN [Dealer_dim] on [staging].[Dealer_Name] = [Dealer_dim].[Dealer_Name] AND [staging].[Dealer_Region] = [Dealer_dim].[Dealer_Region]
AND [staging].[Dealer_No] = [Dealer_dim].[DealerNo]
WHERE NOT EXISTS (
    SELECT 1
    FROM CarSalesFact csf
    WHERE [Date_dim].DateId = csf.[Date]
      AND [Car_dim].CarId = csf.[Car]
      AND [Customer_dim].CustomerId = csf.[Customer]
      AND [Dealer_dim].DealerId = csf.[Dealer]
      AND [staging].[Phone] = csf.[Phone]
);
```

After that I deployed the SSIS package on the SQL server and then I set up a job to run the ETL process every day at 2 A.M.

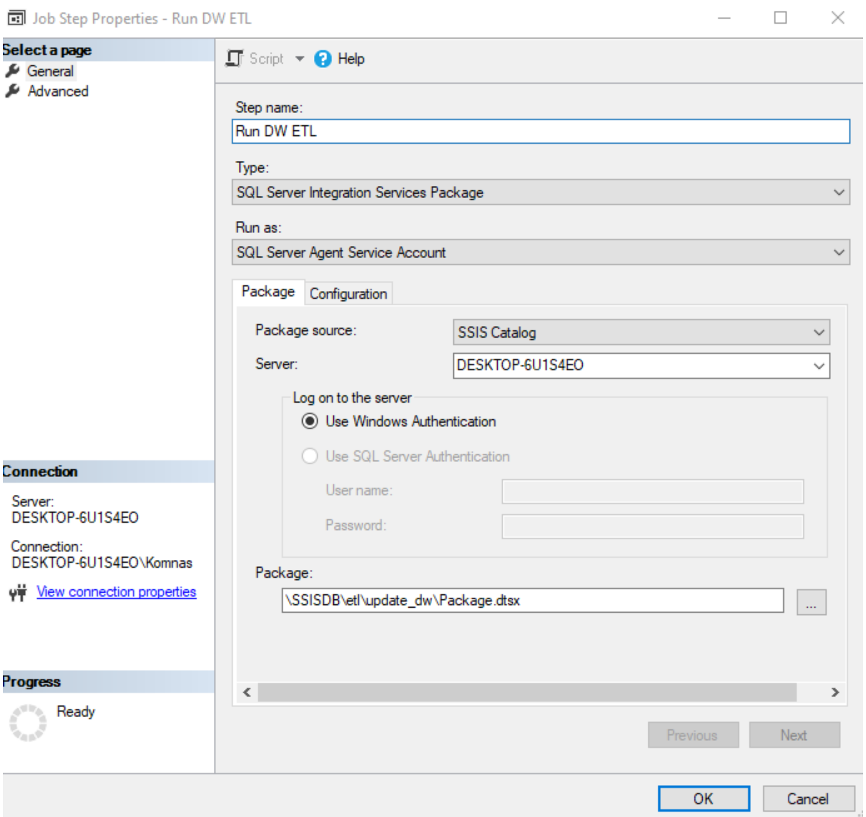
Below is an image of the SQL Server folders after the deployment of the SSIS package and the scheduling of the job. As you see there's the SSIS package named Package.dtsx and a job called DailyDWUpdate. VideoGames is just a database I used before for testing.



Below is an image showing the steps of the job to run the Data Warehouse ETL process, which is just one.



Below is an image showing the settings of the step.

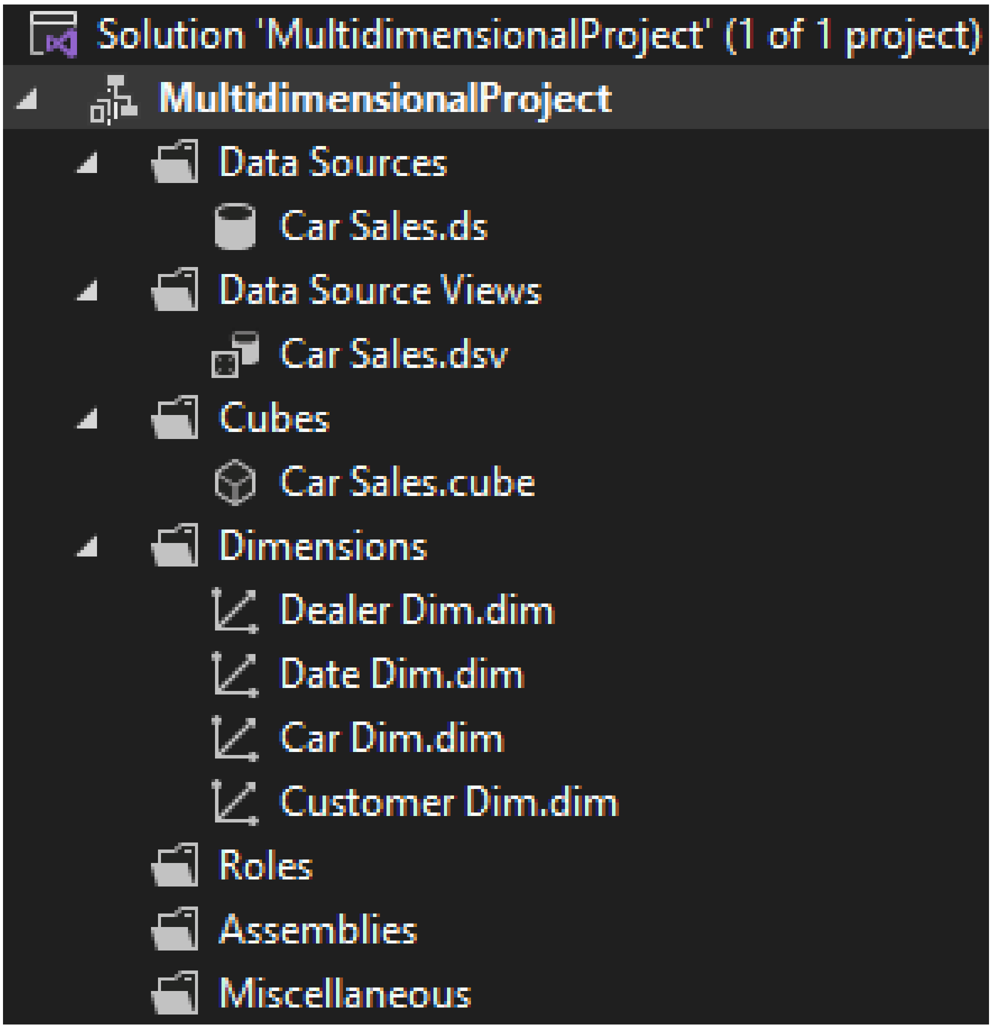


Cube

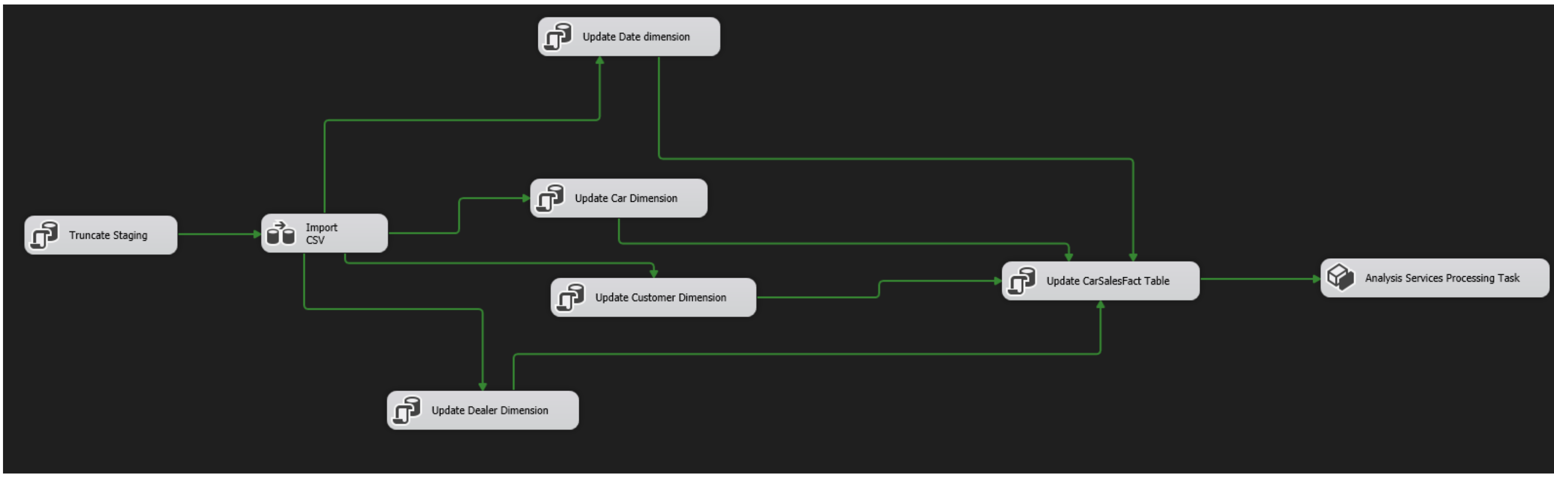
The next step to be done was to create the cube. For this reason, I set up an Analysis Services project on Visual Studio 2022 using the Microsoft Analysis Services Projects extension.

