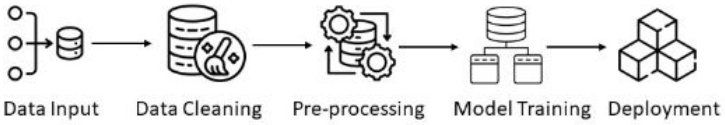
Hello Everyone

- Take you through our hackathon submisssion

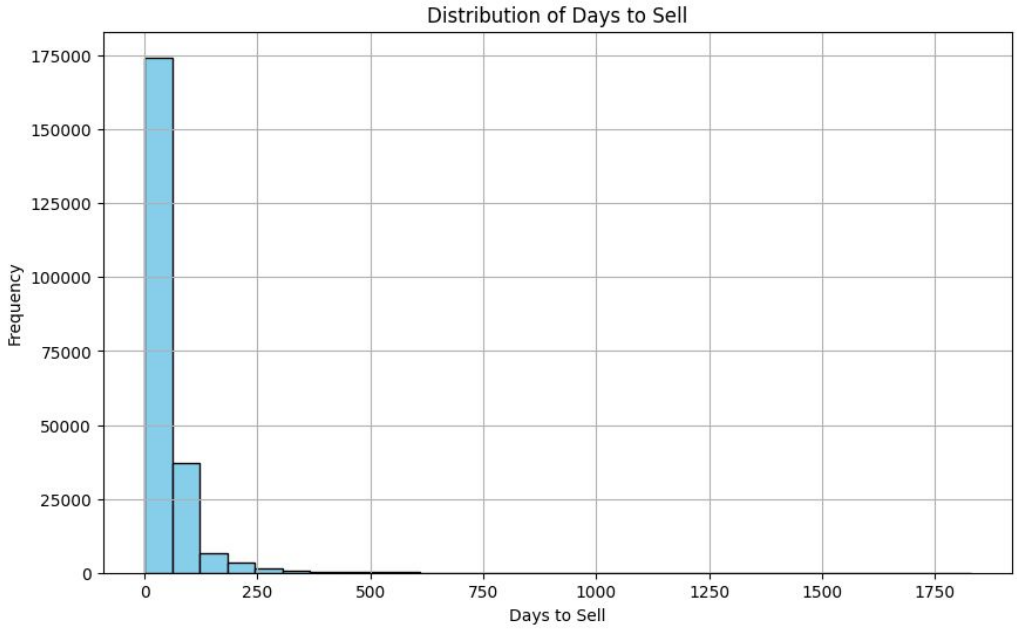


How did we Begin?

Recall our basic ML pipeline

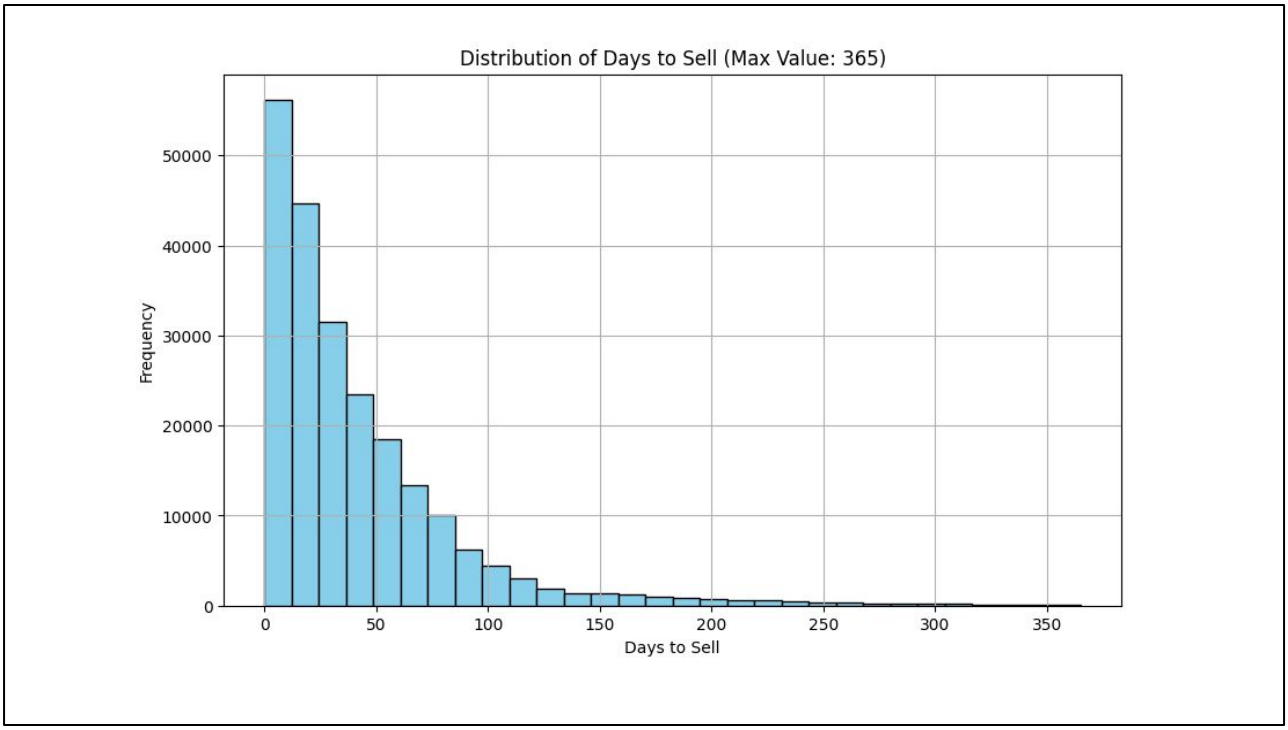
Defined a game plan

Got to tackling our problem



DATA EXPLORATION

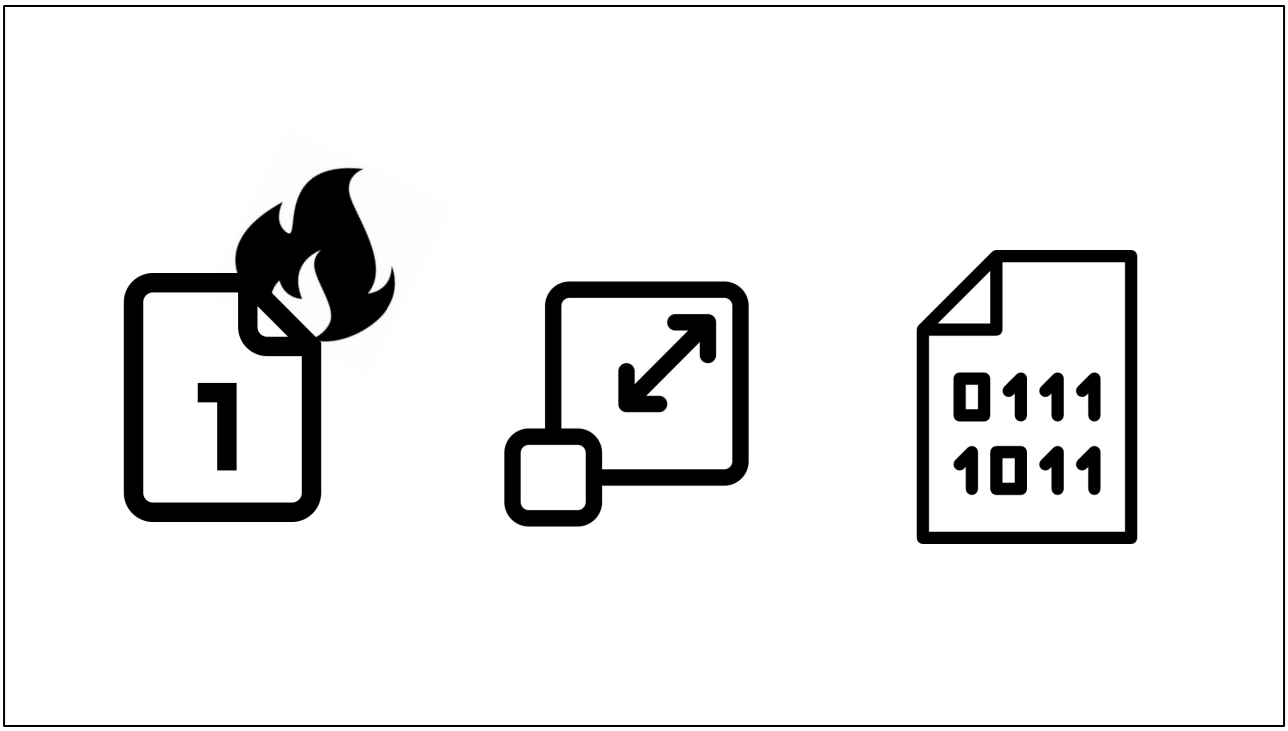
- Explored the distribution of days



- Refined the distribution by removing the outlier rows
- Considered our data distribution
- Quickly moved on to feature engineering

FEATURE ENGINEERING

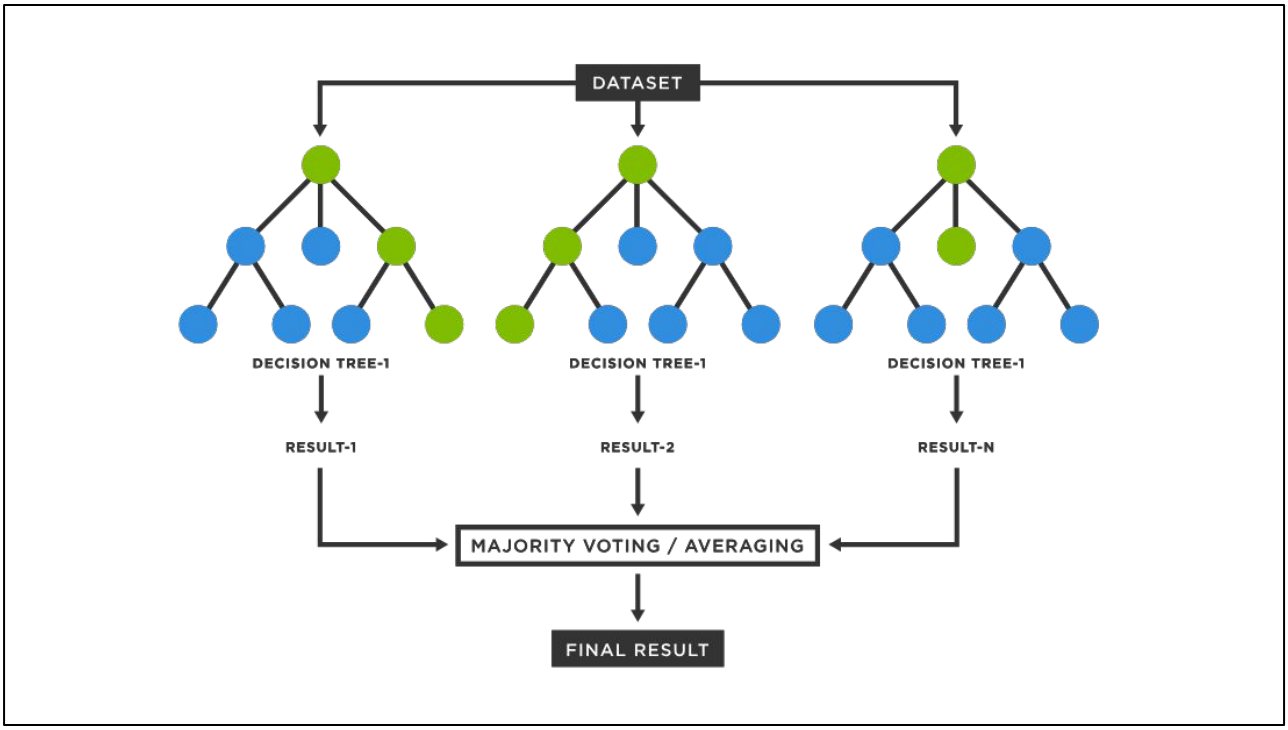
- Observed and Extracted what we THOUGHT were the most useful features
- Removed any values that were NaN / Missing
- Divided them into categorical, binary and numerical data.
- Converted first_date_listed to percentage through year -> reflect seasonality



