Analysing Second Hand Car Sales Data with Supervised and Unsupervised Learning Models

1 Introduction

This report presents analysis of the second-hand car market using supervised and unsupervised learning for car price prediction and pattern identification. The surge in demand for used cars in the UK, driven by factors like affordability and sustainability, has led to fluctuating prices in an unstable economy (Car Dealer Magazine, 2023). Consequently, buyers seek realistic pricing information, and sellers aim to optimise their deals. Factors contributing to car price include make, model, and condition. Determining the relative importance of these features is challenging (Bukvic et al., 2022).

Existing ML literature for used car price prediction include linear regression (Alhakamy et al., 2023), SVR, and DT (Bukvic et al., 2022), RF (Chen et al., 2022), and ANN, SVR, and RF combined (Gegic et al., 2019). High accuracy has been demonstrated using BP neural networks (Sun et al., 2017; Liu et al., 2022).

A comprehensive comparison of ML models to aid decision-making in the used car market is provided.

2 Methods

The code and environment versions are provided in the Jupyter notebook (Car sales analysis_AI_SarahRogers.ipynb).

The car_sales_data.csv, includes 50000 car sales with numerical features engine size, year of manufacture, mileage and price and categorical features manufacturer, model and fuel type (appendix 6.1, figure 1).

2.1 Supervised Models

2.1.1 Model selection

2.1.1.1 Regression

Regression models for price prediction are linear, polynomial, multiple linear, DT, RF, and SVR (Scikit learn, n. d.-w; W3 Schools, n. d.-b).

Single feature price prediction was performed by linear, polynomial, DT and RF regression. Combined numerical features were analysed using multiple linear, DT, RF, and SVR (Scikit learn, n. d.-v; n. d.-c; n. d.-h).

2.1.1.1.1 Categorical and numerical features

RF was explored with all features for price prediction. Categorical values were converted using label encoding (Scikit learn, n. d.-s).

2.1.1.2 ANN

The optimised model has a typical architecture detailed in appendix 6.2, table 1 (Schröder et al., 2018; Shen et al., 2021). The model comprises three hidden layers, with 64 units, ReLU activation, dropout (0.1) and L2 regularisation (rate 0.01). The output layer has a linear regression function for single continuous output (Montesinos López et al., 2022). The model was trained for 200 epochs, with validation split of 0.1, Adam optimizer (0.001) and mean squared error loss (for monitoring model performance) including early stopping (patience 15) for improved generalisation.

ReLU activation is widely used in hidden layers introducing non-linearity and training acceleration (TensorFlow, n. d.). Dropout, L1, L2 and elastic regularisation prevent overfitting, improving generalisation though neuron randomisation and weight penalties (Geeks for Geeks, n. d.-a; n. d.-c; Scikit learn, n. d.-i; n. d.-k; n. d.-j). Early stopping prevents overfitting by halting training before degradation (Keras, n. d.-d).

2.1.2 Optimisation

2.1.2.1 Regression

Cross-validation, resampling applied to the train-test split, was used for polynomial degree optimisation, providing the best balance between fit and generalisation, with minimal complexity (HowStuffWorks, 2023; Scikit learn, n. d.-b).

DT and RF hyperparameters were tuned using randomised search (Scikit learn, n. d.-r). Optimal hyperparameters are detailed in appendix 6.2, table 2. These models can capture complex relationships but can overfit, especially with deep trees (Codecademy, n. d.-b). RF is more robust with bootstrap sampling (Geeks for Geeks, n. d.-b).

For SVR, randomised search optimised C (regularisation parameter) and epsilon (margin of tolerance) as 10 and 1, respectively. Larger C values are for complex datasets with less noise, where the model is required to fit the training data more closely. Epsilon controls the width of the epsilon-tube around the predicted value, determining which data points are included in the margin. Kernel choice was 'poly'. (Scikit learn, n. d.-x; n. d.-v).

2.1.2.2 ANN

Optimisation included the number of hidden layers and hyperparameter tuning to ensure robust learning and generalisation for optimiser, learning and drop-out rate, regularisation, activation, and early stopping (described in appendix 6.2, table 3) (Montesinos López et al., 2022; Keras, n. d.-c; n. d.-e; n. d.-b; n. d.-a).

Overfitting was possible in all cases where early stopping was not triggered. As training loss decreased, Validation did not increase at later epochs or demonstrate reduced loss overall (AWS, 2023).

2.1.3 Evaluation metrics

Data for regression analysis was reshaped with StandardScaler so that the intercept corresponds to the dependent variable and the mean of the independent variable (Scikit learn, n. d.-u).

Evaluation metrics for supervised models are MAE, MSE, RMSE and R^2 (Scikit learn, n. d.-n; n. d.-m; n. d.-o). Lower MAE, MSE and RMSE, indicates increased accuracy, whilst an R^2 of 1 indicates perfect fit.

2.2 Unsupervised Models

2.2.1 Model selection

k-Means, AHC, MS and DBSCAN models were explored (Scikit learn, n. d.-a; n. d.-f; n. d.-d; n. d.-e; n. d.-g).

Six pairs of features were inputs for k-Means (appendix 6.2, table 5). The best clustering was for year of manufacture and price, chosen as the input pair for AHC, MS and DBSCAN.

2.2.2 Optimisation

For k-Means, optimal k was determined by calculating the inertia for k, the difference between consecutive inertia values, the relative differences in inertia, and then the elbow point for optimal k (Codecademy, n. d.-a; Scikit learn, n. d.-f).

For AHC, k was selected using optimal Silhouette and DB score (Scikit learn, n. d.-d). Single, average and complete linkage was assessed along with an alternative scaler, MinMax (Scikit learn, n. d.-t).

Bandwidth range 0.1 to 20 was supplied to the MS algorithm (Scikit learn, n. d.-g).

DBSCAN hyperparameters were optimised with ParameterGrid for five values of epsilon (size of the neighborhood around each data point) and three minimum samples (minimum number of data points required to form a cluster) (Scikit learn, n. d.-q; n. d.-e). Chosen hyperparameters gave the optimal Silhouette and DB score.

2.2.3 Evaluation metrics

Internal evaluation metrics, Silhouette score and DB score, were calculated without "ground truth" labels (Hyperskill, n. d.).

Low DB score indicates good separation and compactness of clusters (Scikit learn, n. d.-l). High Silhouette score indicates good clustering performance (Scikit learn, n. d.-p).

3 Results and Discussion

3.1 Data exploration

Car price ranges up to £170000, with higher numbers of cars at lower price points (appendix 6.1, figure 2).

3.1.1 Numerical features

Year of manufacture and price show the biggest correlation, followed by mileage and engine size (appendix 6.1, figure 3) (W3 Schools, n. d.-a).

The strongest correlation between all numerical values is year of manufacture and mileage.

3.1.2 Categorical features

Median price and range is similar across fuel types. Price variability is greater for petrol cars with higher priced outliers (appendix 6.1, figure 4).

Car manufacturers BMW and Porsche have higher median prices, wider range, and higher value outliers (appendix 6.1, figure 5).

3.2 Supervised regression models

3.2.1 Regression

Linear regression was performed to predict price with numerical variables (appendix 6.1, figure 6-8).

The evaluation metrics (appendix 6.2, table 4) indicate that year of manufacture accounts for a higher proportion of the variance in price (R^2 0.511), with the lowest MAE, MSE and RMSE. The R^2 scores are lower for mileage and engine size (appendix 6.2, table 4), with particularly poor fit for engine size. The best model for year of manufacture and mileage may be non-linear, in contrast to literature (Alhakamy et al., 2023).

Second degree polynomial regression (appendix 6.1, figure 9, 11, 13) improved the fit of the numerical features with lower MAE, MSE, RMSE, for year of manufacture and mileage. The optimal polynomial degree for these features was 7^{th} and 5^{th} (appendix 6.1, figure 10, 12), respectively, achieving best fit. Caveats of using higher degrees include overfitting, failing to generalise, and increased complexity, with noise sensitivity (W3 Schools, n. d.-c).