As a first step I created a Data Source which would be the SQL Server. After that I created a Data Source View using the fact table and the dimension tables. Then I created the cube. Lastly, I added the label and other columns as attributes to the dimension tables and processed the cube.

The Analysis Services Project structure looked like this after the creation of the cube.



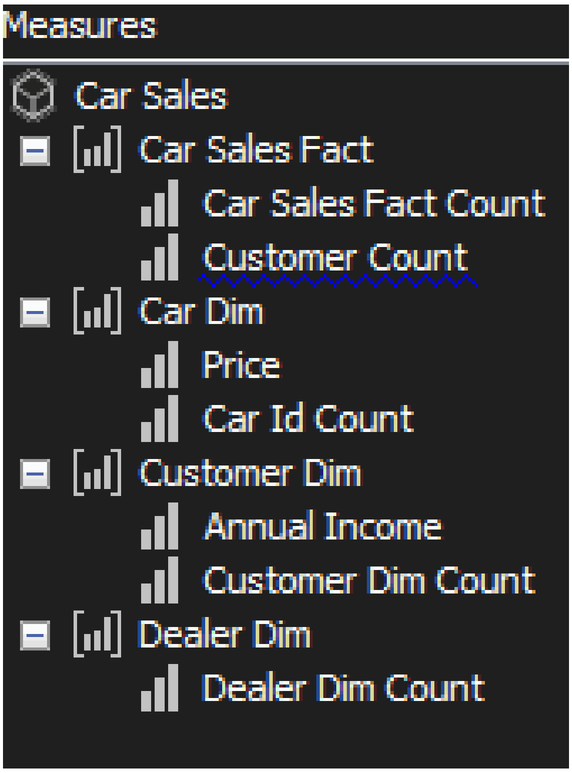
The next step was to add the processing of the cube to the ETL process like in the image below and deploy the package to the SQL Server.



VISUALIZATIONS

For the visualizations I created some measures and calculations.

Here’s the total measures:



Here’s the total calculations:

		Command
1		CALCULATE
2		[AVG_ANNUAL_INCOME]
3		[Average Price]

For the AVG_ANNUAL_INCOME calculation the expression is :

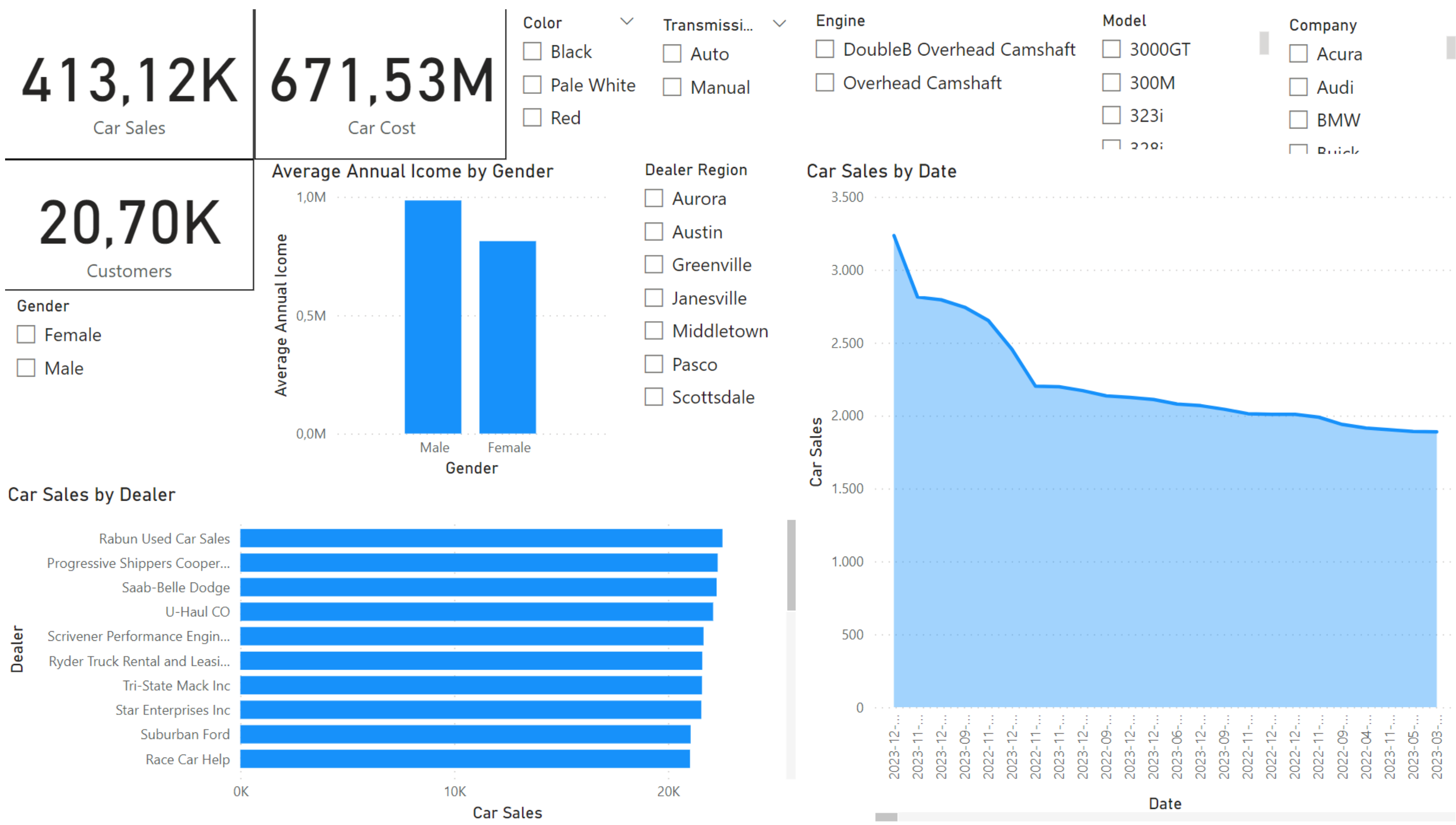
```
[Measures].[Annual Income]/[Measures].[Customer Dim Count]
```

For the Average Price calculation the expression is :

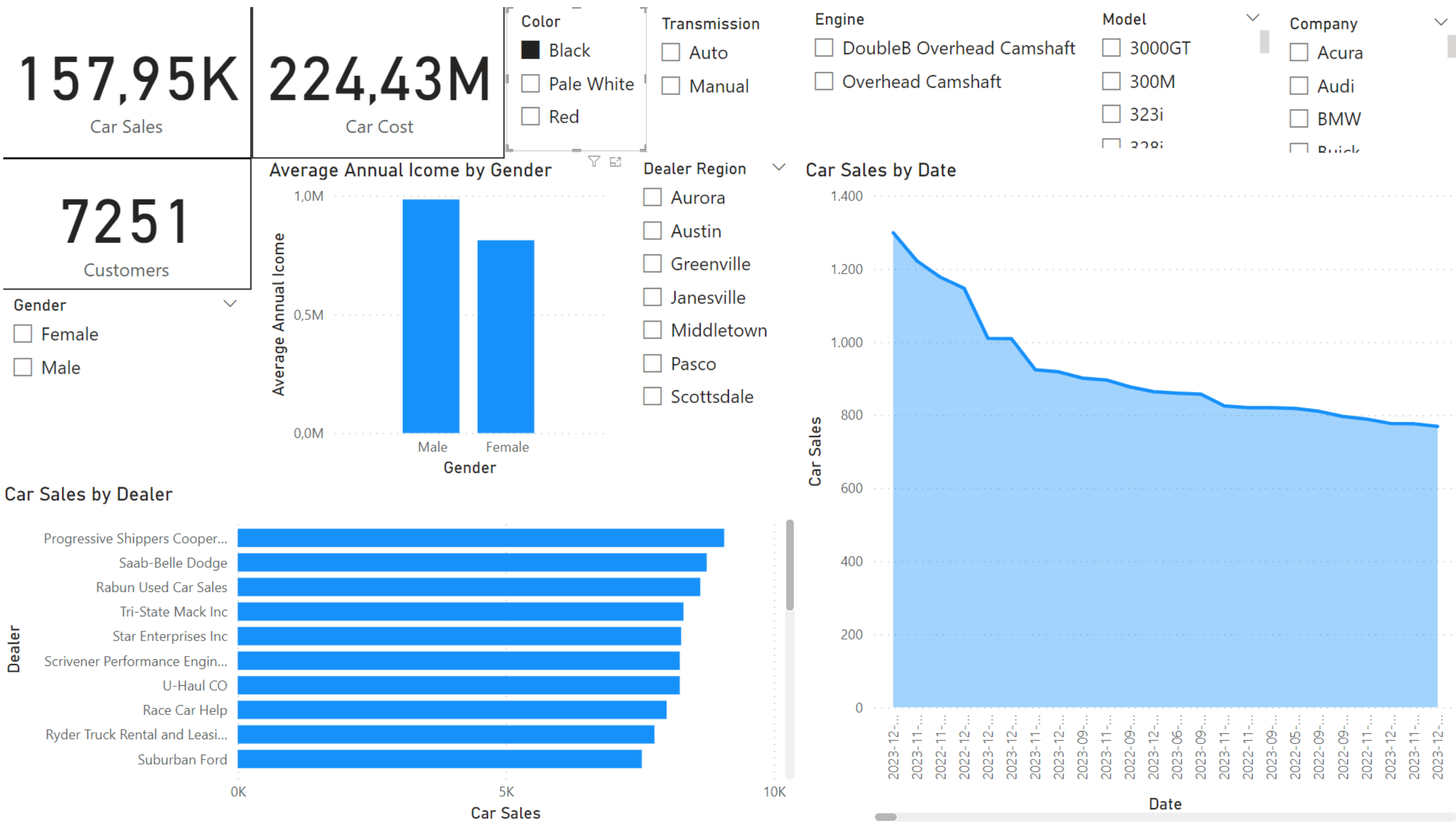
```
[Measures].[Price]/[Measures].[Car Id Count]
```

I used Power BI to create the visualizations.