TRANSFORMER PIPELINE

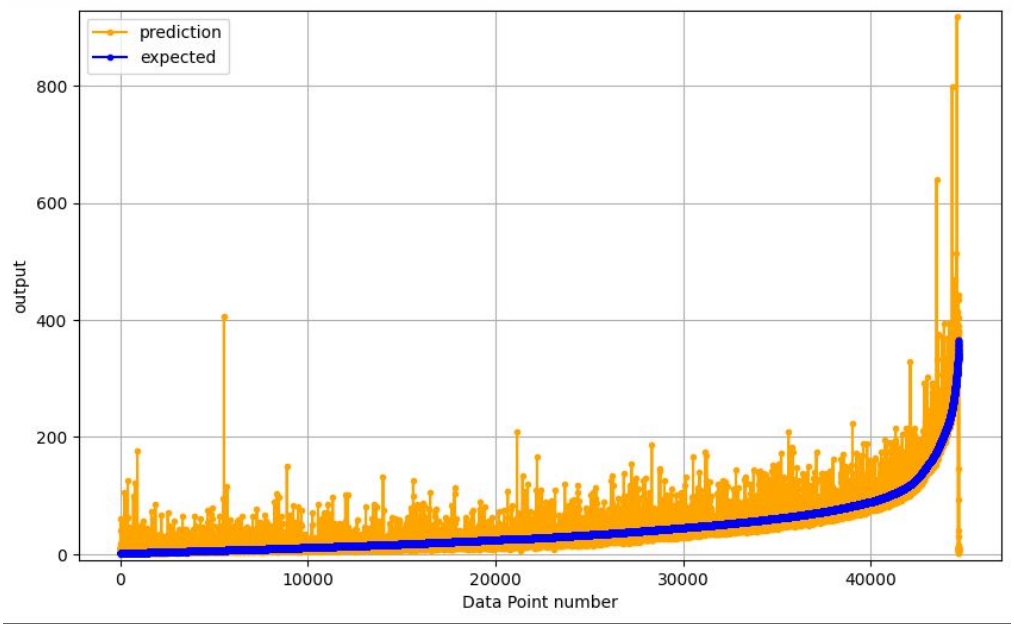
- One-hot encoded Categorical Features
- Scaled Numerical using a StandardScaler
- Encoded Binary Data

- Feature Vector of ~1200 Features Total (high dimensionality)



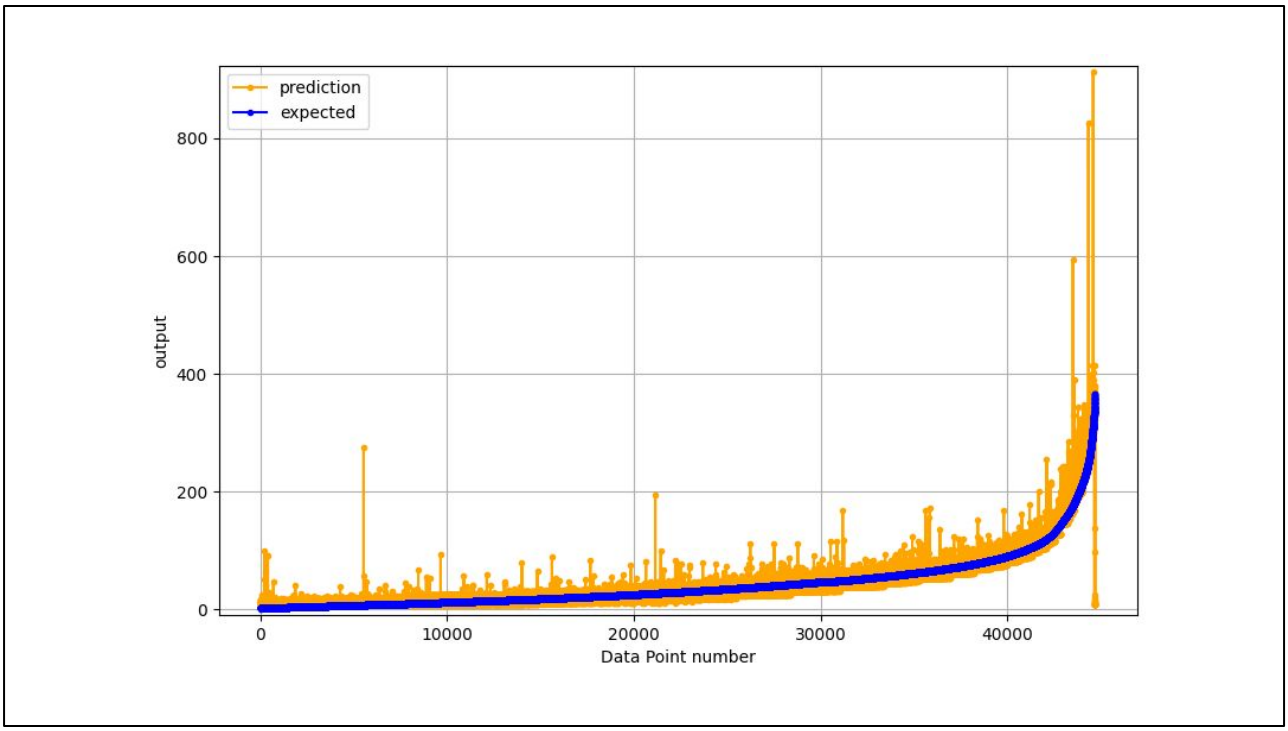
RANDOM FOREST REGRESSOR

- We want to predict as closely as possible the number of days
- Regression task suitable for this
- Random Forests Approach was a straightforward and interpretable model



RESULTS

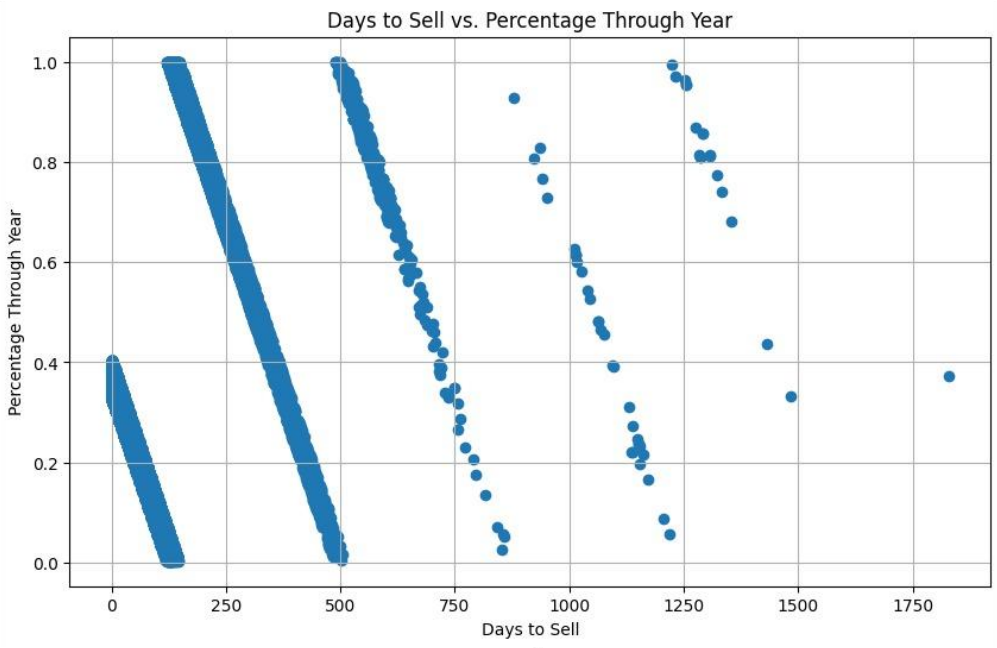
- Model was not underestimating
- Manageable overestimation



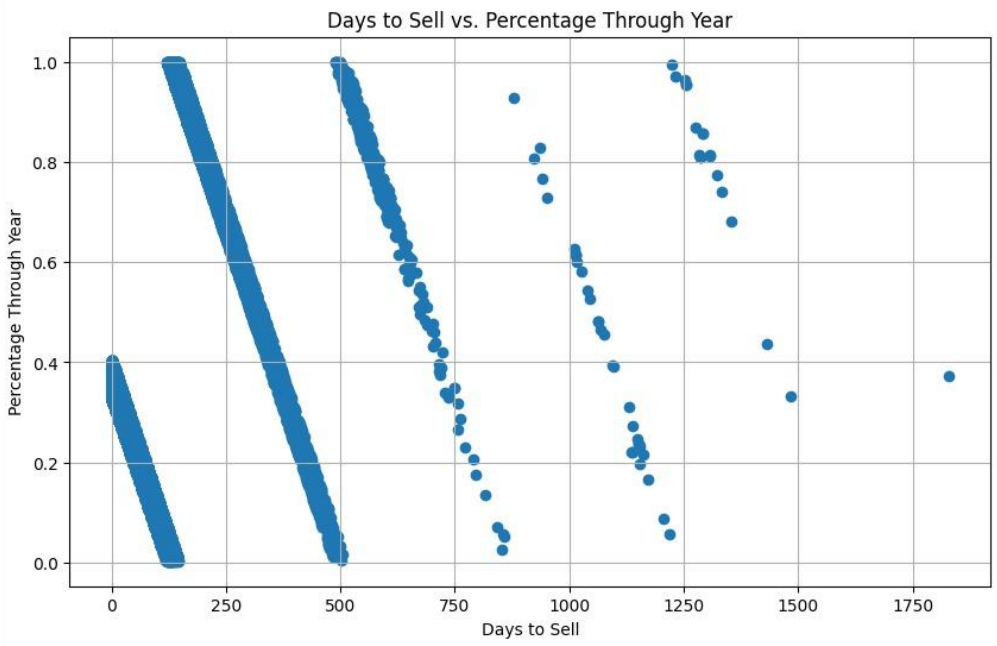
- hyperparameter tuning
- overnight Random Choice Cross Validation (10hrs long)
- Improved model