DT and RF (appendix 6.1, figure 14-17) regression performed slightly below the polynomial models for year of manufacture and mileage.

Analysis combining all numerical features saw RF (appendix 6.1, figure 21) out-perform (R^2 0.938) multiple linear, DT and SVR (appendix 6.1, figure 18-20) with improved accuracy and lowest MAE, MSE, RMSE (appendix 6.2, table 4).

RF of all features provided the best price prediction model (R^2 0.998), superior to numerical regression alone (appendix 6.1, figure 22), correlating with the literature (Chen et al., 2022).

3.2.2 ANN

Training and validation loss and model performance (R^2 0.995) are displayed in appendix 6.1, figure 23, 24. Performance was slightly below RF.

Fit is improved over published neural networks for car price prediction (Sun et al., 2017; Liu et al., 2022). Some model iterations under-predicted higher prices, indicating underfitting, possibly due to fewer data points. The chosen final architecture generalises well, balancing loss minimisation and good predictive ability, minimising under- and overfitting (AWS, 2023).

3.3 Unsupervised clustering models

3.3.1 k-Means clustering

Optimal k for input feature combinations, along with internal evaluation metrics are detailed in appendix 6.2, table 5.

Silhouette score was highest for year of manufacture and price, however, DB score was lowest for year of manufacture and mileage (appendix 6.1, figure 25-28). DB score difference is small with year of manufacture and price providing the best clustering result, chosen for further analysis.

3.3.2 AHC

AHC (appendix 6.1, figure 29, 30) was performed for year of manufacture and price (appendix 6.2, table 6).

Single linkage scored highly but a small number of data points are included in the second cluster and the distribution is different to those produced by complete, average linkage, and k-Means. Poor data separation means single linkage connects distant points resulting in elongated clusters (Zepeda-Mendoza & Resendis-Antonio, 2013).

Complete linkage performed the best clustering with higher scores than k-Means., with optimal k and cluster distribution comparable.

3.3.3 MS and DBSCAN

Over a wide range of bandwidths, the MS model failed. All data points were assigned to the same cluster because the data distribution does not have clear density variations (Comaniciu & Meer, 2002). DBSCAN also failed (Schubert et al., 2017).

4 Conclusion

Polynomial regression was superior in price prediction of single numerical features with year of manufacture accounting for most variance. Improved prediction was achieved using RF for all numerical features combined. Inclusion of the categorical features for RF provided the greatest prediction accuracy of all models, with ANN also performing well.

Complete linkage AHC for year of manufacture and price produced the best clustering, showing similarity in data distribution and clustering to k-Means. Single linkage AHC clustering highlights the importance of visual inspection alongside evaluation metrics for appropriate model selection.

Word Count 1452

5 References

Alhakamy, A., Alhowaity, A., Alatawi, A. A. & Alsaadi, H. (2023) Are used cars more sustainable? Price prediction based on linear regression. *Sustainability*, 15(2).

AWS (2023) *What is overfitting?* Available online: https://aws.amazon.com/what-is/overfitting/ [Accessed 11/12/2023].

Bukvic, L., Skrinjar, J. P., Fratrovic, T. & Abramovic, B. (2022) Price prediction and classification of used-vehicles using supervised machine learning. *Sustainability*, 14(24).

Car Dealer Magazine (2023) *Used car dealers slashing prices unnecessarily over fears of 'disconnect' between trade and retail values.* Available online: https://cardealermagazine.co.uk/publish/used-car-dealers-are-slashing-prices-unnecessarily-over-fears-of-disconnect-between-trade-and-retail-values/295024 [Accessed 19/12].

Chen, J., Li, F. F., Xu, J., Wang, Q., Han, Q. Z. & Yan, M. (2022) Comparisons of different methods used for second-hand car price prediction, *2nd International Conference on Applied Mathematics, Modelling, and Intelligent Computing (CAMMIC)*. Electr Network, Mar 25-27.

Codecademy (n. d.-a) *Clustering: K-Means.* Available online: https://www.codecademy.com/learn/machine-learning/modules/dspath-clustering/cheatsheet [Accessed 11/12/2023].

Codecademy (n. d.-b) *Decision trees for classification and regression.* Available online: https://www.codecademy.com/article/mlfun-decision-trees-article [Accessed 11/12/2023].

Comaniciu, D. & Meer, P. (2002) Mean shift: A robust approach toward feature space analysis. *Ieee Transactions on Pattern Analysis and Machine Intelligence*, 24(5), 603-619.

Geeks for Geeks (n. d.-a) *Dropout in neural networks.* Available online: https://www.geeksforgeeks.org/dropout-in-neural-networks/ [Accessed 11/12/2023].

Geeks for Geeks (n. d.-b) *Random forest regression in Python.* Available online: https://www.geeksforgeeks.org/random-forest-regression-in-python/ [Accessed 11/12/2023].

Geeks for Geeks (n. d.-c) *Regularization in machine learning.* Available online: https://www.geeksforgeeks.org/regularization-in-machine-learning/ [Accessed 11/12/2023].

Gegic, E., Isakovic, B., Keco, D., Masetic, Z. & Kevric, J. (2019) Car price prediction using machine learning techniques. *Tem Journal-Technology Education Management Informatics*, 8(1), 113-118.

HowStuffWorks (2023) *How Occam's razor works.* Available online: https://science.howstuffworks.com/innovation/scientific-experiments/occams-razor.htm [Accessed 11/12/2023].

Hyperskill (n. d.) *Introduction to clustering evaluation.* Available online: https://hyperskill.org/learn/step/28809 [Accessed 13/12/2023].

Keras (n. d.-a) *Adadelta.* Available online: https://keras.io/api/optimizers/adadelta/ [Accessed 11/12/2023].

Keras (n. d.-b) *Adagrad.* Available online: https://keras.io/api/optimizers/adagrad/ [Accessed 11/12/2023].

Keras (n. d.-c) *Adam.* Available online: https://keras.io/api/optimizers/adam/ [Accessed 11/12/2023].

Keras (n. d.-d) *Early stopping.* Available online: https://keras.io/api/callbacks/early_stopping/ [Accessed 11/12/2023].

Keras (n. d.-e) *RMSprop.* Available online: https://keras.io/api/optimizers/rmsprop/ [Accessed 11/12/2023].

Liu, E. C., Li, J., Zheng, A. N., Liu, H. R. & Jiang, T. (2022) Research on the prediction model of the used car price in view of the PSO-GRA-BP neural network. *Sustainability*, 14(15).

Montesinos López, O. A., Montesinos López, A. & Crossa, J. (2022) Fundamentals of artificial neural networks and deep learning, *Multivariate statistical machine learning methods for genomic prediction*. Cham: Springer International Publishing, 379-425.

Schröder, L., Dimitrov, N. K., Verelst, D. R., Sorensen, J. A. & Iop (2018) Wind turbine site-specific load estimation using artificial neural networks calibrated by means of high-fidelity load simulations, *7th Conference on Science of Making Torque from Wind (TORQUE)*. Milan, ITALY, Jun 20-22.

Schubert, E., Sander, J., Ester, M., Kriegel, H. P. & Xu, X. W. (2017) DBSCAN revisited, revisited: Why and how you should (still) use DBSCAN. *Acm Transactions on Database Systems*, 42(3).

Scikit learn (n. d.-a) *Clustering.* Available online: https://scikit-learn.org/stable/modules/clustering.html [Accessed 11/12/2023].

Scikit learn (n. d.-b) *Cross-validation: evaluating estimator performance.* Available online: https://scikit-learn.org/stable/modules/cross_validation.html [Accessed 14/12/2023].

Scikit learn (n. d.-c) *Decision tree regression.* Available online: https://scikit-learn.org/stable/auto_examples/tree/plot_tree_regression.html [Accessed 11/12/2023].