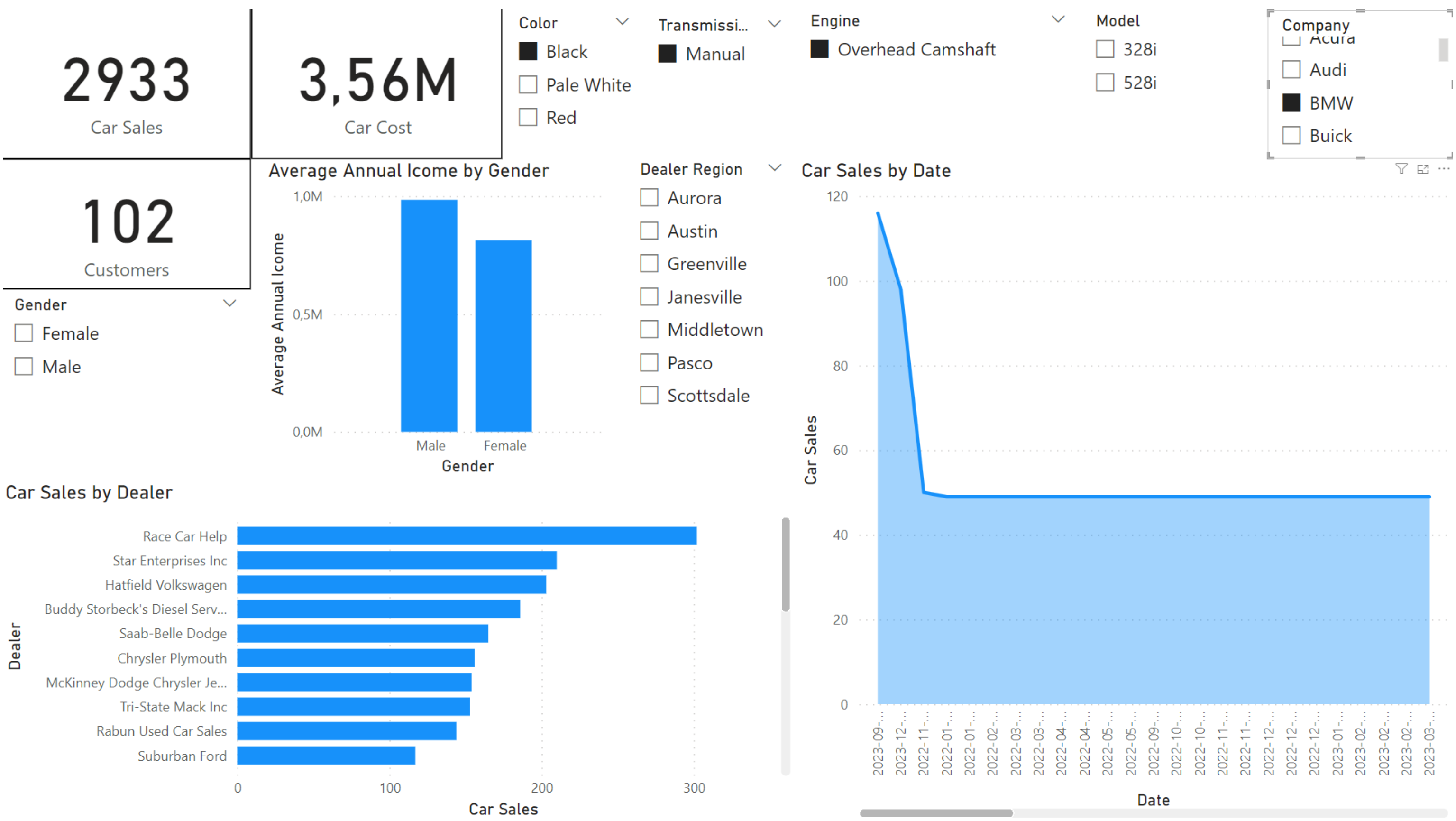
Below is a visualization that shows the number of car sales, the total cost of the cars, the number of the customers, the average annual income by gender, the number of car sales by date and the number of car sales by dealer.



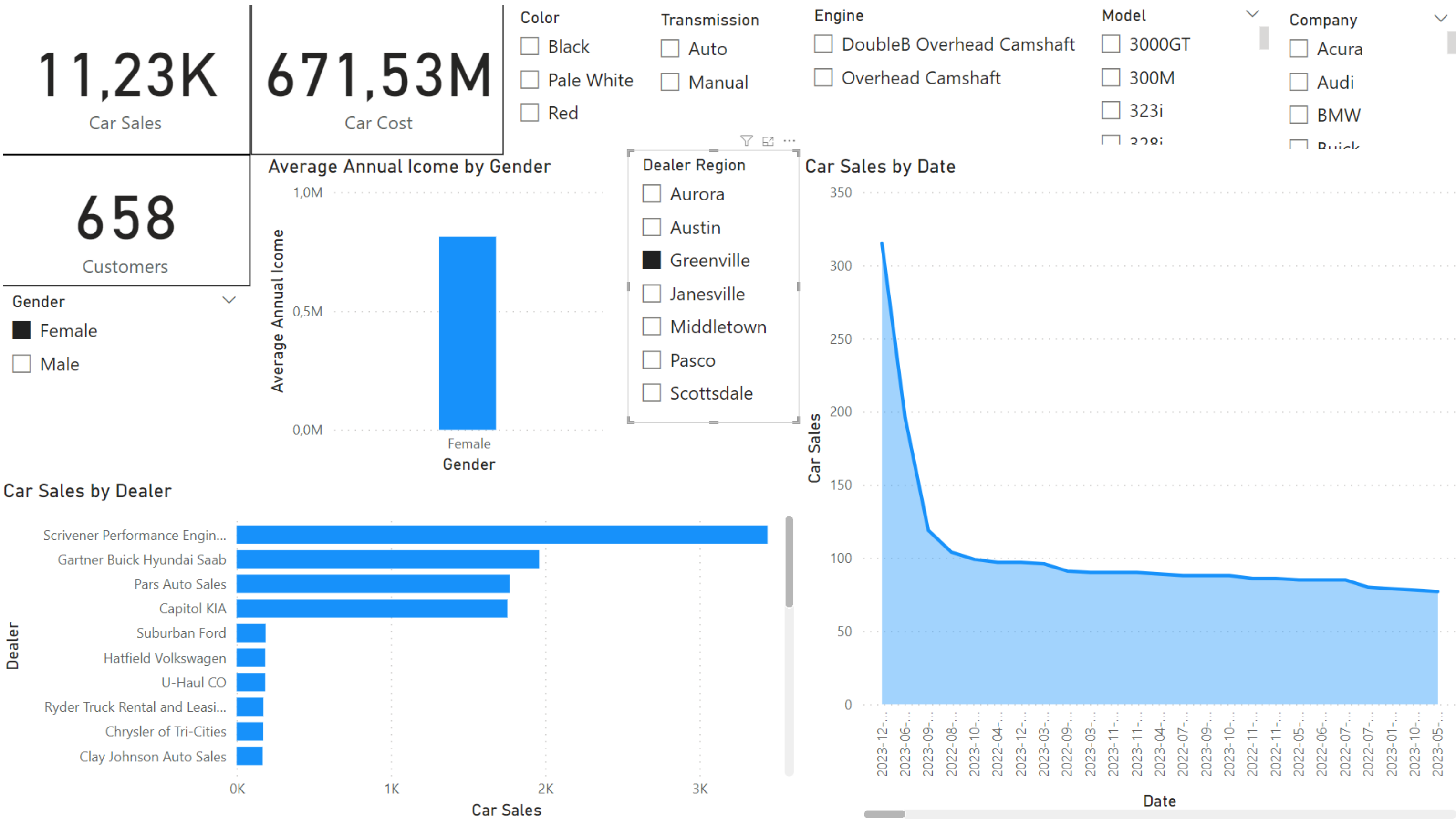
I can select to sort the visualizations by Color, in this case by Black