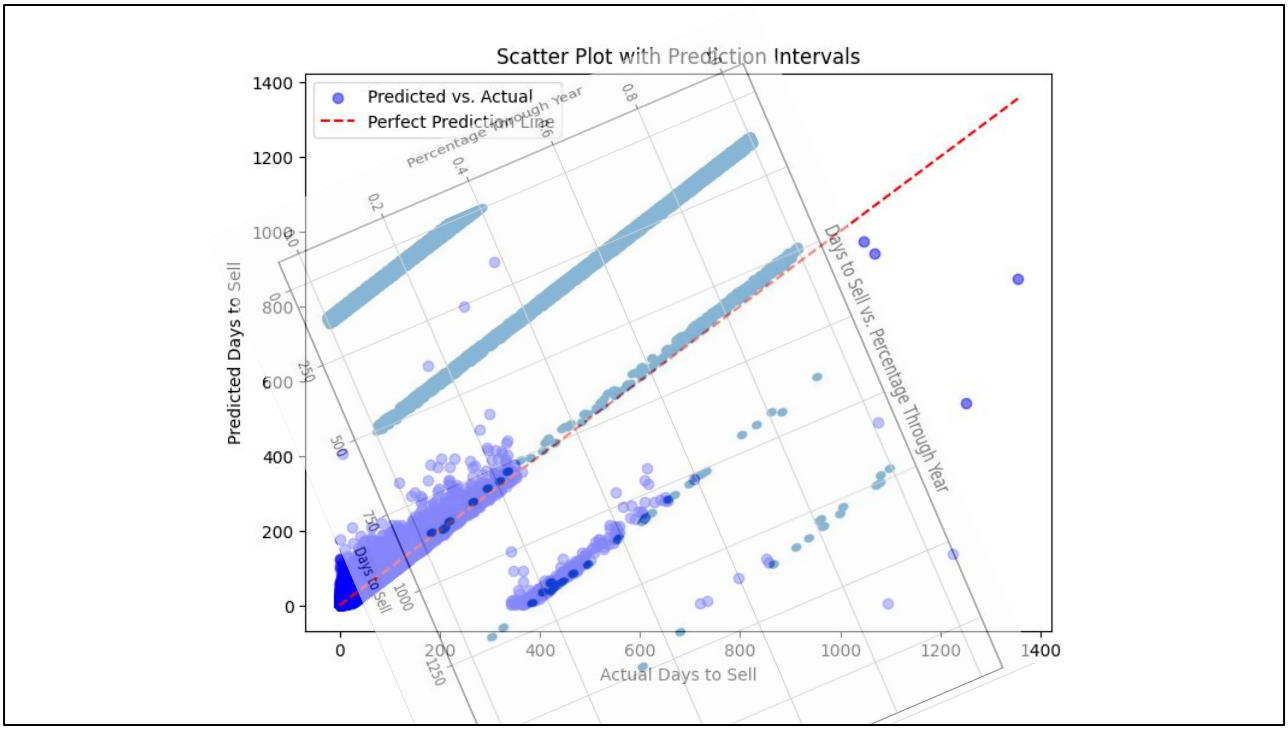
- MAE: 9.00 days
- RMSE: ~28.0 days
- R-Squared: 0.711

- Pretty great first attempt!
- Suspicious of high RMSE….



- Plotted our Predicted vs Actual Values
- Very Suspicious…. Seemed familiar from somewhere….
-

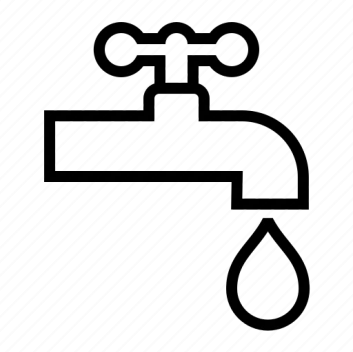

Days to Sell vs. Percentage Through Year
Percentage Through Year
Days to Sell
0.0
0.2
0.4
0.6
0.8
1.0
0
250
500
750
1000
1250
1500
1750

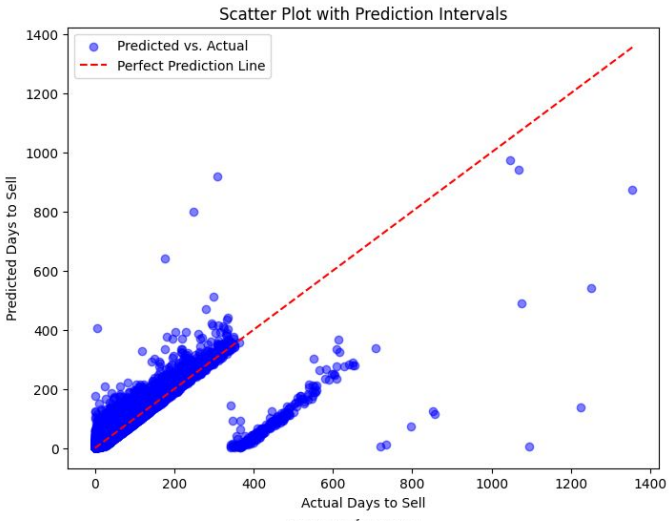

Scatter Plot with Prediction Intervals
Predicted vs. Actual
Perfect Prediction Line
Predicted Days to Sell
Actual Days to Sell
Days to Sell vs. Percentage Through Year
Percentage Through Year
Days to Sell
0
200
400
600
800
1000
1200
1400

Our model was Biased

Check presentation time here

Data Leakage





Explain the importance of data leakage

- When you accidentally give your model hints about our target feature - without realising
- We thought we had accounted for this by removing the year and calculating the listing date as a percentage of the way through the year
- All of our data were sales in May
- Model had learned the interval of how far through the year from may each sale was, and was following the same trend as our seasonality graph

- This is a MASSIVE issue
-
- Lots of features use the data from the LAST day of sale
- Careful that we did not leak this trend into our new features.

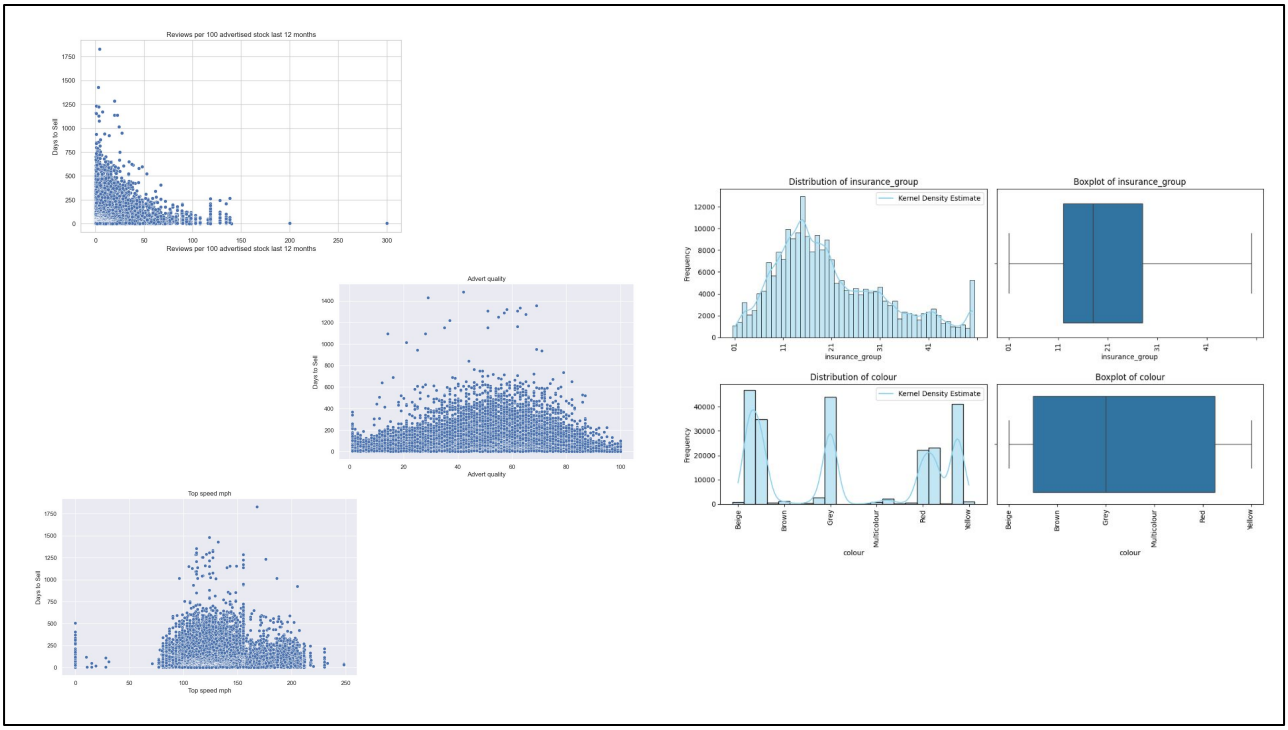
	stock_item_id	last_date_seen	first_date_seen	days_to_sell	first_retailer_asking_price	last_retail
0	52ae009b671ab58b3d4ff109a9fbdcf8d847de0fa190e1...	2023-05-05	2021-03-25	771	6995	
1	32b1bac6934b1f64ff43cffa9df5aa296ead8143c36f9f...	2023-05-09	2021-05-25	714	13725	
2	21703d22d87eaa95c4dc81a60ba2c8cbe3b90ab659292c...	2023-05-12	2021-11-26	532	15499	
3	661acafc271373946cea7d30ac7f34257404ab89a1ad33...	2023-05-16	2022-02-17	453	10995	
4	638216dc92410d965b416fea5b3cec9ca903368795fdde...	2023-05-04	2022-03-21	409	46000	
5	e3c8b08856a8736bb48c38f083d42f43f3e3e8e3466610...	2023-05-21	2022-04-27	389	1395	
6	82369d8013f2ab13f8f49fb780797298a8dd19974d3b60...	2023-05-14	2022-06-06	342	8257	
7	1fd13f137d7ed19e993b07dd1708992582537e56efb863...	2023-05-03	2022-06-16	321	23500	
8	c34a29671d55abf60ea1ab1c23ad21a0a7437c8ffea756...	2023-05-16	2022-06-23	327	96950	
9	db6f342f73f5c7819fef4254e6886387eac15e026878ab...	2023-05-22	2022-06-24	332	15995	

Observing the dataset as a whole

- What were the MOST useful features
- What were most prone to noise?
- What were most correlated with our target?
- How can we engineer our features to relate CLOSELY to AutoTrader's Goals/Customers?