Scikit learn (n. d.-d) *sklearn.cluster.AgglomerativeClustering.* Available online: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html [Accessed 13/12/2023].

Scikit learn (n. d.-e) *sklearn.cluster.DBSCAN.* Available online: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html [Accessed 13/12/2023].

Scikit learn (n. d.-f) *sklearn.cluster.KMeans.* Available online: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html [Accessed 13/12/2023].

Scikit learn (n. d.-g) *sklearn.cluster.MeanShift.* Available online: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.MeanShift.html [Accessed 14/12/2023].

Scikit learn (n. d.-h) *sklearn.ensemble.RandomForestRegressor.* Available online: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html [Accessed 11/12/2023].

Scikit learn (n. d.-i) *sklearn.linear_model.ElasticNetCV.* Available online: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.ElasticNetCV.html [Accessed 11/12/2023].

Scikit learn (n. d.-j) *sklearn.linear_model.Lasso.* Available online: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Lasso.html [Accessed 11/12/2023].

Scikit learn (n. d.-k) *sklearn.linear_model.Ridge.* Available online: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Ridge.html [Accessed 11/12/2023].

Scikit learn (n. d.-l) *sklearn.metrics.davies_bouldin_score.* Available online: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.davies_bouldin_score.html [Accessed 13/12/2023].

Scikit learn (n. d.-m) *sklearn.metrics.mean_absolute_error.* Available online: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_absolute_error.html [Accessed 13/12/2023].

Scikit learn (n. d.-n) *sklearn.metrics.mean_squared_error.* Available online: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_squared_error.html [Accessed 13/12/2023].

Scikit learn (n. d.-o) *sklearn.metrics.r2_score.* Available online: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.r2_score.html [Accessed 13/12/2023].

Scikit learn (n. d.-p) *sklearn.metrics.silhouette_score.* Available online: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html [Accessed 13/12/2023].

Scikit learn (n. d.-q) *sklearn.model_selection.ParameterGrid.* Available online: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.ParameterGrid.html [Accessed 14/12/2023].

Scikit learn (n. d.-r) *sklearn.model_selection.RandomizedSearchCV.* Available online: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html [Accessed 11/12/2023].

Scikit learn (n. d.-s) *sklearn.preprocessing.LabelEncoder.* Available online: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html [Accessed 12/12/2023].

Scikit learn (n. d.-t) *sklearn.preprocessing.MinMaxScaler.* Available online: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html [Accessed 13/12/2023].

Scikit learn (n. d.-u) *sklearn.preprocessing.StandardScaler.* Available online: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html [Accessed 13/12/2023].

Scikit learn (n. d.-v) *sklearn.svm.SVR.* Available online: https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR.html [Accessed 11/12/2023].

Scikit learn (n. d.-w) *Supervised learning.* Available online: https://scikit-learn.org/stable/supervised_learning.html [Accessed 11/12/2023].

Scikit learn (n. d.-x) *Support vector machines.* Available online: https://scikit-learn.org/stable/modules/svm.html [Accessed 11/12/2023].

Shen, Z. W., Yang, H. Z. & Zhang, S. J. (2021) Neural network approximation: Three hidden layers are enough. *Neural Networks*, 141, 160-173.

Sun, N., Bai, H. X., Geng, Y. X. & Shi, H. (2017) Price evaluation model in second-hand car system based on BP neural network theory, *18th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD)*. Kanazawa, JAPAN, Jun 26-28.

TensorFlow (n. d.) *tf.keras.activations.ReLU.* Available online: https://www.tensorflow.org/api_docs/python/tf/keras/activations/relu [Accessed 11/12/2023].

W3 Schools (n. d.-a) *Data science - statistics correlation matrix.* Available online: https://www.w3schools.com/datascience/ds_stat_correlation_matrix.asp [Accessed 11/12/2023].

W3 Schools (n. d.-b) *Machine learning - linear regression.* Available online: https://www.w3schools.com/python/python_ml_linear_regression.asp [Accessed 11/12/2023].

W3 Schools (n. d.-c) *Machine learning - polynomial regression.* Available online: https://www.w3schools.com/python/python_ml_polynomial_regression.asp [Accessed 11/12/2023].

Zepeda-Mendoza, M. L. & Resendis-Antonio, O. (2013) Hierarchical agglomerative clustering, in Dubitzky, W., Wolkenhauer, O., Cho, K.-H. & Yokota, H. (eds), *Encyclopedia of systems biology*. New York, NY: Springer New York, 886-887.