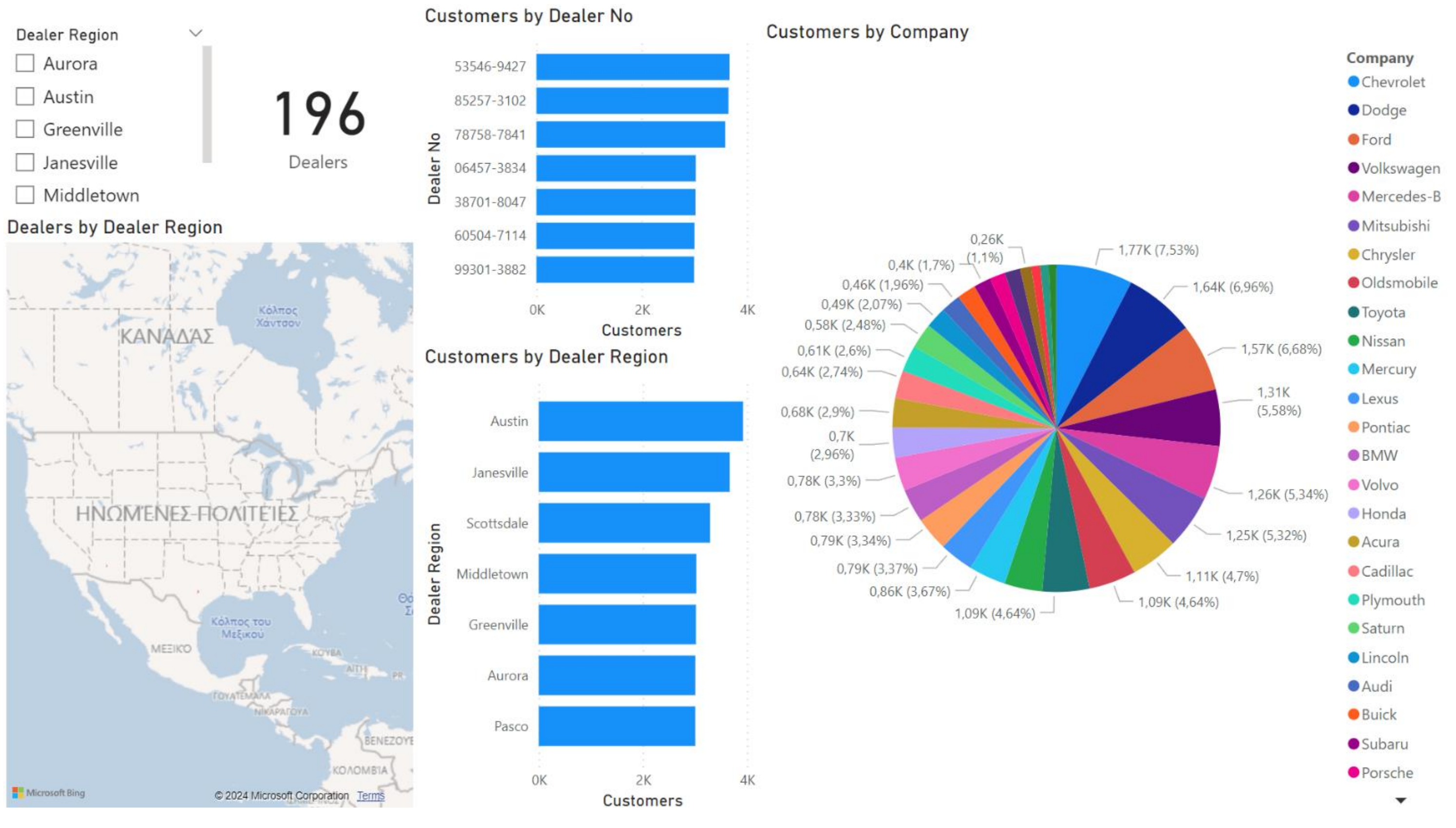
I can also select to sort the visualizations by color, transmission, engine and company, in this case Black, Manual, Overhead Camshaft and BMW.



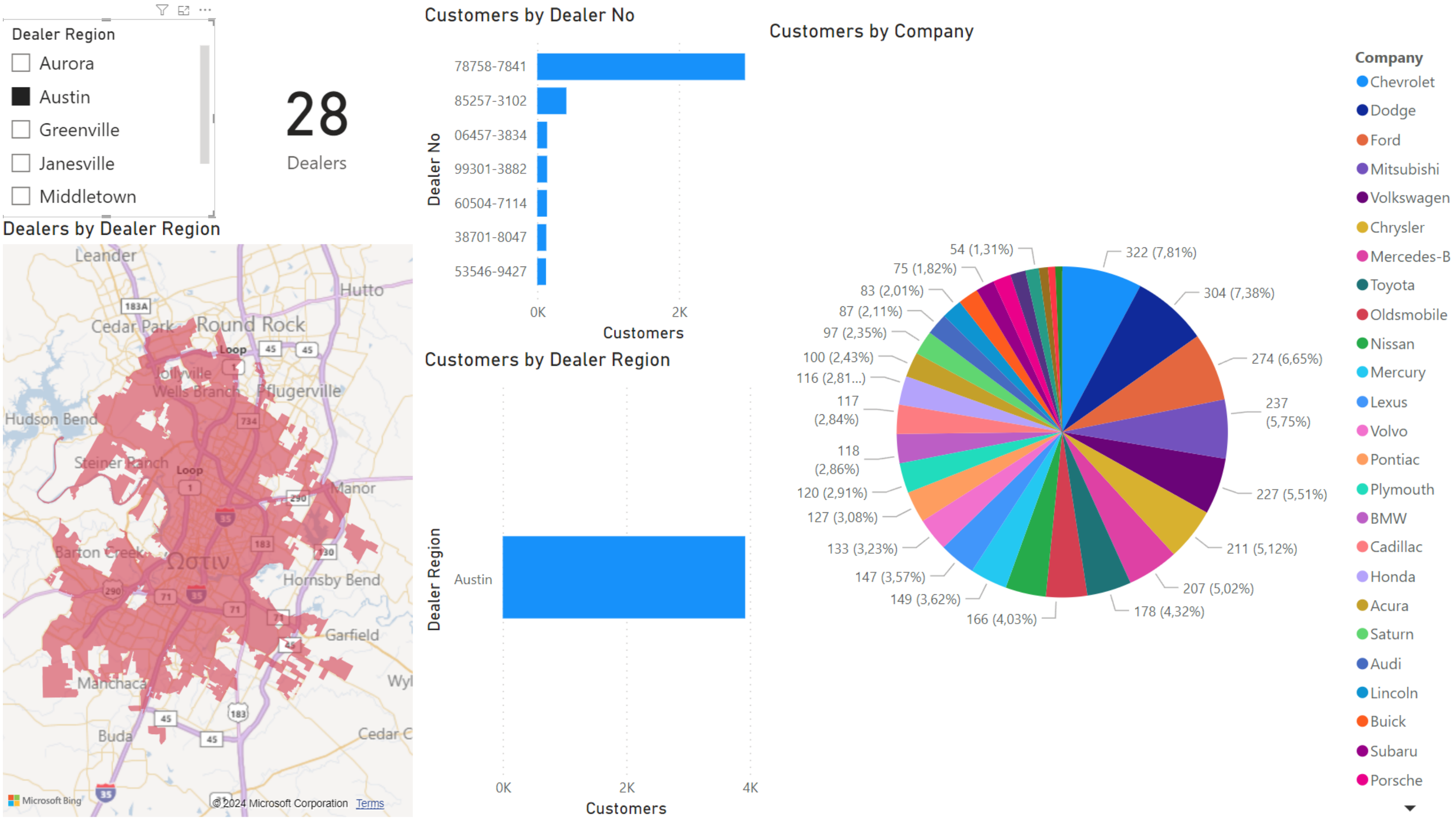
I can also select to sort the visualizations by gender and dealer, in this case Female and Greenville.