To do this we carried out…

In-Depth Data Exploration

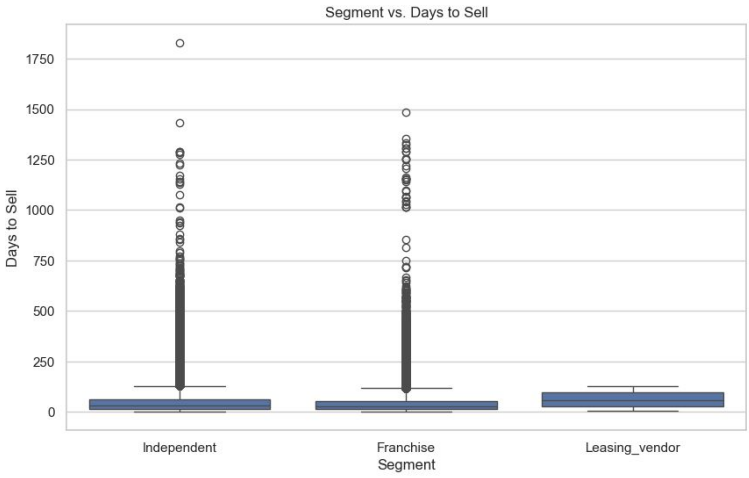


- Exploration Via Distribution and Correlation Plots
- Bar Charts, ScatterPlots, Histograms, BoxPlots

- Allowed us to identify noisy features
- Identifier outliers and trends in our data
- Overall usefulness of features

But there is a problem…

The Problem with Plots



Is the `segment` feature worth including, or just **noise**?

- Noise/Outliers make plots **ambiguous**
- Is there a significant difference between Independent and Franchise?
- How do we combat this?

Statistical Tests

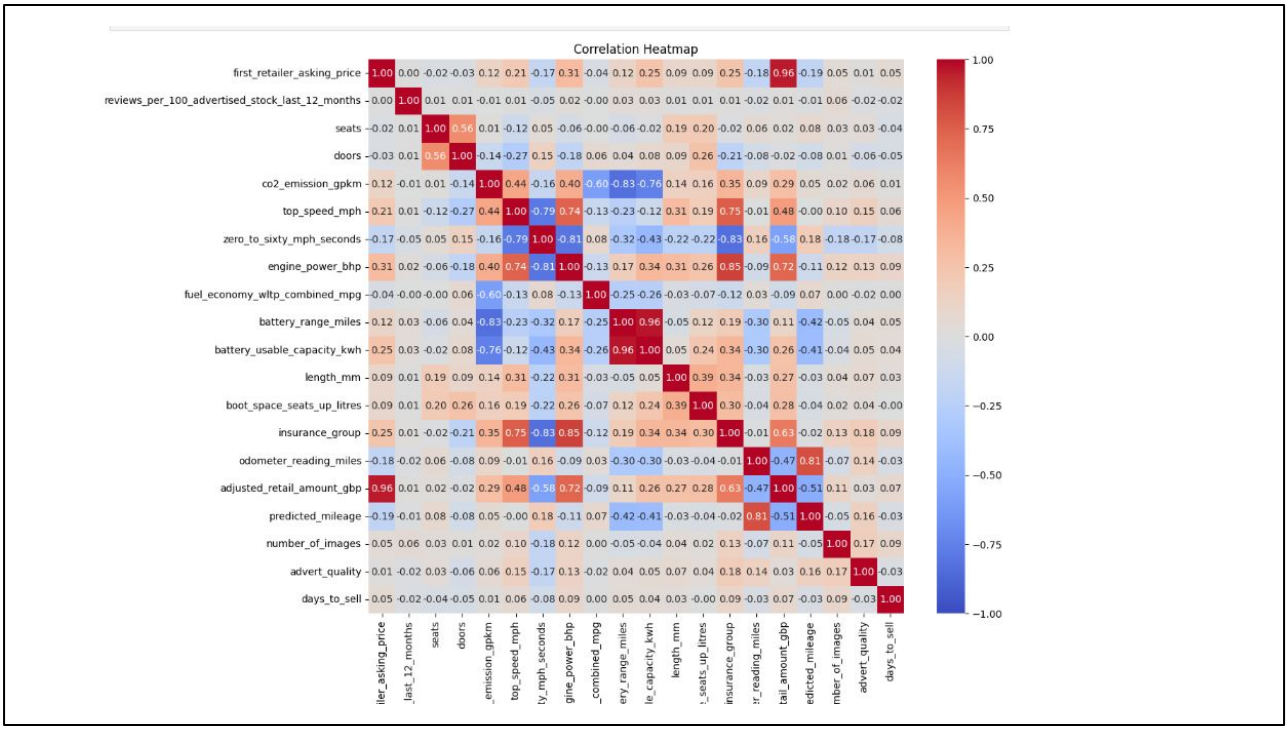
Mann-Whitney U Test (α=0.05)

```
Mann-Whitney U test statistic: 153450447.5
P-value: 0.0192223104085375
There is a statistically significant difference in days to sell between the groups.
```

So, **include** the `segment` feature

Mann Whitney U test - checks if **populations are equal**

- Like a t-test where you cannot assume that underlying distribution is normal
- Works with non-continuous data

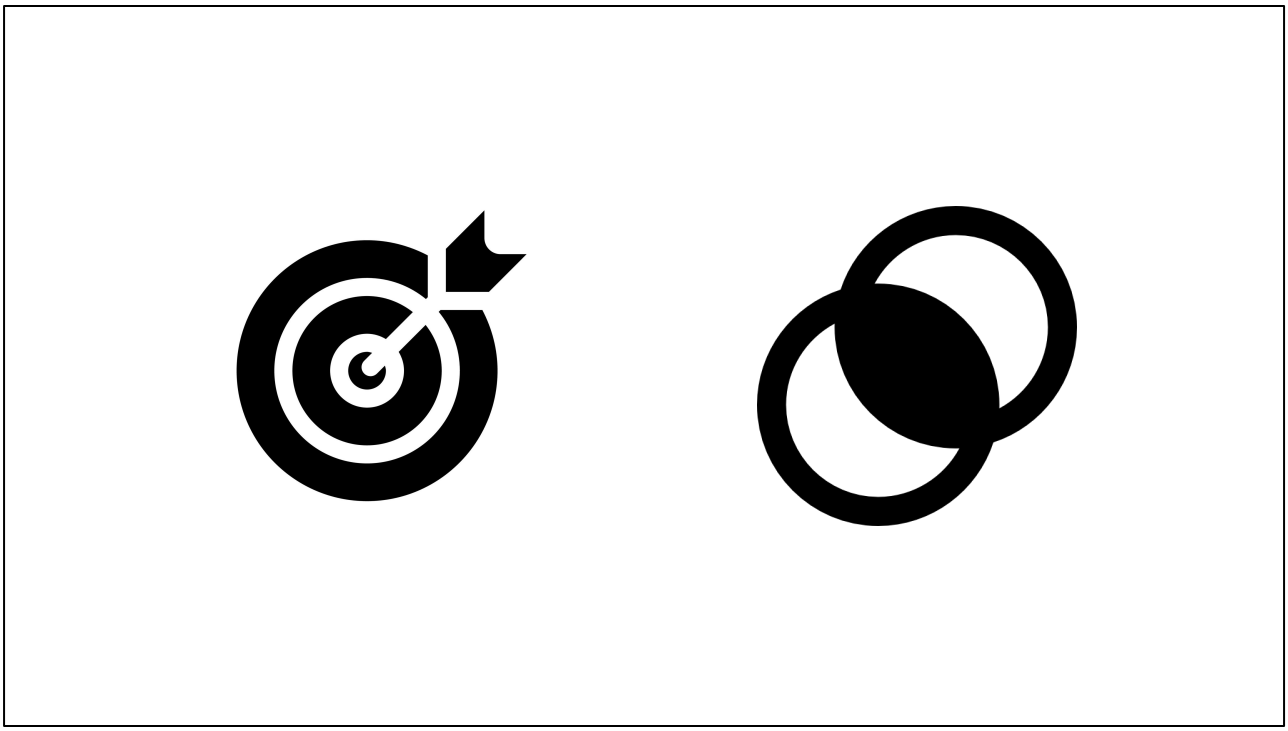


Correlation Heatmap of Numerical Features

- Allowed us to identify which features are correlated with each other
- **Informed feature engineering**

“The Curse of High Dimensionality”

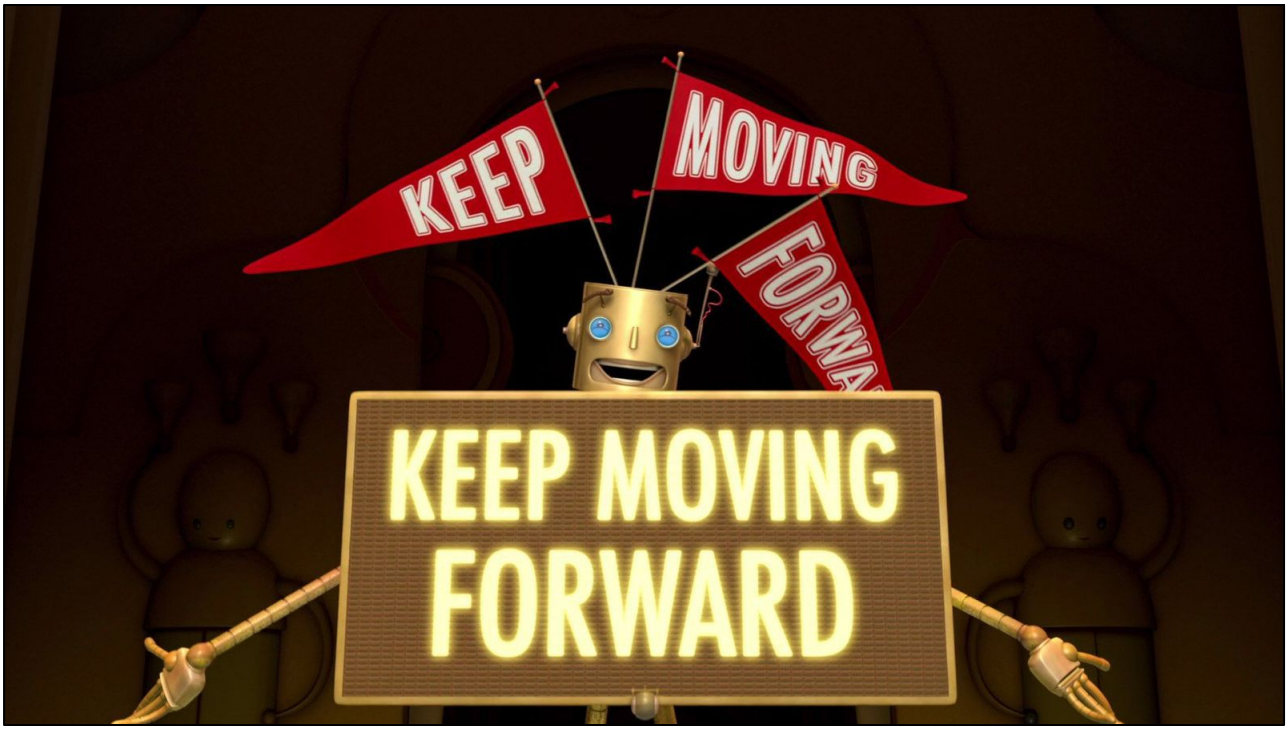
- **Increases computational resources** and runtimes
- Extreme GPU requirement for ANN models
- Makes it difficult for our models to pick our underlying trends



Combatted this by:

- Target Encoding -> reflects large categorical data as numeric

- Feature Combination
- Any features which represented the same data or was correlated -> combine into a single/reduced set of features.