6 Appendix

6.1 Figures

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50000 entries, 0 to 49999
Data columns (total 7 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Manufacturer          50000 non-null  object
 1   Model                 50000 non-null  object
 2   Engine size (litres)  50000 non-null  float64
 3   Fuel type             50000 non-null  object
 4   Year of manufacture   50000 non-null  int64
 5   Mileage               50000 non-null  int64
 6   Price (GBP)           50000 non-null  int64
dtypes: float64(1), int64(3), object(3)
memory usage: 2.7+ MB
```

Figure 1. Used car dataframe features

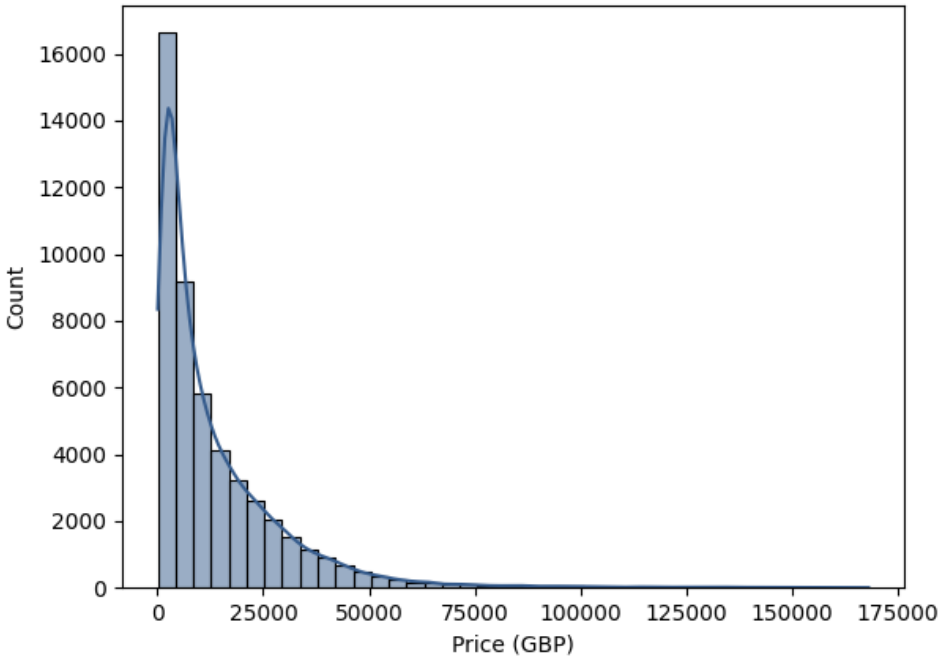

Figure 2. Price distribution

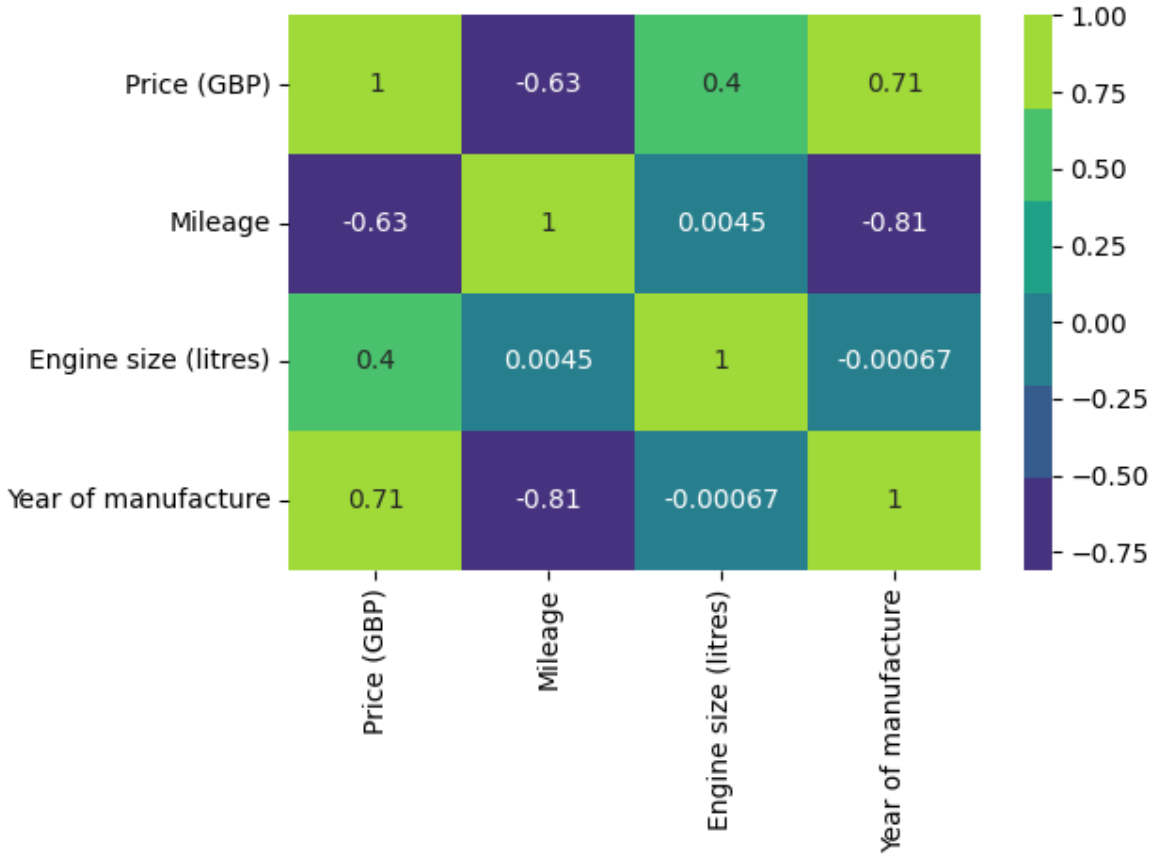

Figure 3. Correlation between numerical features

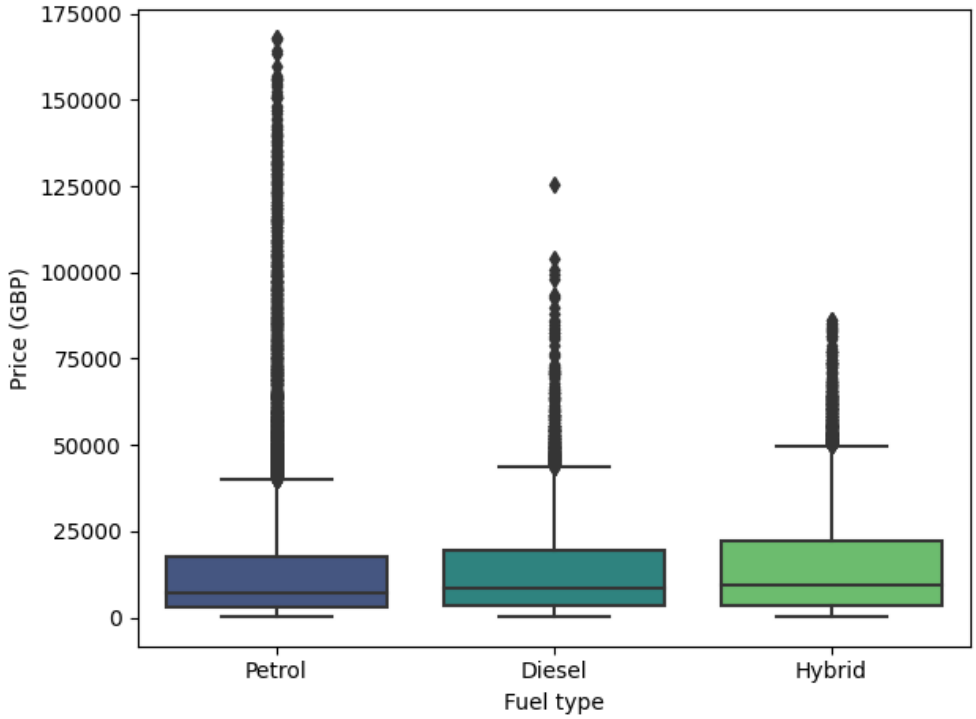

Figure 4. Boxplot of price by fuel type

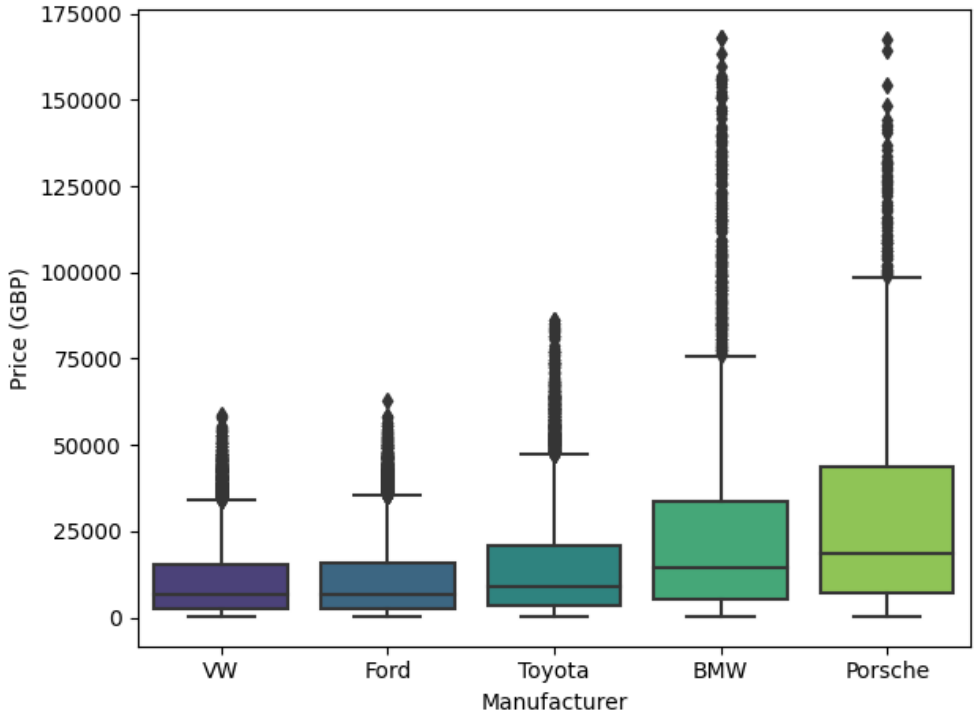

Figure 5. Boxplot of price by manufacturer

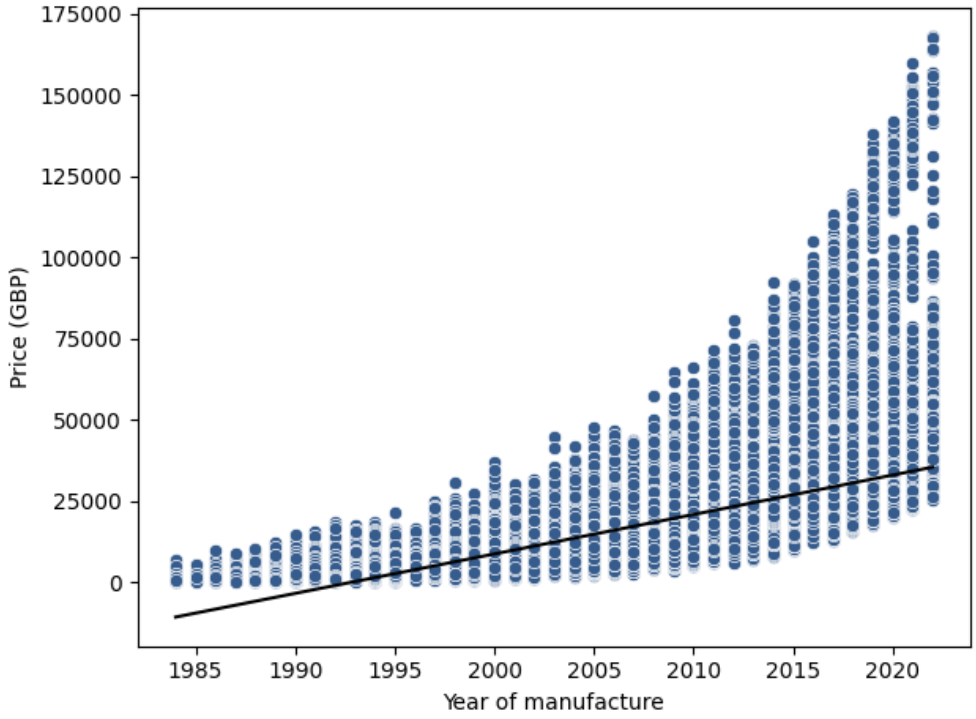