Below are other visualizations about the number of customers by dealer number, number of customers by dealer region and number of customers by car company. These visualizations can be sorted by region.

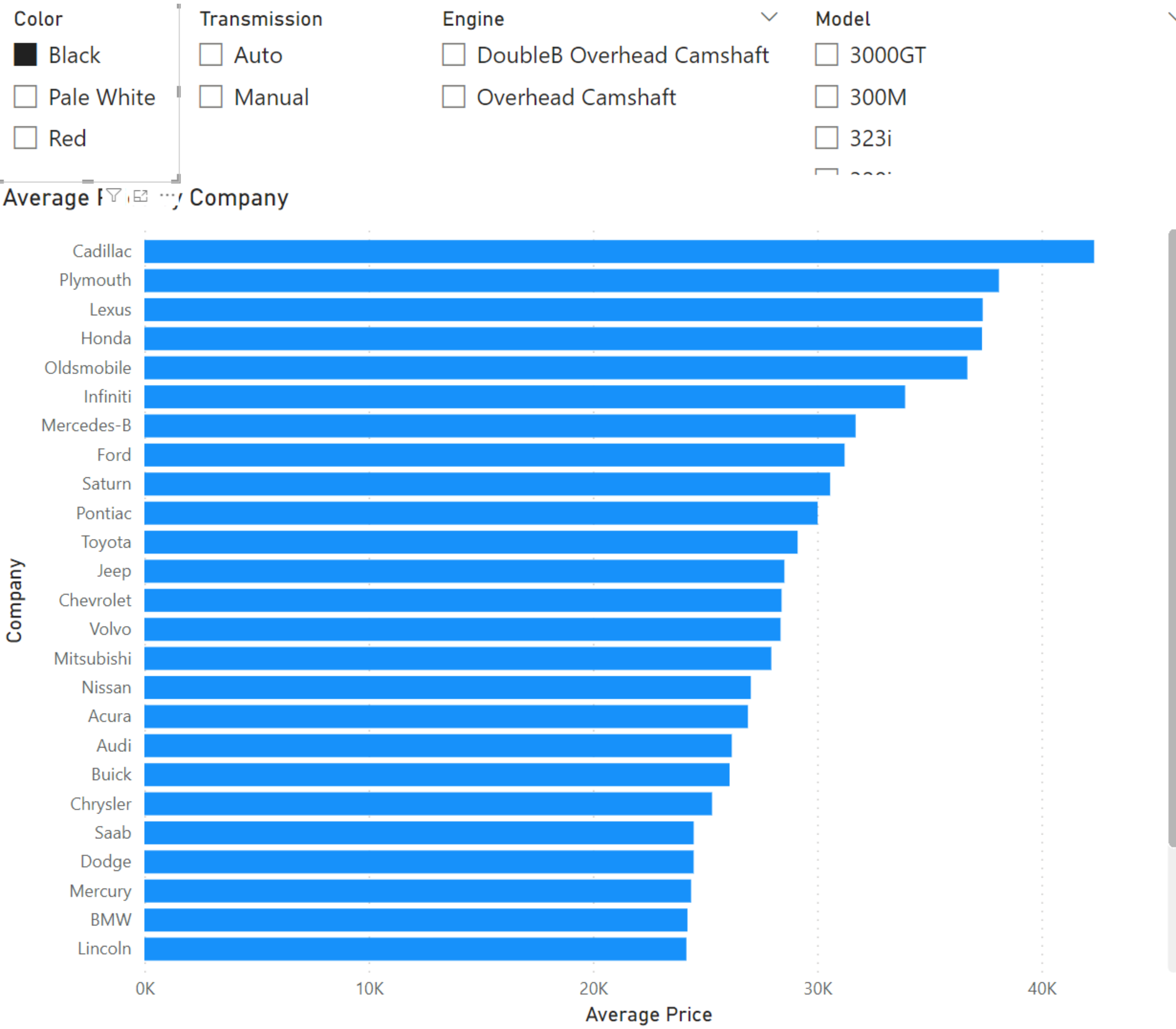


Here are the visualizations sorted by the dealer region Austin.



Below is another visualization showing the average price of the cars by company. This can be sorted by color, Transmission, Engine and Model.

Below is the visualization sorted by the black color.



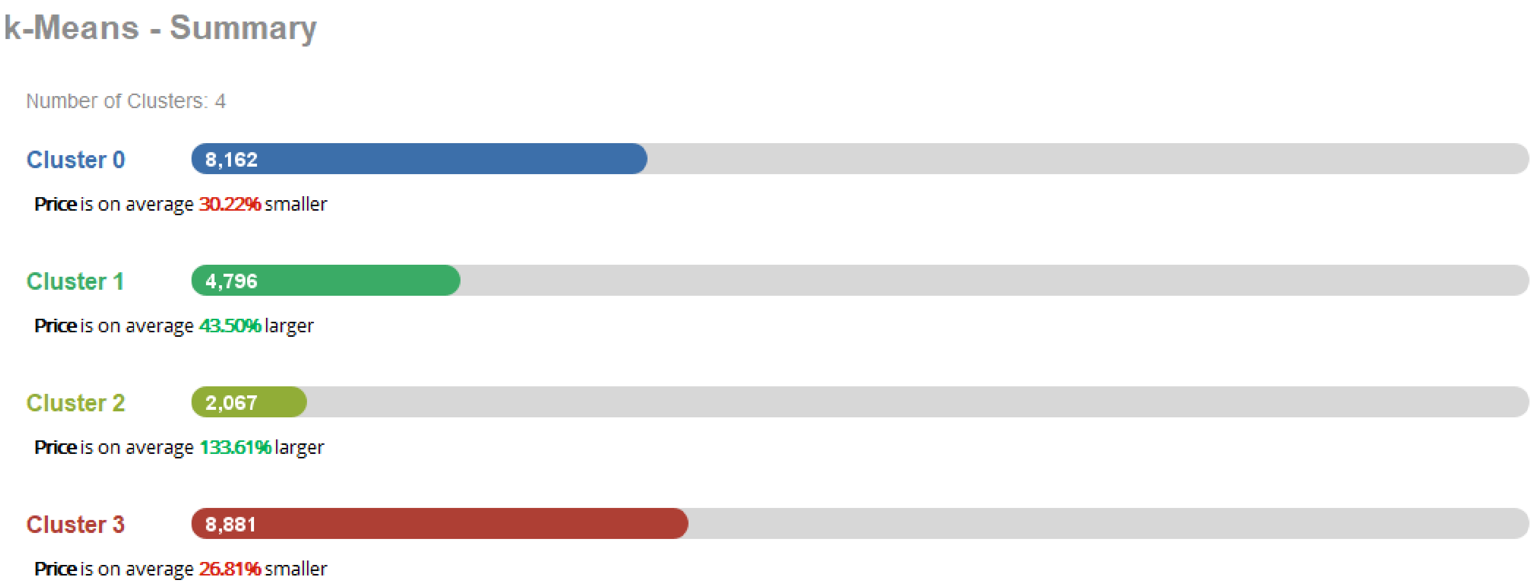
ANALYSIS MODELS

To create analysis models using data mining I used RapidMiner Studio. I imported the data and used the Auto Model view to create clustering and classification models. The Auto Model view creates the data mining models automatically, without the need to design the process these models are calculated.

Clustering

For the clustering model I used the annual income and body style columns as input. I also used the k-means algorithm and created 4 clusters.