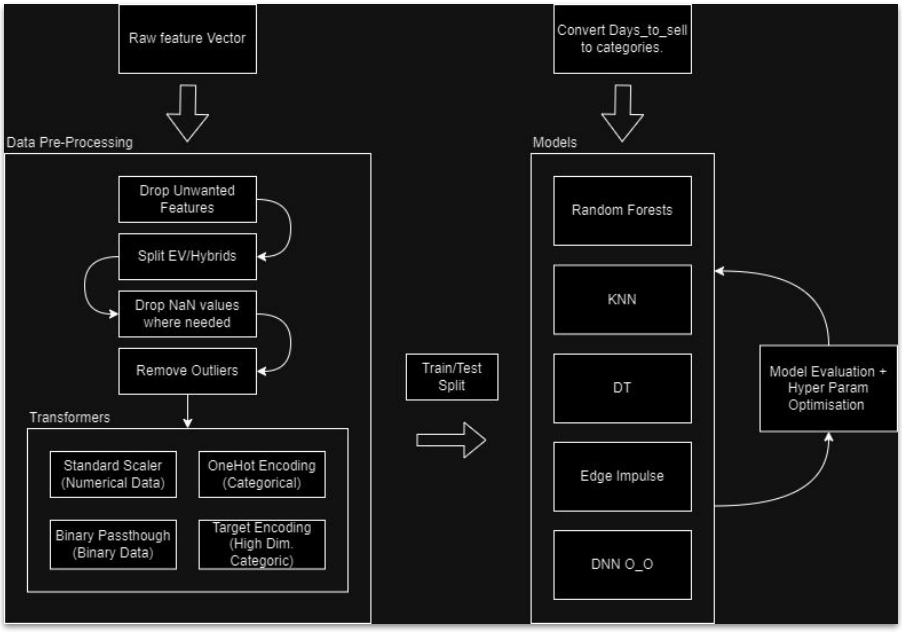


Given all the setbacks, we must keep moving forward

Sat down for some hours and rethought the whole approach using what we just spoke about

We thought about how to test everything that could be changed in our pipeline including data cleaning, feature engineering, feature encoding, model selection, and more

We came up with this workflow



Given this, we came up with a specific data pipeline we wanted to follow.
This was split up into:

- Data pre-processing and cleaning
- Transformers
- Various models to experiment with

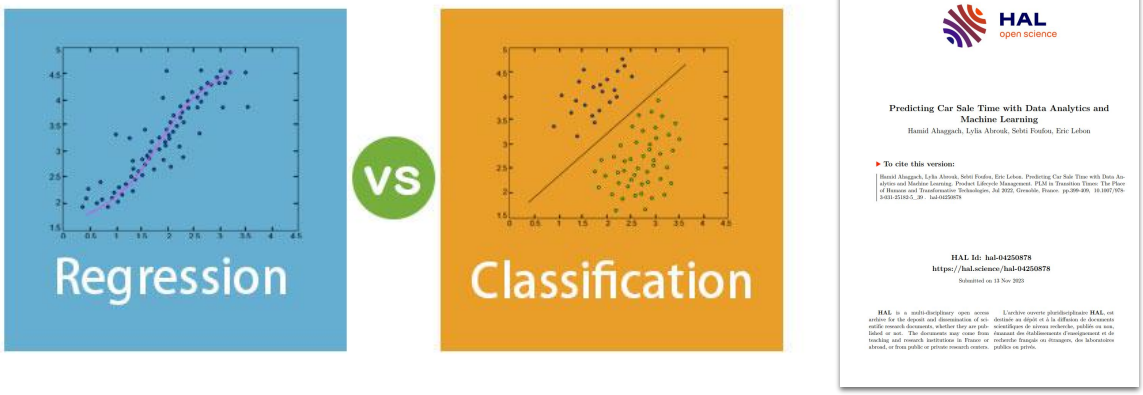
first_retailer_asking_price - seats	first_retailer_asking_price - vehicle_age	model_count - number_of_images	model_count - odometer_reading_miles	model_count - reg_year
6990.0	6976.0	1726.0	-63224.0	-228.0
13720.0	13721.0	7716.0	-8287.0	5712.0
15494.0	15494.0	784.0	-30287.0	-1212.0
10990.0	10987.0	2797.0	-76173.0	812.0
45995.0	45999.0	553.0	-9639.0	-1447.0
...	...	...	...	...
11040.0	11038.0	3820.0	-55171.0	1813.0
8995.0	8991.0	1154.0	-36544.0	-841.0
11295.0	11296.0	7727.0	-43757.0	5712.0
4695.0	4687.0	26.0	-38352.0	-1955.0
8696.0	8689.0	2307.0	-88513.0	327.0

Automated feature engineering

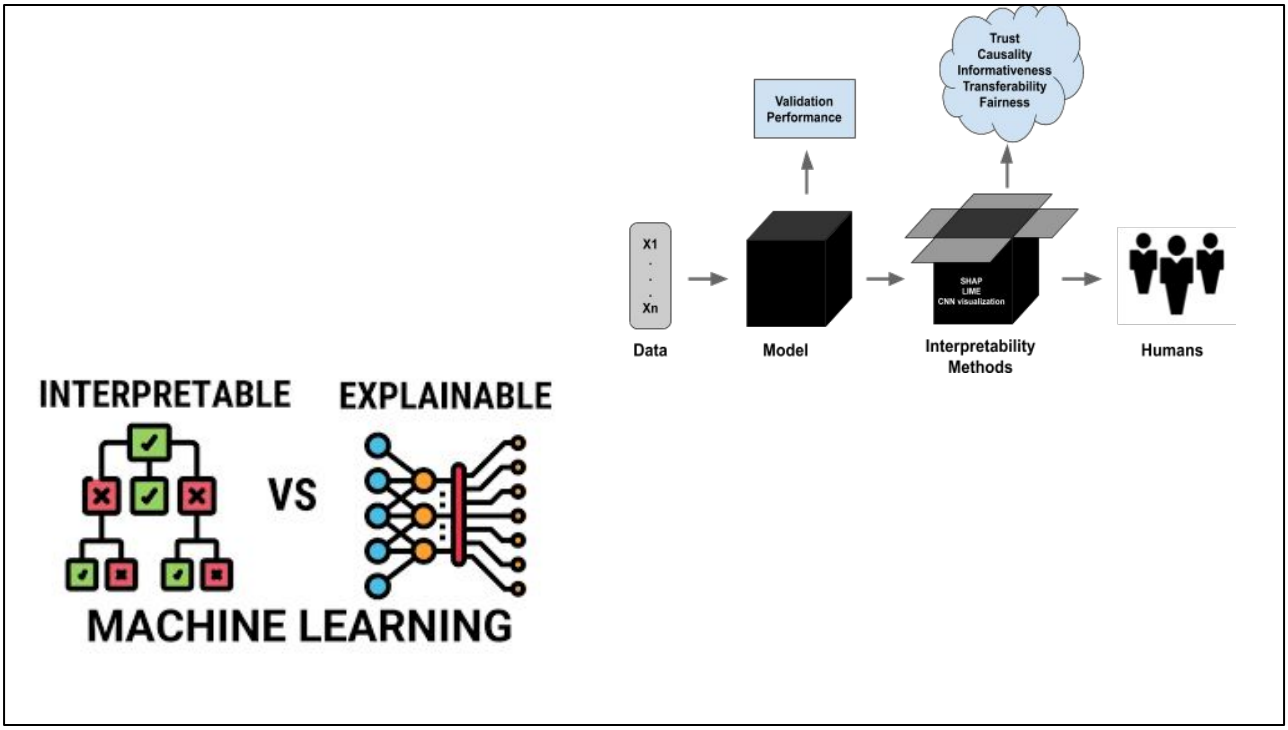
- We also experimented with featuretools to automate feature engineering.
- We had limited success with this, however could work with more training on models.

We also experimented with Edge Impulse, an automated machine learning platform.

- We created a neural network from our data.
- This completely failed, as the neural network always predicted the mean of the training data.