Figure 6. Linear regression for year of manufacture and price

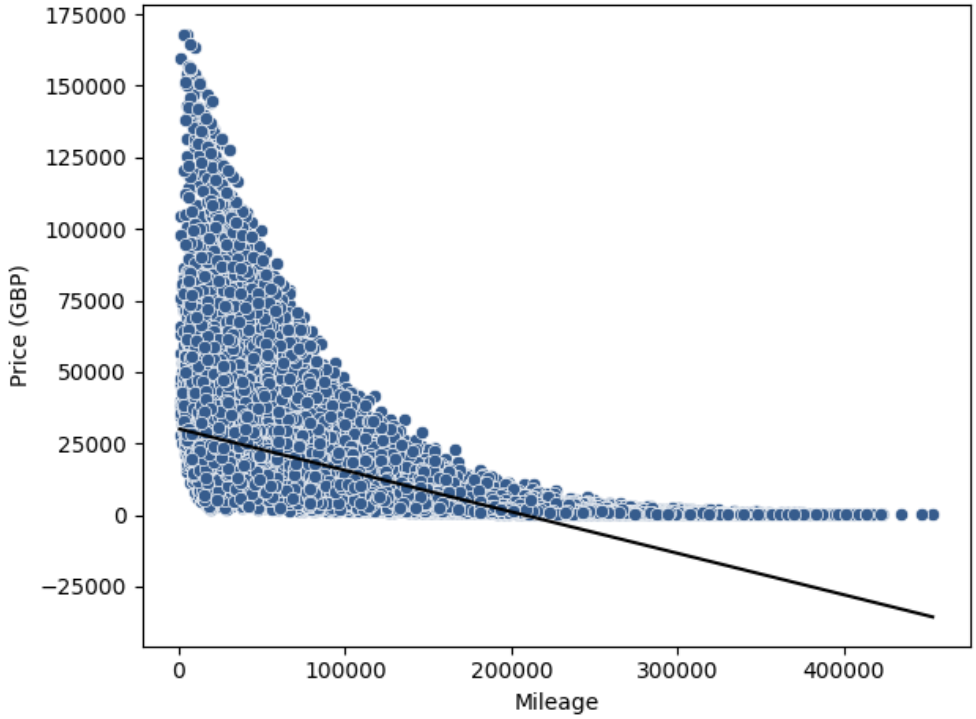

Figure 7. Linear regression for mileage and price

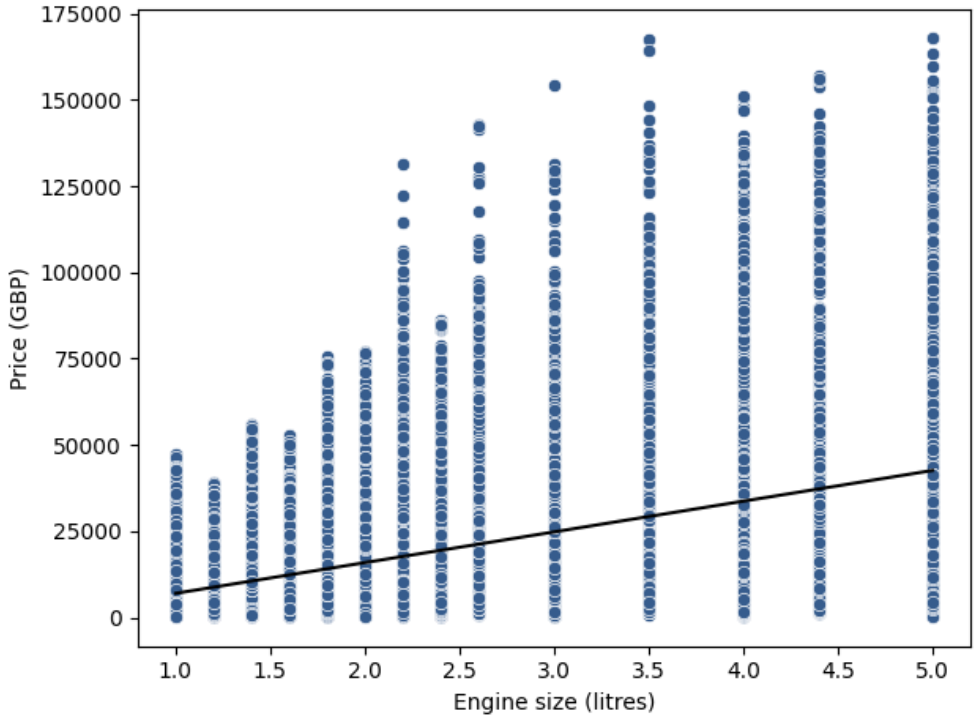

Figure 8. Linear regression for engine size and price

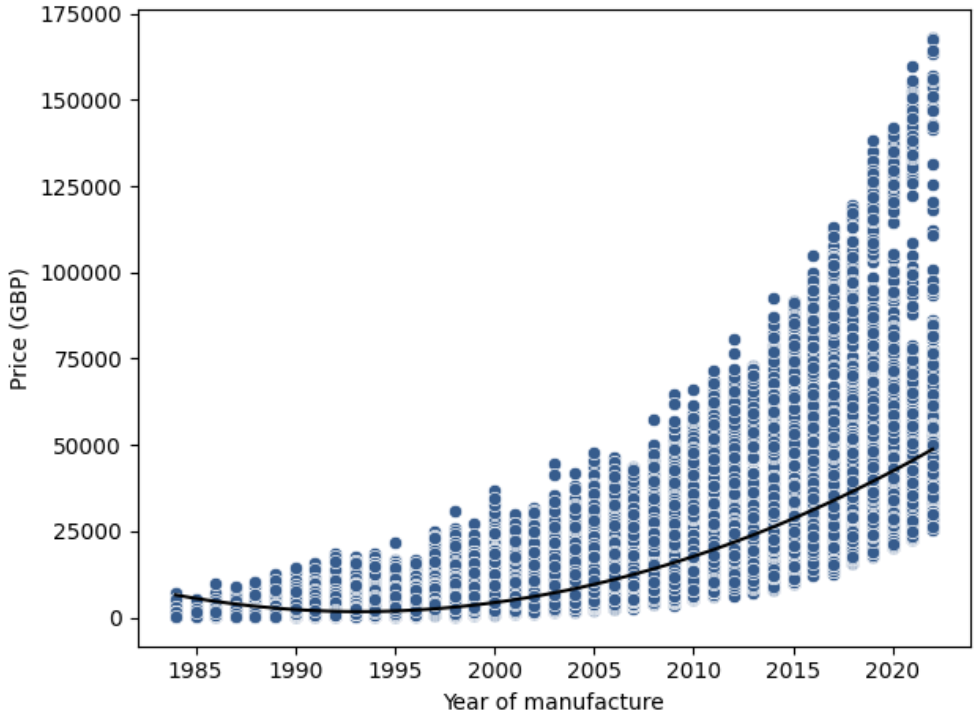

Figure 9. Polynomial regression, 2nd degree, for year of manufacture and price

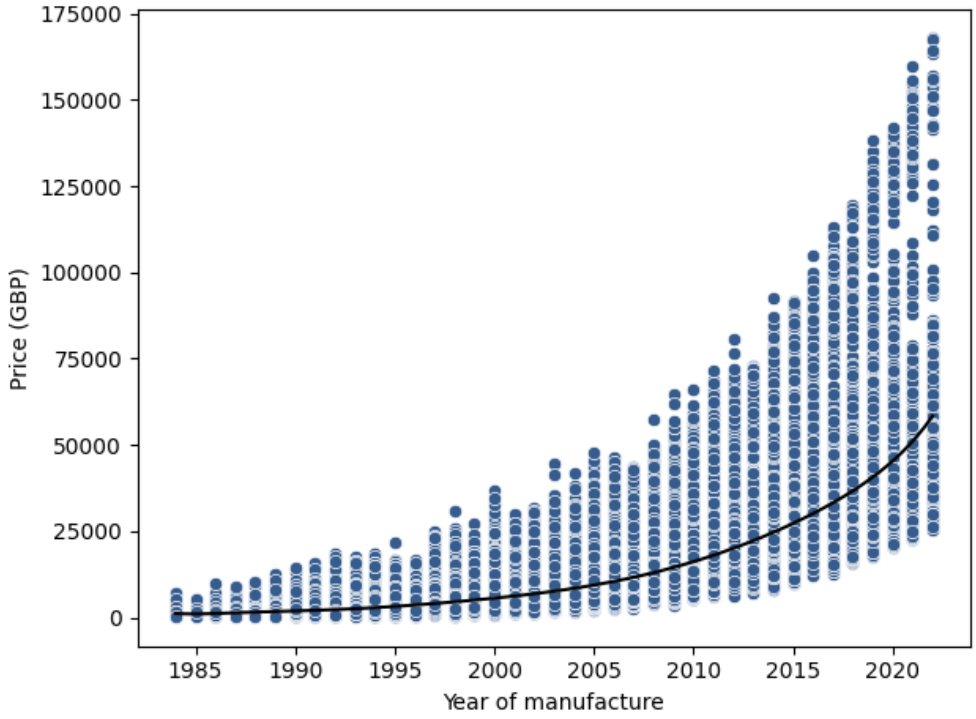

Figure 10. Polynomial regression, 7th degree, for year of manufacture and price