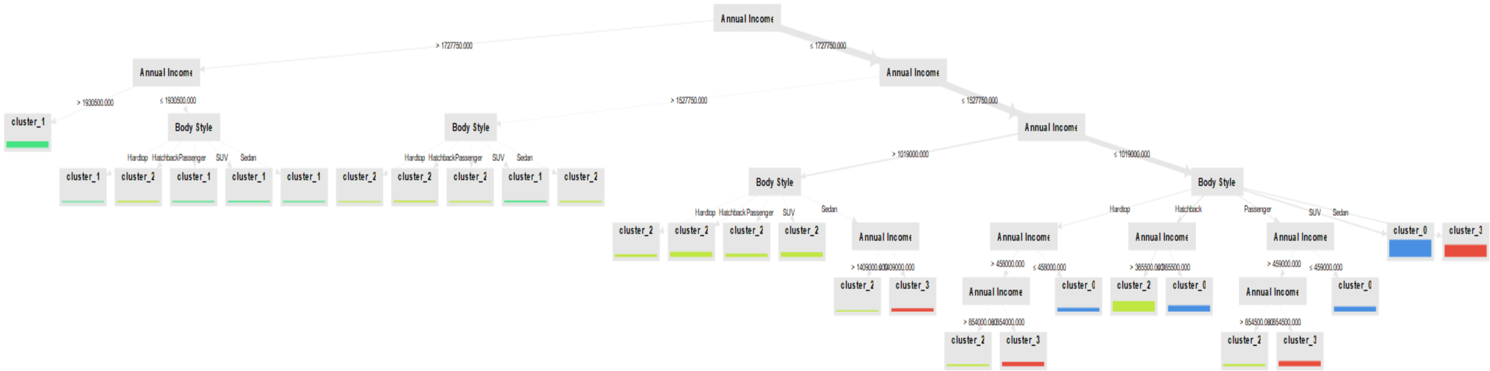
Below is a summary visualization of the k-means clustering model. There's information on the price value on average.



As you can see, there's 4 clusters from 0 to 3. There's also an indication showing how much smaller or larger on average is the price on each cluster.

Below is a representation of the k-means cluster tree which shows how the clusters are created, in a tree form.

k-Means - Cluster Tree



Below is a table showing the cluster of each combination of Body Style and annual income.

k-Means - Clustered Data

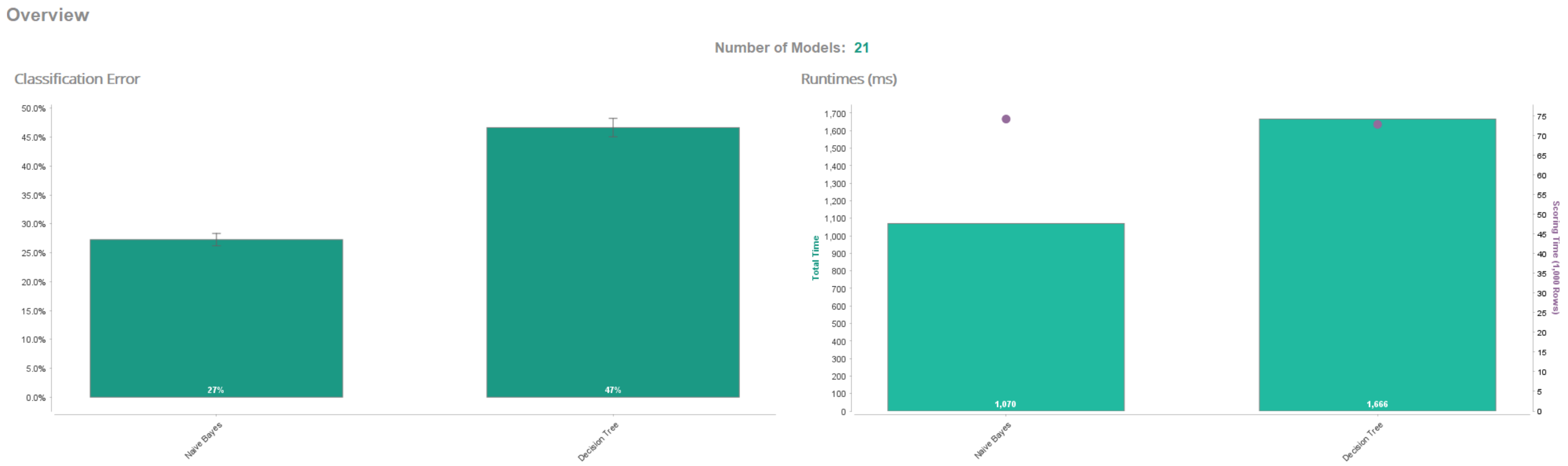
Row No.	id	cluster	Annual Income	Body Style
1	1	cluster_0	13500.000	SUV
2	2	cluster_2	1480000.000	SUV
3	3	cluster_2	1035000.000	Passenger
4	4	cluster_0	13500.000	SUV
5	5	cluster_2	1465000.000	Hatchback
6	6	cluster_2	850000.000	Hatchback
7	7	cluster_2	1600000.000	Passenger
8	8	cluster_0	13500.000	Passenger
9	9	cluster_3	815000.000	Hardtop
10	10	cluster_0	13500.000	Passenger
11	11	cluster_0	13500.000	SUV
12	12	cluster_0	13500.000	Hardtop
13	13	cluster_0	885000.000	SUV
14	14	cluster_0	13500.000	Hatchback
15	15	cluster_3	722000.000	Sedan
16	16	cluster_2	746000.000	Hatchback
17	17	cluster_2	535000.000	Hatchback
18	18	cluster_3	570000.000	Passenger
19	19	cluster_0	685000.000	SUV
20	20	cluster_3	455000.000	Sedan
21	21	cluster_3	13500.000	Sedan
22	22	cluster_1	2500000.000	Hardtop
23	23	cluster_3	585000.000	Hardtop
24	24	cluster_2	920000.000	Passenger
25	25	cluster_3	672000.000	Passenger
26	26	cluster_0	801250.000	SUV
27	27	cluster_3	820000.000	Passenger
28	28	cluster_2	791000.000	Hatchback
29	29	cluster_0	13500.000	Hatchback
30	30	cluster_2	1020000.000	Hatchback
31	31	cluster_3	210000.000	Sedan
32	32	cluster_2	750000.000	Hatchback

