Introduction: **Aaron Squire CarMax Data Analytics Showcase**

Given the directions, it was clear the best approach would be to use a neural network to predict outcomes for each category given input (appraisal) data, as neural networks have a superb ability to find complex patterns in large amounts of data, much like the cardata.csv file given. The set-up for such is as follows:

Data purge: Appraisal data was taken as X data, and new car data was taken as y data to use for backpropagation. Categories “make,” “body,” “engine,” “mpg_city,” “mpg_highway,” “horsepower,” and “fuel_capacity” were removed, as each is represented with the model alone, so their inclusion is redundant. All data was then appropriately converted to be able to use as an input for the neural network. The columns which gave a range were averaged, and the columns with unrelated string data were converted to one-hot encoded columns, as there is no inherent relationship between the different values in these cells, so no other method would suffice. Lastly, the trim column was converted to integer boolean representations of the cell value (1 or 0). All rows containing a null value were erased, a decision that could impact the results, but visuals alone indicated no pattern among rows containing a null value. Below are the differing model architectures:

Category	Loss function	Metric	Final layer activation function	Optimizer
color and model	binary crossentropy	accuracy	softmax	adam
trim	binary crossentropy	accuracy	sigmoid	adam
mileage, model year, price	mse	mae	linear	adam

Specific number and size of dense and dropout layers differ for each category as accuracy can vary greatly.

Data: **(all tests were run with sample size n=1000)**

These values represent how much better the model is at predicting values than taking the top n most popular values throughout the whole dataset (as a **ratio**).

	Top 1	Top 3	Top 5	Top 10	Gr.
Color	1.19	1.023	0.988	n/a	D
Probabilities align nearly identically with color proportions for the full dataset, so no business operations can be improved using this model.					
Model	2.518	1.891	1.822	1.351	A
The model has a >1/3 chance to accurately predict a top 10 car choice for a customer, which is excellent given that there are 450 different models in the dataset. Using this model could allow more expensive options in the model’s top 10 list to be recommended to a customer with a realistic chance the car might be of interest to said customer.					

	Accuracy	Gr.
Trim	1.104	D
Predictions are slightly better than always guessing “not premium,” but since trim is an upcharge, the salesperson should always present the premium option regardless of what this model predicts.		

These values represent how often the model’s outputs are within n categories of the actual car bought (as a **percentage**, higher is better)

	Top cat.	±1 cat.	±2 cat.	Gr.
Mileage	.070	.233	.397	D
Model has >60% chance of being off by more than 12.5k mi., so no takeaways here.				
Model Yr.	.422	.724	n/a	B
>70% chance of guessing the model year within a range of 3 years is adequate. Results may be used to know when it’s appropriate to recommend a newer, more expensive model to a customer.				
Price	.280	.649	n/a	C
Model has appx. 2/3 chance of predicting the price someone is willing to spend within ±7.5k dollars. This could be used to determine when a customer is at a higher chance of potentially spending more.				

*Cat.=category (as determined by cardata.csv)
*Gr.=Grade (subjective, how well I believe the model fits each category)

Key Takeaways:

These models seem to be very good at predicting the model and model year of the car the customer will end up purchasing. Predictions for mileage and price may indicate some information about the type of car someone will purchase, but these predictions should be taken as a rough estimates. Color predictions and trim predictions are accurate but no better than simple pattern recognition from the full dataset.

Conclusion:

The accuracy of these models indicates this could be reliably used on real incoming customer data. To do that practically, CarMax could input the customer data into software with these models pre-loaded and compare the output of the models to cars in any given dealership’s lot, so that the cars closest to the output can be recommended to the customer. By using the cars suggested by these models as X data, and the cars the customers chose as y data, one could even run this new X and y through another neural network to increase accuracy even further and potentially increase recommendation accuracy and sales even more.