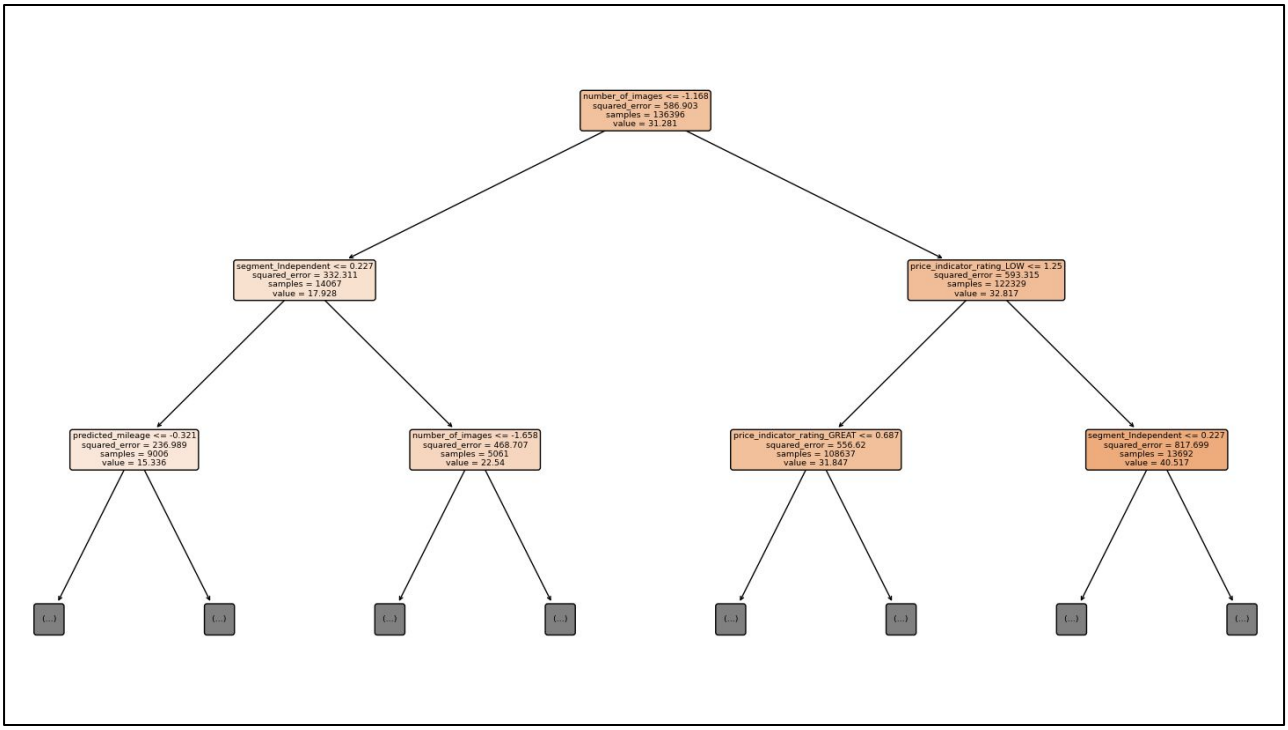
- Looked at literature
- Decided to look at classification as well as regression



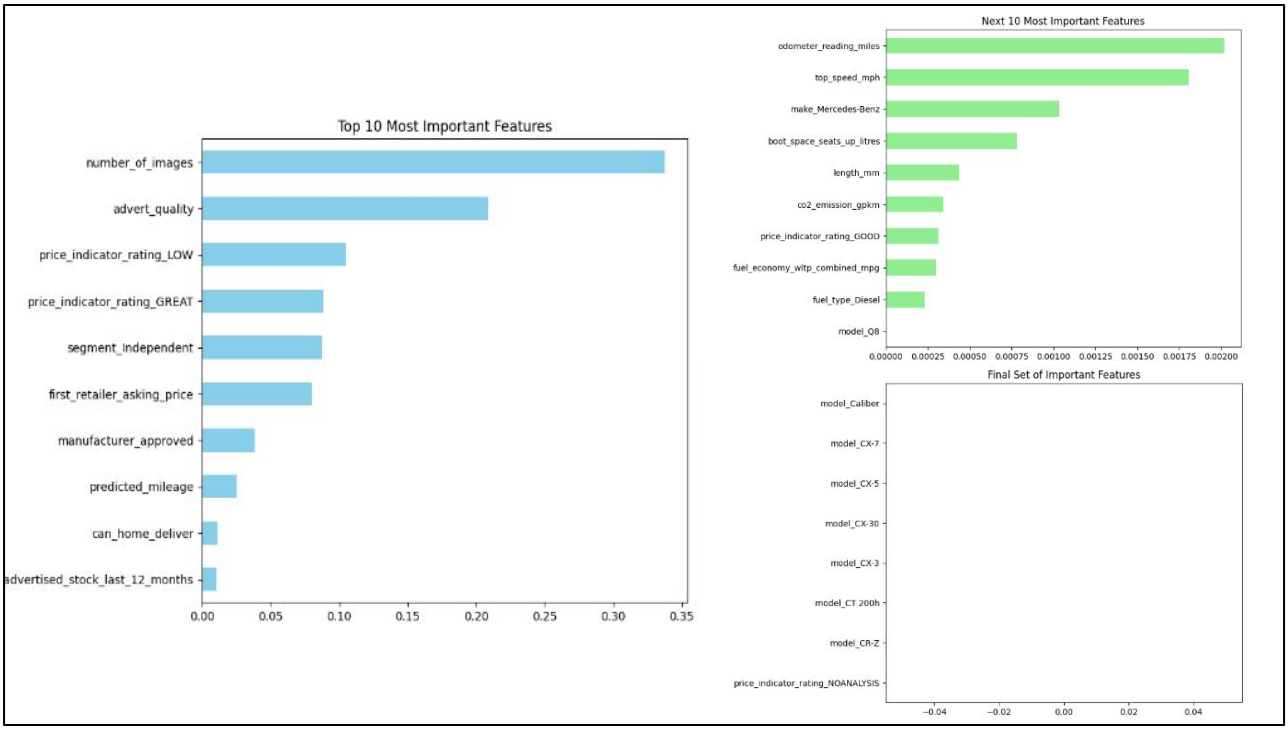
interpretable models can give us information about *feature importance* (more later)

Key takeaway: more useful for **customer**

- Customer lists car, sees 3 months
- Customer sees “add more photos” for lower estimation
- => quicker for customer, more throughput for autotrader
- Human features ended up being more important



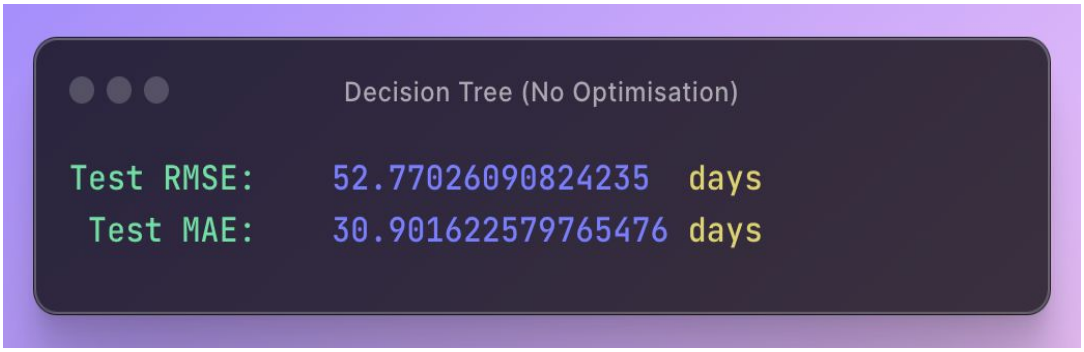
- This is the actual (regression) decision tree
- **Used to experiment** with hyperparameter optimisation, see what worked
- **Laid groundwork** for RF model



- Extremely important aspect of decision tree is that it is **interpretable** - it is possible to see how much the model values each feature
- Plotted the feature importance of the model that gave an RMSE of `22.9` days
- Showed that **human features** were the most important for predicting how quickly a car would sell (`number_of_images`, `advert_quality`, etc) vs what the car actually was

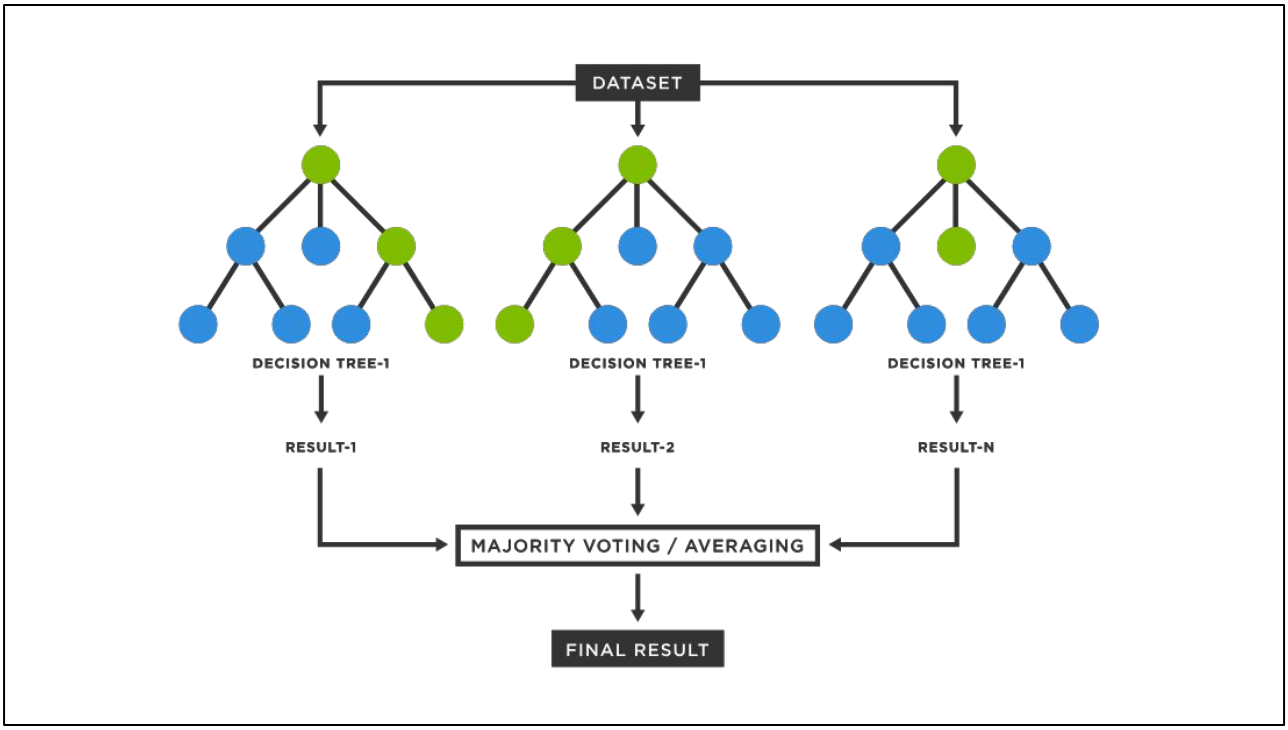
- Similar conclusions as first presentation, but these extracted by the model