Classification

For the classification model I predicted the column Transmission and used the model, body style and price columns as input. I also used the Naive Bayes algorithm for this model.

The Naive Bayes model computes the probability of the Transmission of a specific car sale being auto or manual, based on the model, body style and price of the car. Based on this probability the Transmission is predicted and each car sale is classified based on the prediction.

Below are some statistics about the Naive Bayes model as well as in comparison with the Decision Tree model.

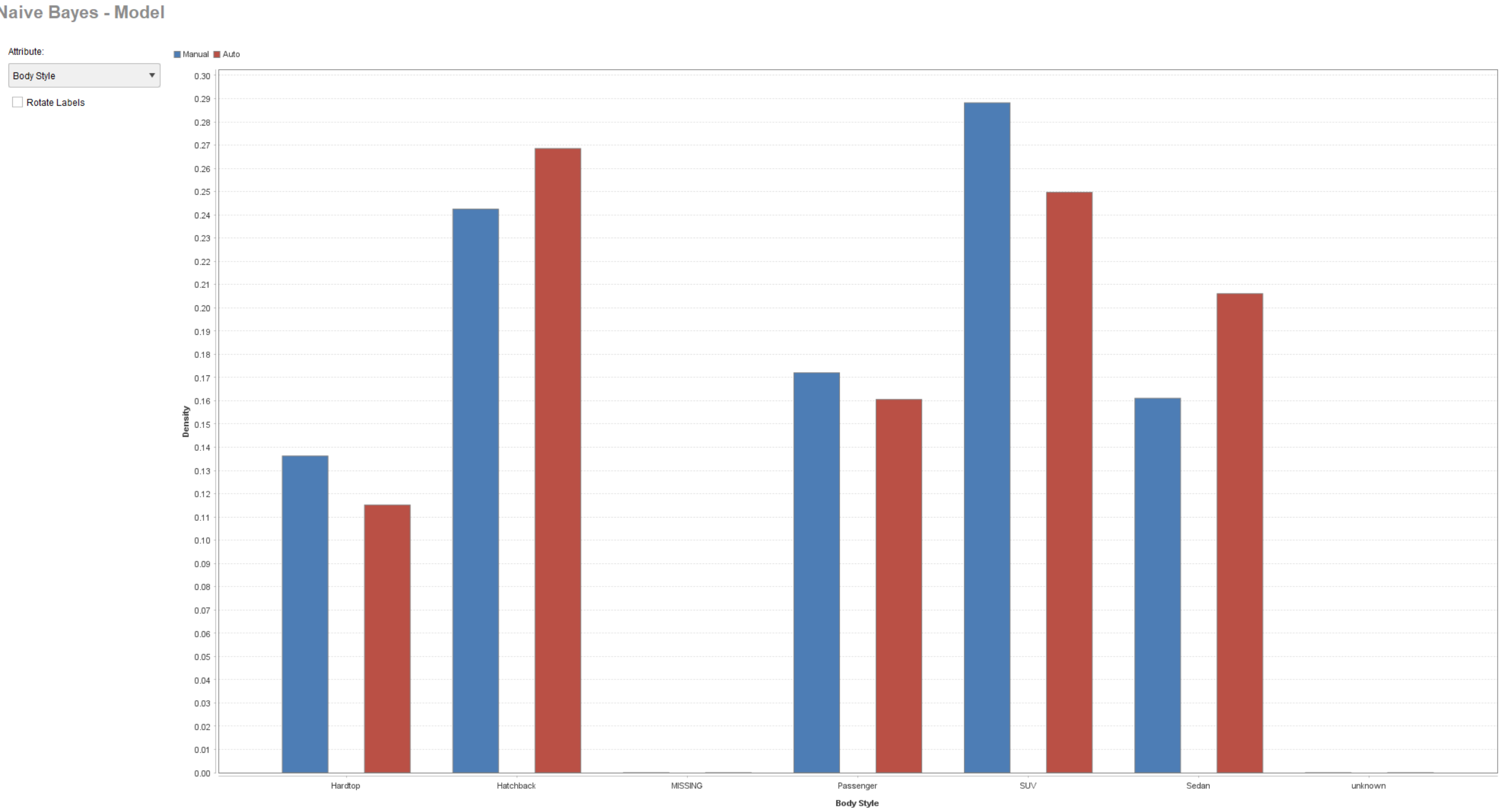


Classification Error ▾

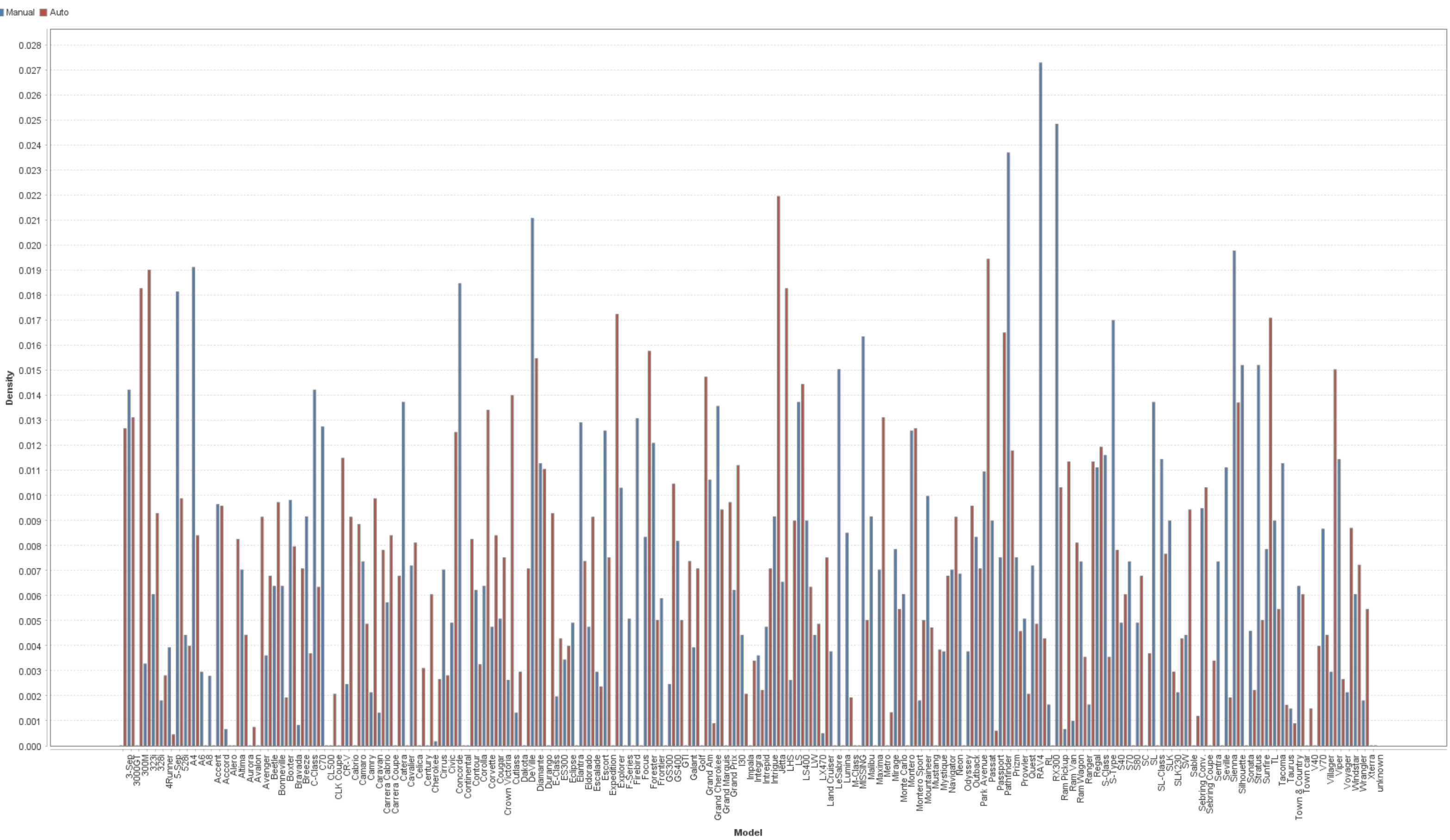
Model	Classification Error	Standard Deviation	Gains	Total Time	Training Time (1,000 Rows)	Scoring Time (1,000 Rows)
Naive Bayes	27.3%	± 1.1%	2,768	1 s	2 ms	74 ms
Decision Tree	46.6%	± 1.6%	116	2 s	2 ms	73 ms

As we see from the stats the Naive Bayes model is more precise on the predictions and was executed in less time than the Decision Trees model.

Below are some other stats that shows the density per attribute, in this case the Body Style attribute.

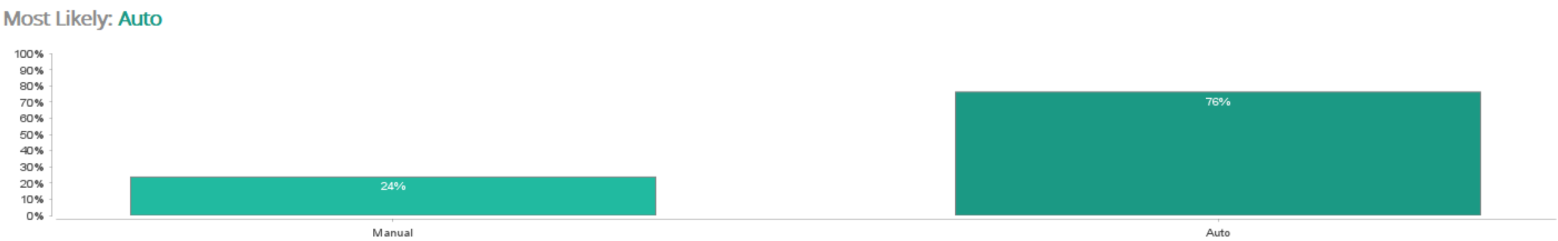


Below are the same stats that shows the density per attribute, in this case the car model attribute.

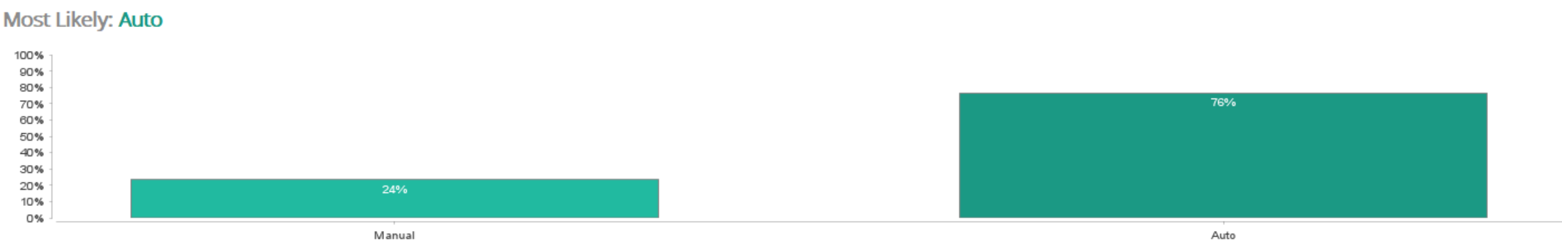


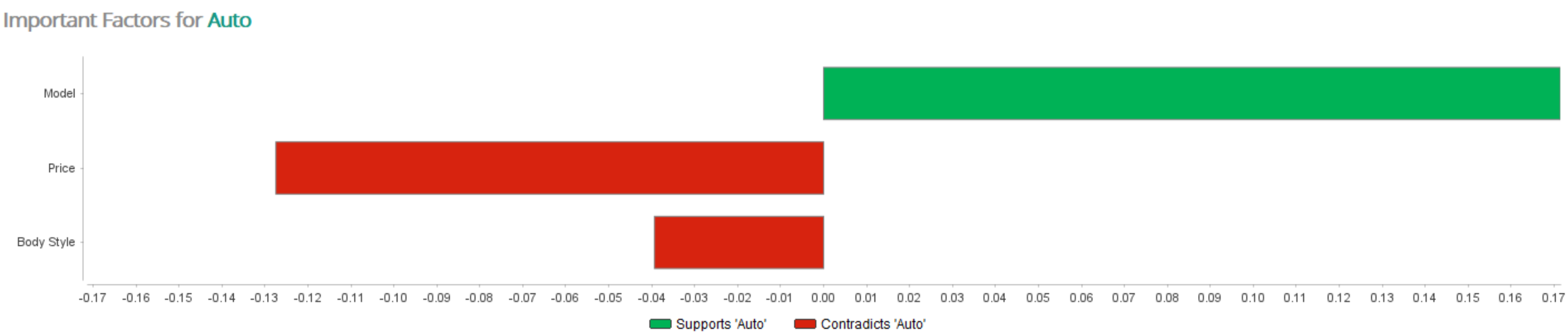
Below are visualizations showing how likely the prediction on the transmission is to be manual or auto. The visualizations are based on the body style, model and price filters. There's also a visualization showing the important factors and how important they are on a scale, for auto or manual. Based on the filters the visualizations change accordingly.