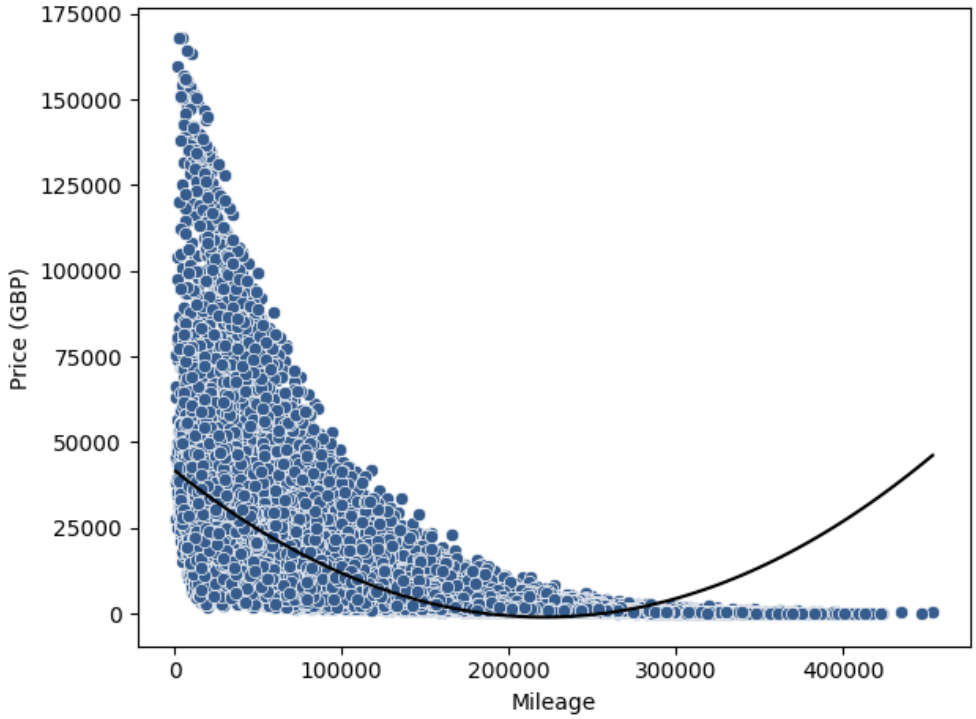

Figure 11. Polynomial regression, 2nd degree, for mileage and price

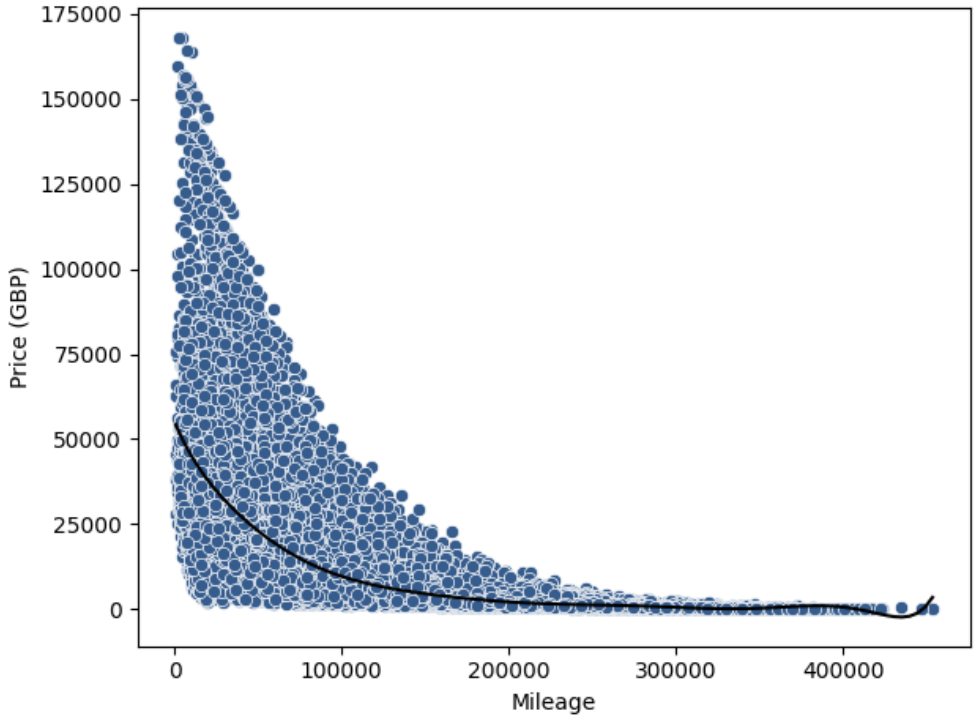

Figure 12. Polynomial regression, 5th degree, for mileage and price

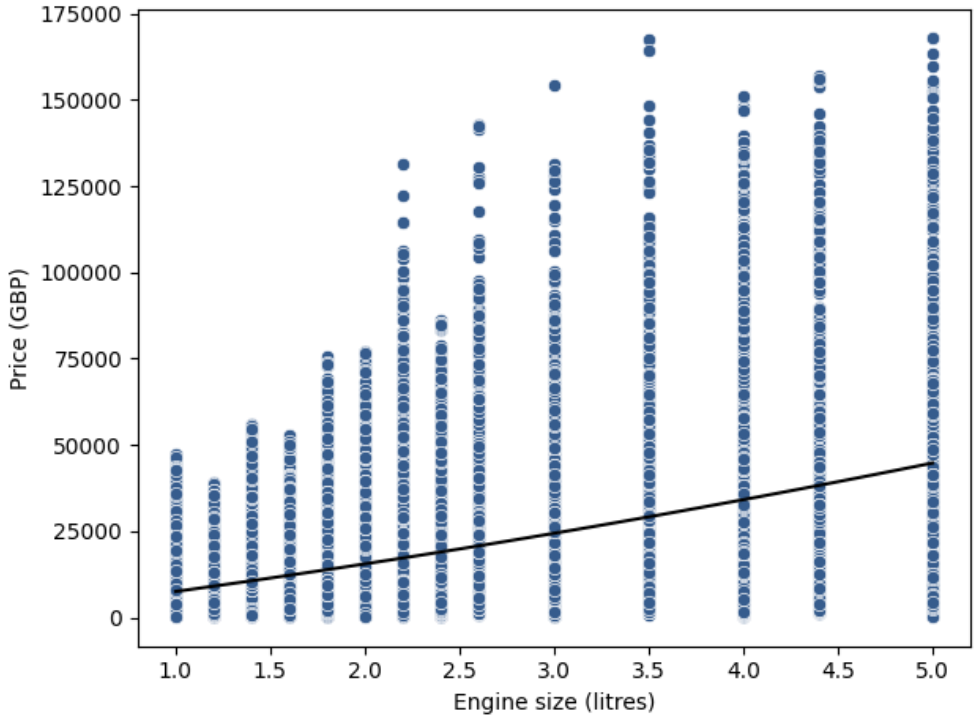

Figure 13. Polynomial regression, 2nd degree, for engine size and price

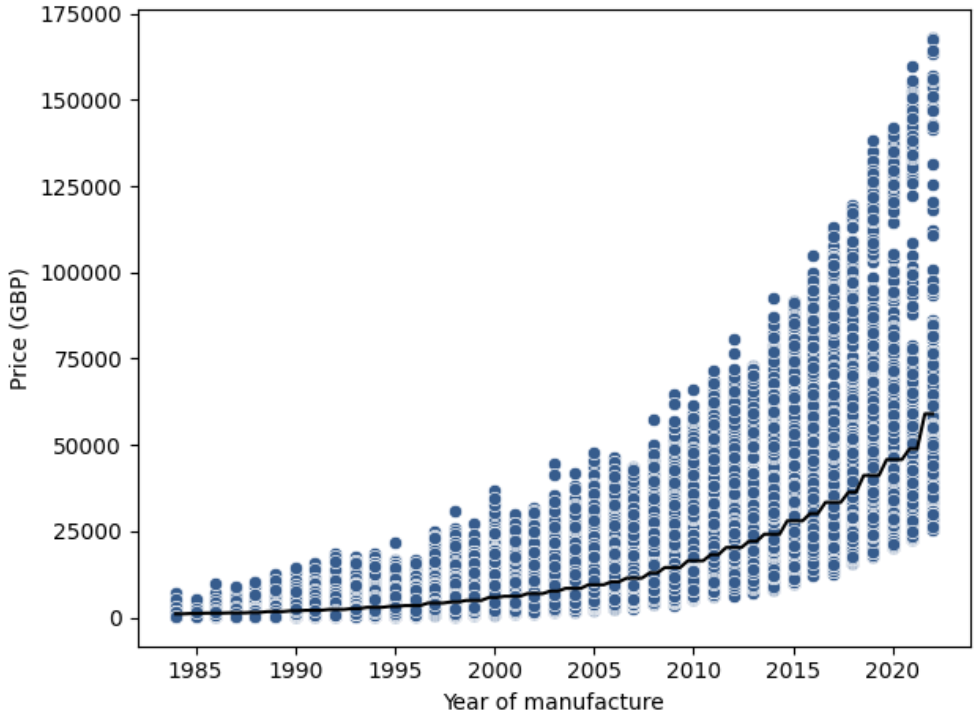

Figure 14. Decision tree regression of year of manufacture and price

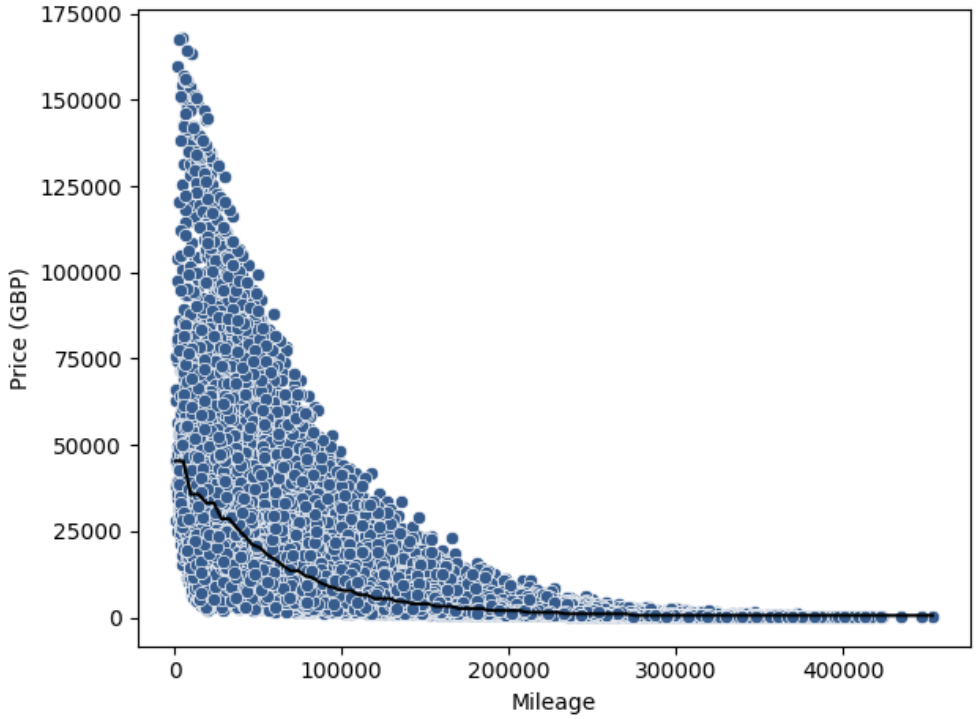