Now will breeze through some results

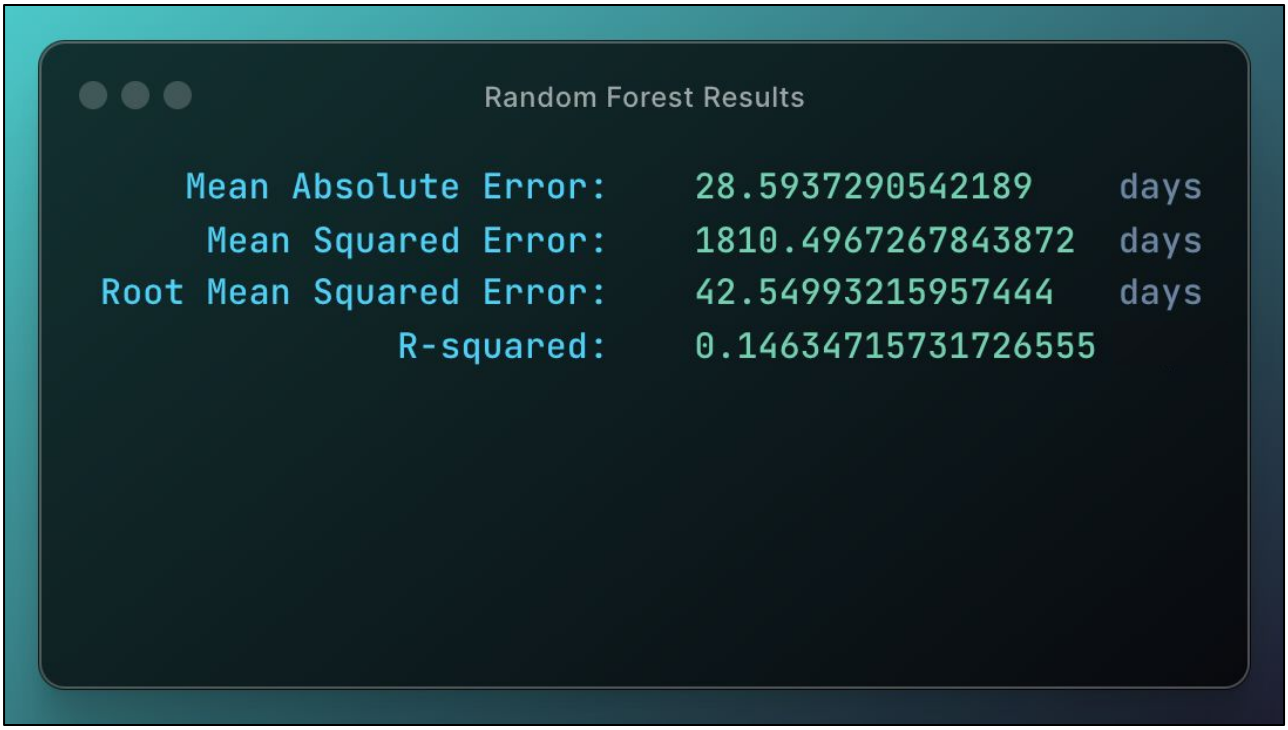


Before optimisation

After hyperparam. optimisation
Showing an decrease of `29.9` (3sf) days in RMSE
Ready to apply process to Random Forest model

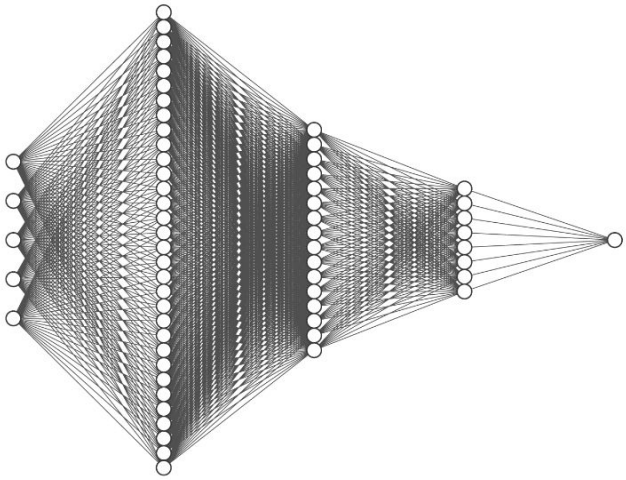


Apply our findings **to a Random Forest** model.
Literature suggests that RF is more able to pick out trends and less likely to overfit vs decision tree
We expected higher performance - not sure why



This significantly underperformed the Decision Tree model,
Which was not what we expected.
We need to investigate why

NN Approach

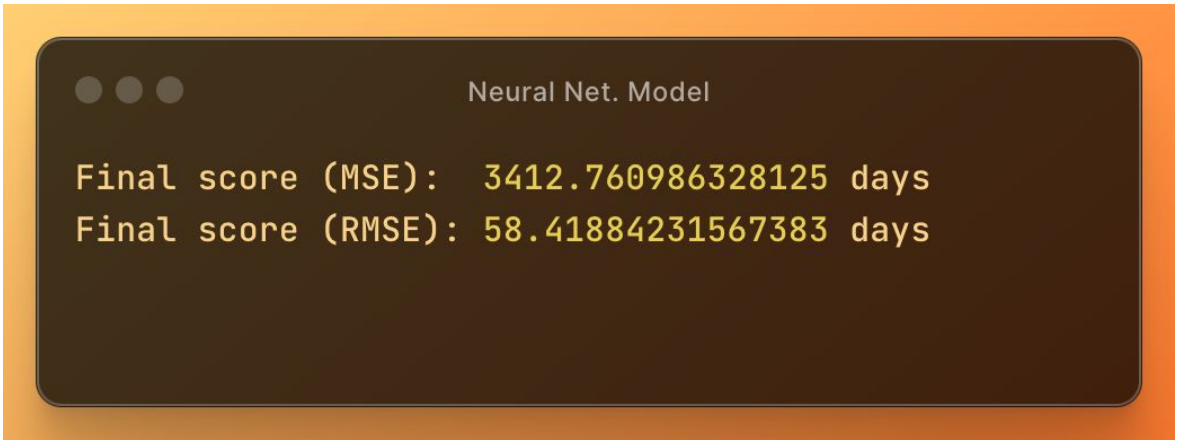


- Tried a Large variety of structures (in combination with large variety of feature combinations)
- Generally better results came from a pyramid-based structure

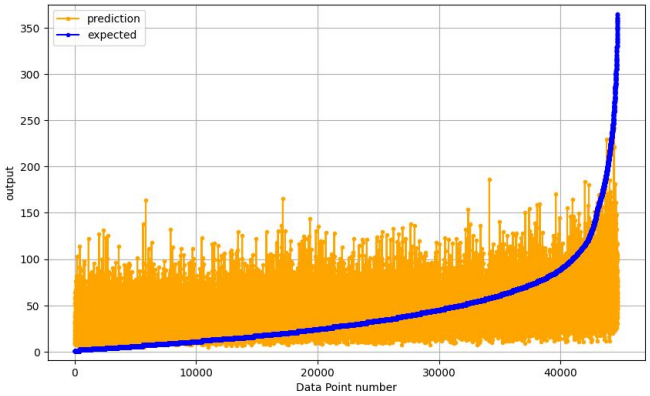
- The Dimension Reduction Improved the models.

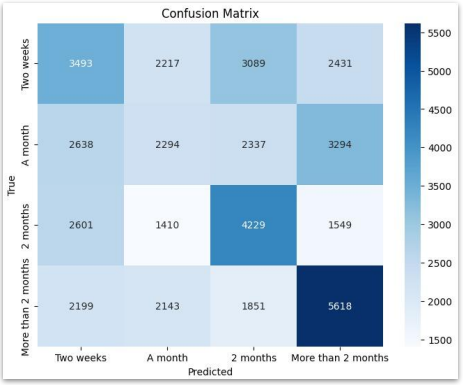
- Tried both classification AND regression
- ADAM optimiser (learning rates between 1x10-3 -> 1x10-7)
- Loss functions: EntropyLoss (classification) MSE loss (regression
- Softmax for multiclass, sigmoid for binary classification
- Variety of layers (started on a single layer -> went all the way to 10 layers) no considerable differences, but generally better with more layers.
- help via our ML lecturer

- Resulted in NO useful results compared to Random Forests
- Training Resources became too expensive



- We also attempted a Neural Network model, despite not being interpretable
- We found that the neural network performed worse than the Random Forest model (by RMSE)
- The plot showing distribution of guesses does not look very different either.





Observed that both types of problem across all models were poor:

- Regression + classification wasnt useful
- Models were either struggling to extract any distinct trends
- Similar results across all models + combinations of engineered features
-

So *what* limited our solution?

“ Simply knowing what a jupyter notebook is

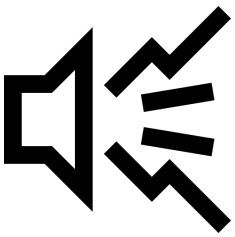
DOES NOT

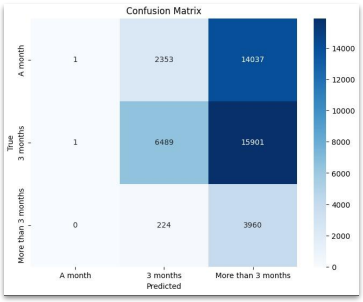
make you an ML engineer! ”

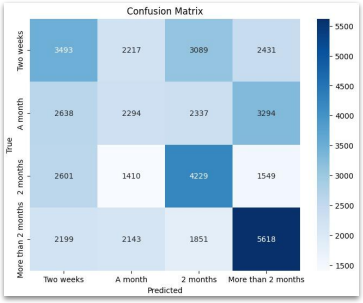
~ Tom Cassar, 2024

On a serious note

- We are halfway through our ML module
- The span of ML is MASSIVE and our problem is a hard one!





