Grayson Nickel
10/19/2025

Cars of 2025

Project 3: Regression

The problem

For this project, I plan to train a model to predict the price of a car based on different specs of said car and then use that model on my dataset at large through use of linear regression. I also plan to experiment with this model by trying to train on different features and how well they help predict the exact prices. To start off, I will use information like horsepower and 0-100mph time. But I plan to also try out different features like torque and top speed. Just like my last two projects, I will be using the "Cars Datasets (2025)" by Abdul Malik on kaggle.

Regression and how it works

Linear regression in terms of datamining is a way to predict a continuous numerical outcome by modeling its relationship with the other variables in the dataset. So, essentially in my case, I will train my model to predict price (a column from the dataset) based on a number of other columns from the dataset and their relationships to the value of price so that my model can reliably predict the price of a car in the cars 2025 dataset, and maybe even future datasets for 2026.

Experiment 1: Data understanding

Before diving into my first experiment, I need to understand what I want and how to achieve it. I want to predict prices, simple as that. To achieve it, I will look at the relationship between price and two other columns of my dataset, horsepower, and 0-100mph time. Based on what I have seen, these things seem to correlate well with how expensive a car is. There is no statistical analysis on my part, just observations from using this dataset two other times.

Experiment 1: Pre-processing

To kick things off, like always, I set up my drive on google colab and imported my dataset into the notebook for easy access. Once that was complete, seeing as I am not using the provided houses dataset, I needed to manually split my dataset 80/20 so I have a decent sample size to work with and train my model on. After the split, the data needed to be cleaned as each column stores strings with symbols by default that make doing math more difficult.

Experiment 1: Modeling

Once the data was cleaned, everything was fairly straightforward from there. I trained my regression model and then ran some tests, looking at the RMSE as well as a small sample comparing the actual prices vs the predicted prices and the difference between them so I have an idea of how accurate the model is. Overall, I'm pretty happy with the accuracy of the model considering how few columns I used and the fairly small size of the sample used for training. You may notice that I did not use typical visualization for this experiment as I couldn't see how any sort of graph would be helpful. Instead I created tables showing actual vs predicted prices

as well as their margins of errors. This provided me with a good visualization of what was happening with the model more than just looking at the RMSE.

```
RMSE: 74,857.73
```

Company Names	Cars Names	actual_price	pred_price	error
Porsche	Cayenne Turbo	130000.0	131176.167793	1176.167793
HYUNDAI	Sonata	25000.0	33881.078227	8881.078227
LAMBORGHINI	AVENTADOR SVJ	518000.0	276195.058312	-241804.941688
Peugeot	5008 GT	40000.0	32286.250267	-7713.749733
Porsche	Macan T	63000.0	48516.947805	-14483.052195
Volkswagen	Crafter	40000.0	17718.123800	-22281.876200
Tata Motors	Nexon EV Prime	22000.0	25286.696220	3286.696220
Mazda	Millenia S	30000.0	34806.834026	4806.834026
TOYOTA	VENZA	33400.0	38278.654438	4878.654438
BMW	118D	34000.0	29064.481842	-4935.518158

Experiment 1: Evaluation

Looking back at experiment 1, everything went well. The hardest part was actually cleaning the dataset as I had to work with a lot more columns than I am used to which all had different types of strings that I had to figure out how to clean correctly.

Experiment 2

For experiment 2, I decided to add a couple more columns from the dataset to see if I could make the model more accurate at guessing price. I decided on using torque and top speed. theoretically this should make the model twice as accurate but that is actually very unlikely. I still wish to see how that changes the RMSE. After making these changes, you can see that the RMSE improved by almost 20%. So adding two more columns was a positive move.

```
Experiment 2 RMSE: 60,998.57
```

Company Names	Cars Names	actual_price	pred_price	error
Porsche	Cayenne Turbo	130000.0	145823.883041	15823.883041
HYUNDAI	Sonata	25000.0	40882.241322	15882.241322
LAMBORGHINI	AVENTADOR SVJ	518000.0	377378.307489	-140621.692511
Peugeot	5008 GT	40000.0	33761.429456	-6238.570544
Porsche	Macan T	63000.0	50383.507853	-12616.492147
Volkswagen	Crafter	40000.0	18211.723641	-21788.276359
Tata Motors	Nexon EV Prime	22000.0	17805.482374	-4194.517626
Mazda	Millenia S	30000.0	37734.095603	7734.095603
TOYOTA	VENZA	33400.0	36111.290424	2711.290424
BMW	118D	34000.0	31818.308809	-2181.691191

Experiment 3

Experiment 3 was very interesting. I was expecting to just do something simple and try a new random 80/20 split of the dataset for training and see how similar the RMSE was. But if you notice, the RMSE is significantly worse. We're talking more than 10 times worse, which is odd because the actual vs predicted price tests appear to be pretty accurate and do not match the

extremely high RMSE. I believe this issue is because of extreme outliers like excessively expensive cars are being trained on. This shows that just using a random 80/20 split to train the model will not yield consistent results and that I may need to trim the fat by manually removing extremely high and low price cars from the training set to help average out accuracy.

```
Experiment 3 RMSE (new split): 913,026.95
Company Names              Cars Names  actual_price    pred_price         error
   Volkswagen          Tiguan Allspace       35000.0  35081.717416     81.717416
      Porsche               911 GT2 RS      293200.0 304847.946228  11647.946228
    Chevrolet                    Spark       14595.0  21933.379952   7338.379952
       Nissan                   Laurel       22000.0  18980.170919  -3019.829081
          BMW          118D SPORT LINE       36000.0  31435.482278  -4564.517722
        Mazda                    CX-80       45000.0  41349.403283  -3650.596717
        Mazda                    Etude       14000.0  21605.765777   7605.765777
 ASTON MARTIN         DBS SUPERLEGGERA      316000.0 313170.220967  -2829.779033
         Ford                 Explorer       35000.0  46769.391076  11769.391076
       Nissan X-Trail Platinum Edition       46000.0  32679.873541 -13320.126459
```

Impact

I believe the impacts of this project are positive because this model can be used to fairly accurately guess the price of a car based on the car's specs. This in turn could be used to compare prices with other cars and determine if you are getting a good deal on a car compared to others.

Conclusion

I learned a lot from this project. I learned how to set up, train and use a linear regression model. I learned how to modify and add to my existing model to make it more accurate. I also learned that I need to be more specific and careful to not overlook some things like how I missed something so simple as to exclude large outliers from the training set which ended up tanking my RMSE in my final experiment. Overall, I really liked this project. However, it was difficult and time consuming. I would like larger deadlines for projects of this scale and complexity. As someone who is new to python, it took me a lot of trial and error.

References

The dataset: https://www.kaggle.com/datasets/abdulmalik1518/cars-datasets-2025
Seaborn API: https://seaborn.pydata.org/api.html
Scikit linear regression:
https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html
Scikit train split:
https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html
Copilot for error checking and formatting. The biggest example would be Copilot bringing huge outliers in the training set to my attention and explaining how my RMSE tanked in experiment 3.

The Code

All of the project files including the code and csv can be found here:
https://github.com/GMNICKEL/Projects/tree/main/Python%20regression%20project%203