For the Body Style passenger, model 323i and price 8023 these are the visualizations.

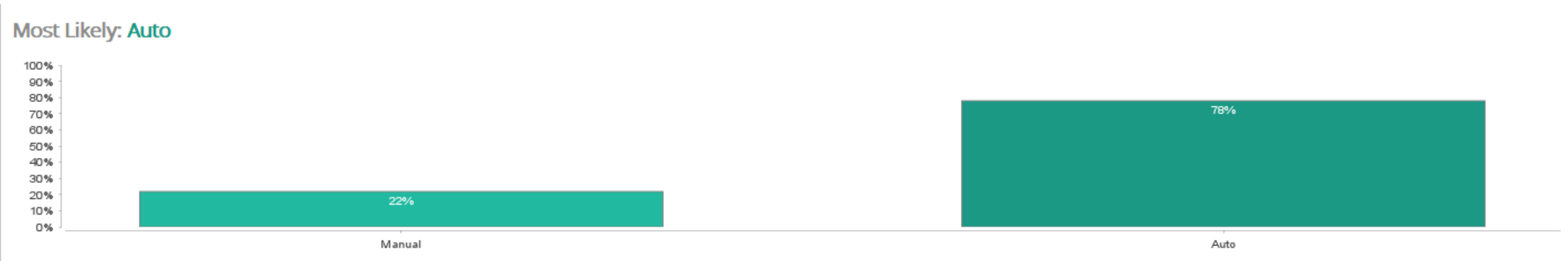


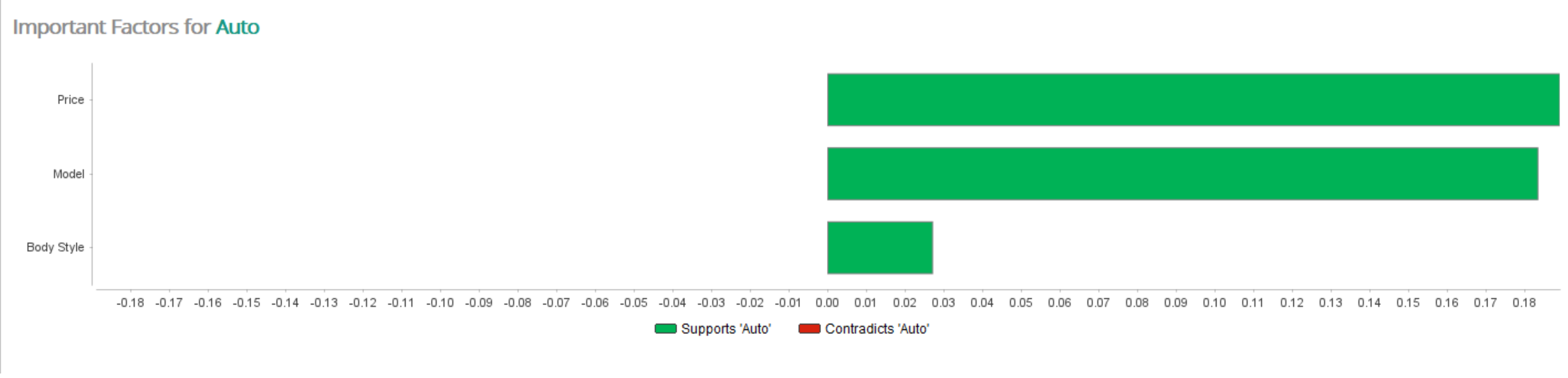
If the price is adjusted to 45774 the visualizations change accordingly.





For the Body Style Sedan, model 323i and price 7568 these are the visualizations.





Below are some statistics about the performance of the Naive Bayes model, regarding accuracy, classification error etc.

Naive Bayes - Performance

Profits

Profits from Model: 3,106   Profits for Best Option (Auto): 338   Gain: 2,768   Show Costs / Benefits...

Performances

Criterion	Value	Standard Deviation
Accuracy	72.7%	± 1.1%
Classification Error	27.3%	± 1.1%
AUC	81.1%	± 1.2%
Precision	75.7%	± 2.3%
Recall	70.7%	± 1.6%
F Measure	73.1%	± 1.5%
Sensitivity	70.7%	± 1.6%
Specificity	75.0%	± 1.0%

Confusion Matrix

	true Manual	true Auto	class precision
pred. Manual	2435	1051	69.85%
pred. Auto	811	2533	75.75%
class recall	75.02%	70.68%	

Below are the predictions of the Naive Bayes model. There's a column about the actual transmission as well as the prediction of the transmission. There are also 2 columns showing the confidence of model about the transmission being manual or auto.

Naive Bayes - Predictions

Row No.	Transmission	prediction(Transmission)	confidence(Manual)	confidence(Auto)	cost	Model	Body Style	Price
1	Auto	Manual	0.526	0.474	0.051	Durango	SUV	19000
2	Manual	Manual	0.686	0.314	0.371	Corolla	Passenger	14000
3	Auto	Manual	0.576	0.424	0.152	Escort	Passenger	15000
4	Manual	Auto	0.389	0.611	0.223	Sebring Coupe	Sedan	26000
5	Auto	Auto	0.416	0.584	0.168	Accord	Sedan	21000
6	Manual	Auto	0.320	0.680	0.359	4Runner	Sedan	25000
7	Manual	Manual	0.587	0.413	0.174	A4	Hardtop	12000
8	Auto	Auto	0.234	0.766	0.533	Viper	SUV	31250
9	Manual	Manual	0.575	0.425	0.150	LW	Hatchback	13000
10	Auto	Auto	0.421	0.579	0.158	Accord	Sedan	19000
11	Manual	Manual	0.715	0.285	0.431	Civic	Hatchback	43000
12	Auto	Manual	0.649	0.351	0.297	S40	Sedan	42000
13	Auto	Manual	0.692	0.308	0.383	Park Avenue	Hatchback	61000
14	Auto	Manual	0.531	0.469	0.062	Montero Sport	SUV	39000
15	Auto	Auto	0.174	0.826	0.653	Sentra	Passenger	16000
16	Manual	Auto	0.389	0.611	0.223	Sebring Coupe	Sedan	26000
17	Manual	Manual	0.892	0.108	0.785	S80	Sedan	21000
18	Manual	Manual	0.892	0.108	0.785	Lumina	Passenger	42500
19	Auto	Auto	0.499	0.501	0.002	Montero Sport	Hatchback	45001
20	Auto	Manual	0.636	0.364	0.273	Stratus	Hatchback	31000
21	Manual	Manual	0.671	0.329	0.343	C70	Hatchback	17000
22	Manual	Manual	0.693	0.307	0.387	C-Class	Hatchback	17000
23	Auto	Manual	0.692	0.308	0.383	Park Avenue	Hatchback	61000
24	Auto	Manual	0.532	0.468	0.063	Diamante	Hatchback	21000
25	Manual	Manual	0.790	0.210	0.580	S-Type	Passenger	45000
26	Auto	Auto	0.174	0.826	0.653	Contour	Sedan	62000
27	Auto	Auto	0.304	0.696	0.392	Jetta	Passenger	22700
28	Auto	Manual	0.565	0.435	0.131	Montero Sport	SUV	45000
29	Manual	Auto	0.476	0.524	0.048	Town car	Hatchback	17000
30	Manual	Manual	0.892	0.108	0.785	Focus	Hardtop	49000
31	Manual	Manual	0.730	0.270	0.461	A6	SUV	15000
32	Auto	Auto	0.379	0.621	0.243	Corvette	SUV	45000
33	Manual	Manual	0.647	0.353	0.294	Impala	Hatchback	22001