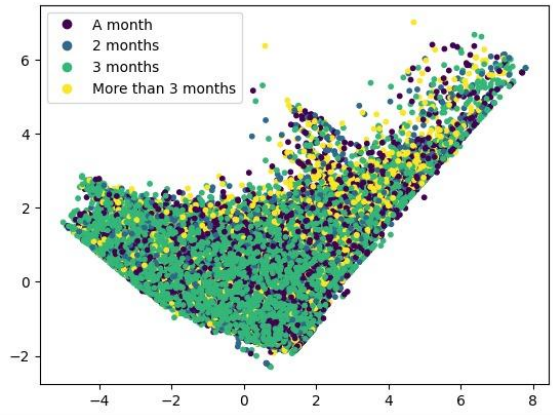


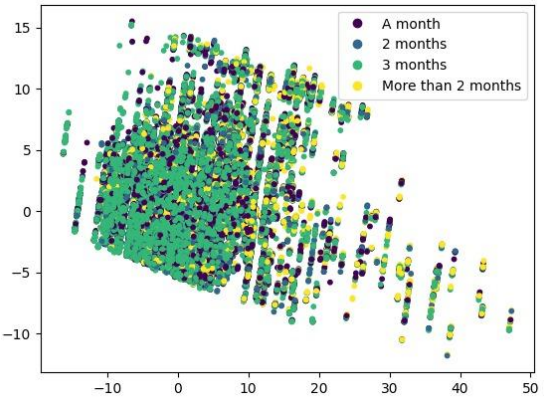
Lack of variety in seasonal data

Imbalance of data

Noise of data

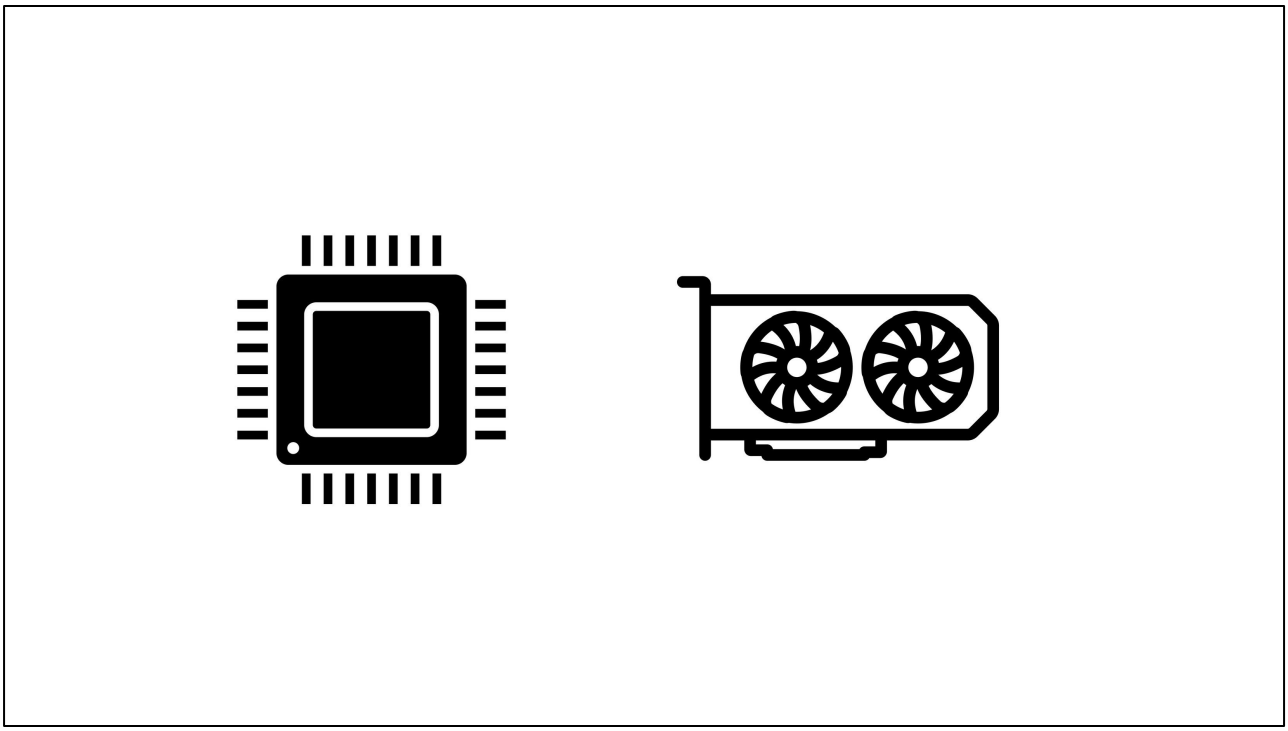
- majority of the features were noises
- No effects from custom class weights
-





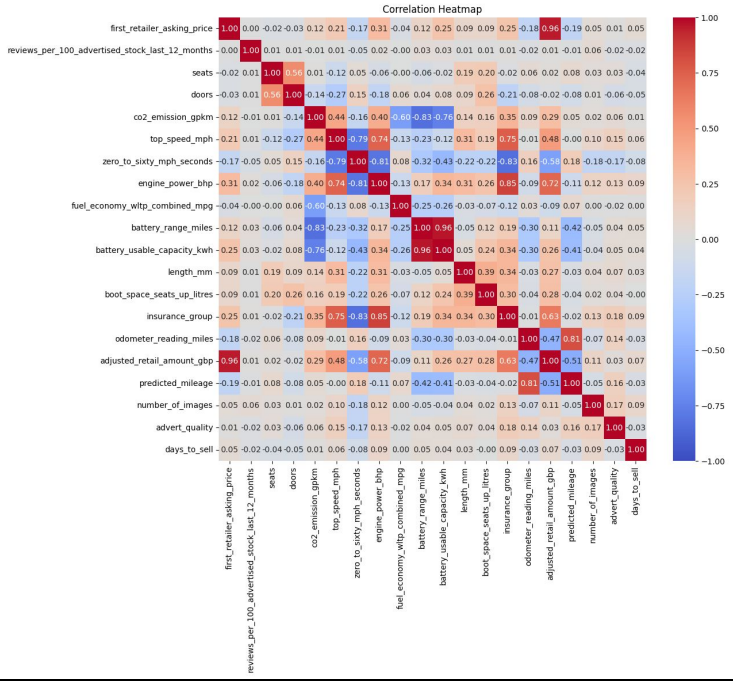
PCA Analysis

- NO separation in useful bins of data (minimum being 1- 2-3 4 months)
- No significant effect by reducing classifier amounts
- We suspect this again is due to the previously mentioned limitations



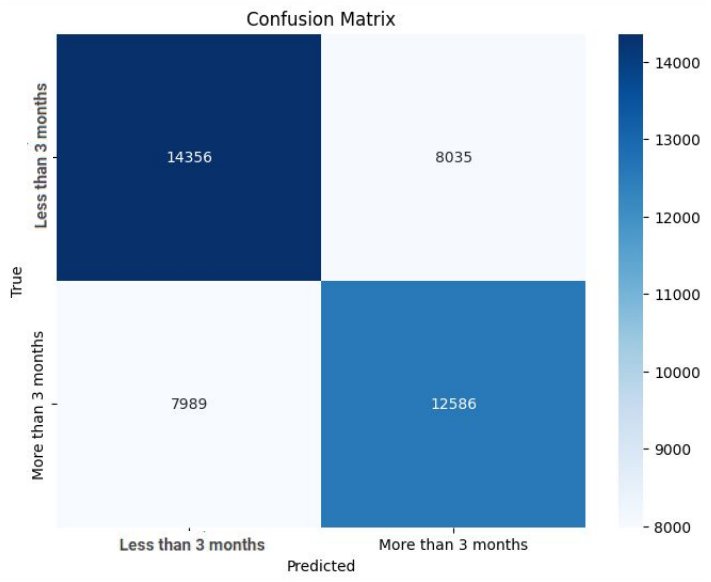
Computation and Resources

- As students we cant afford a GPU rig, or use the universities GPU array
- Time is an expensive resource
- large dimensionality RF took over 12hrs for CV



Correlation between data

- Results from correlation heatmap show interesting results
- There were several features very closely correlated with each other
- Most relevant features for days_to_sell include number_of_images and engine_power



Random Forest:

n_estimators: 1200

min_samples_leaf: 2

Min_samples_split: 2

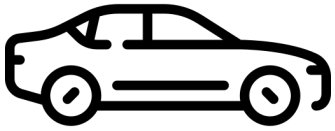
max_depth: 100

bootstrap: true

Class Weights:
< 3 months: 0.1
> 3 months 0.1029

- This random forest classifier was our best model, classifying between more or less of three months.
- This resulted in 60% accuracy
- Show there is in fact a correlation, even though weak with little feature engineering.
- Requires further analysis and feature engineering to extract correlation for model training.

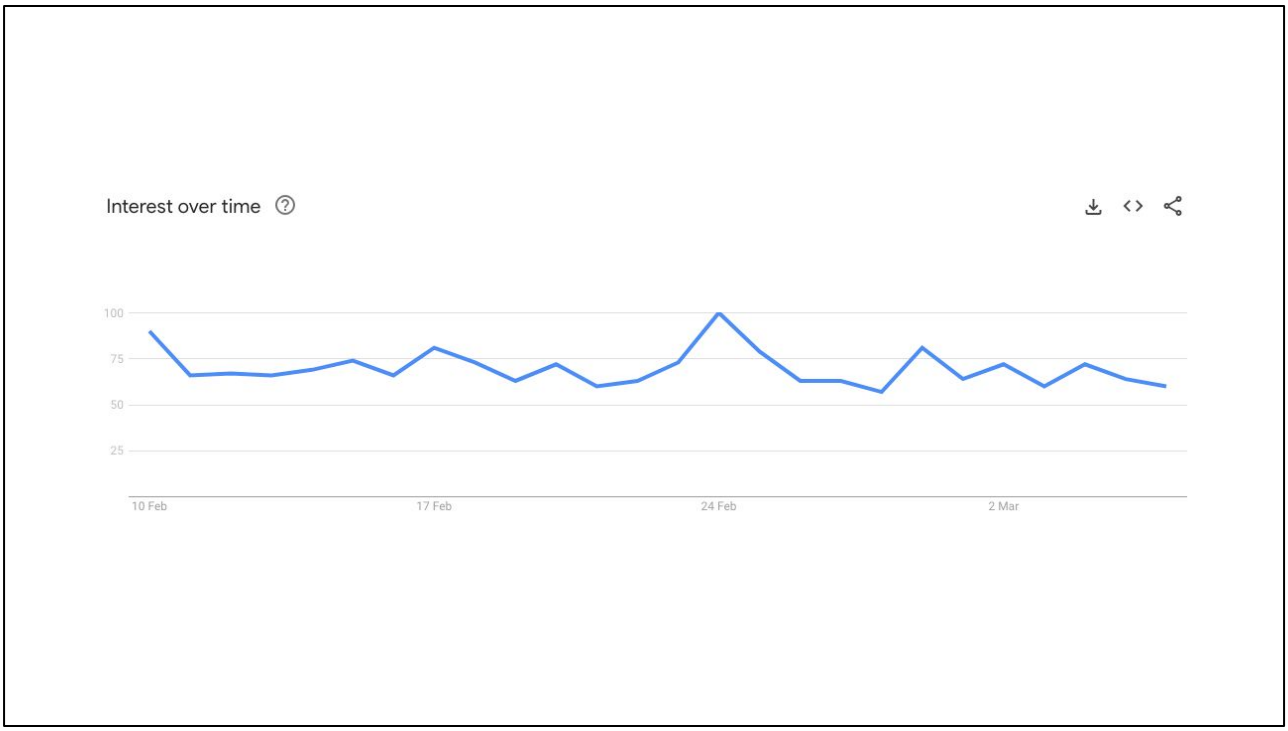
VS



Physiological effects vs product quality

- From feature analysis, most important features were those related to the advert itself
- Not the actual car or specifications of what was being sold.
- We can then conclude that these are the most crucial for advert performance.

Improvements
&
Takeaways



Improvements / Takeaways

- Could integrate with external data / business knowledge.
- Such as google trends of interest in specific cars or UK economy market performance.
- Add extra relevant data points related to advert such as response rate of seller, add question asking if sold

MathSoc AutoTrader Hackathon 2024

Ioan Gwenter, Lourenço Silva, Tom Cassar