Figure 15. Decision tree regression of mileage and price

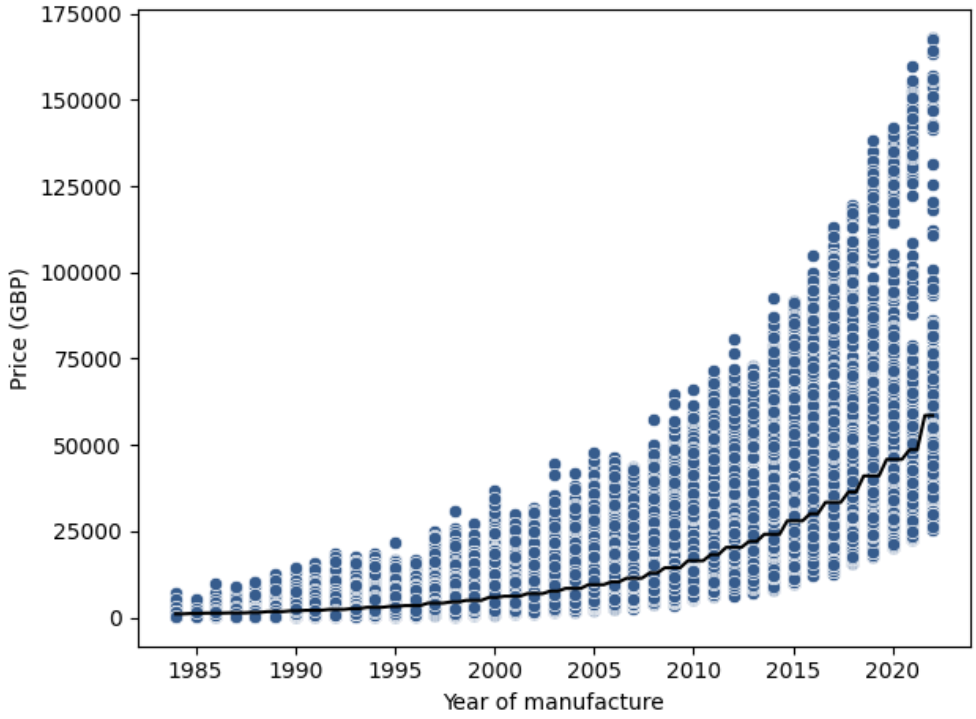

Figure 16. Random forest regression of year of manufacture and price

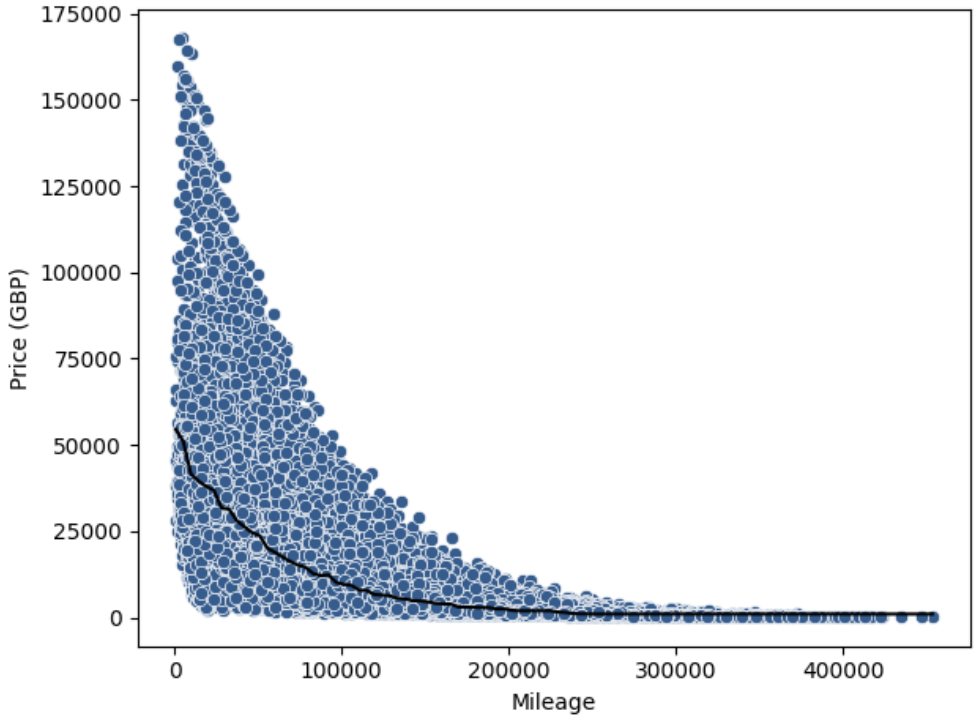

Figure 17. Random forest regression of mileage and price

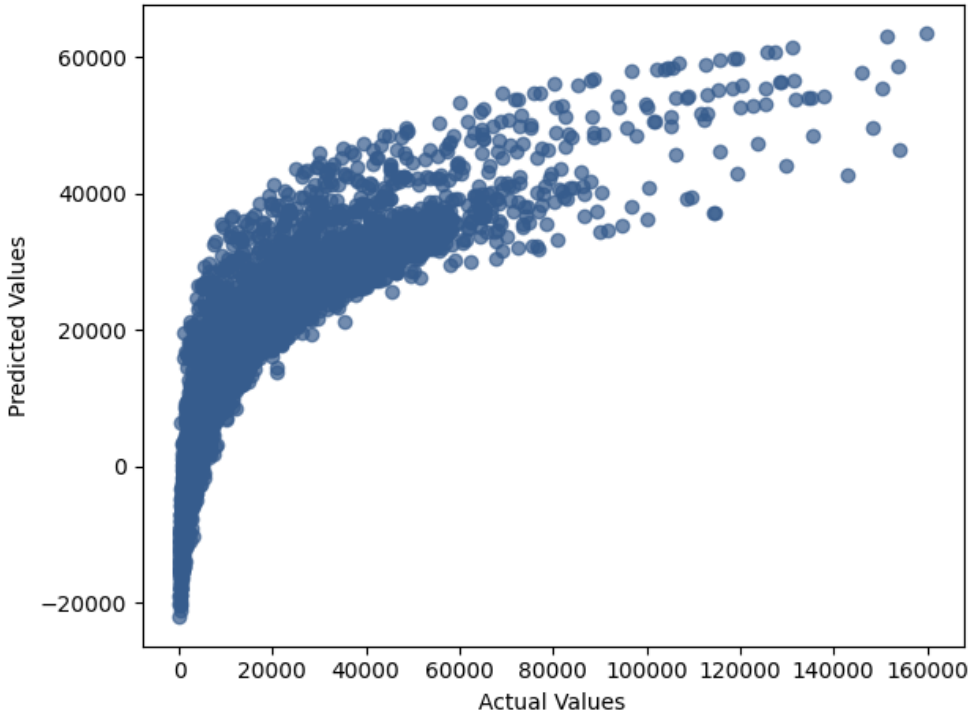

Figure 18. Multiple linear regression of all numerical features for price prediction

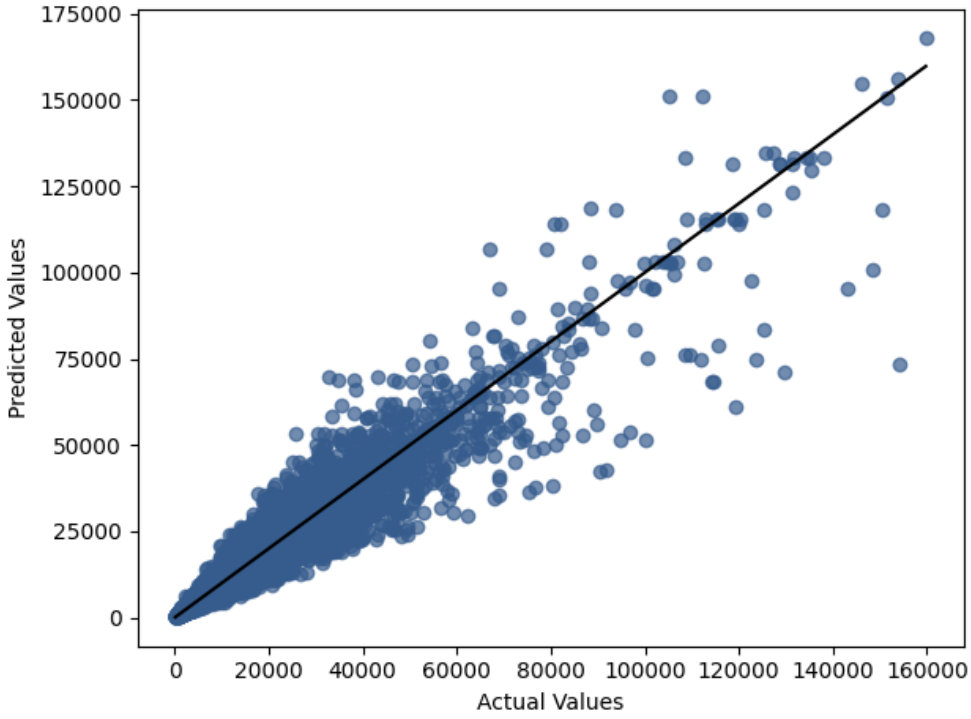

Figure 19. Decision tree regression of all numerical features for price prediction

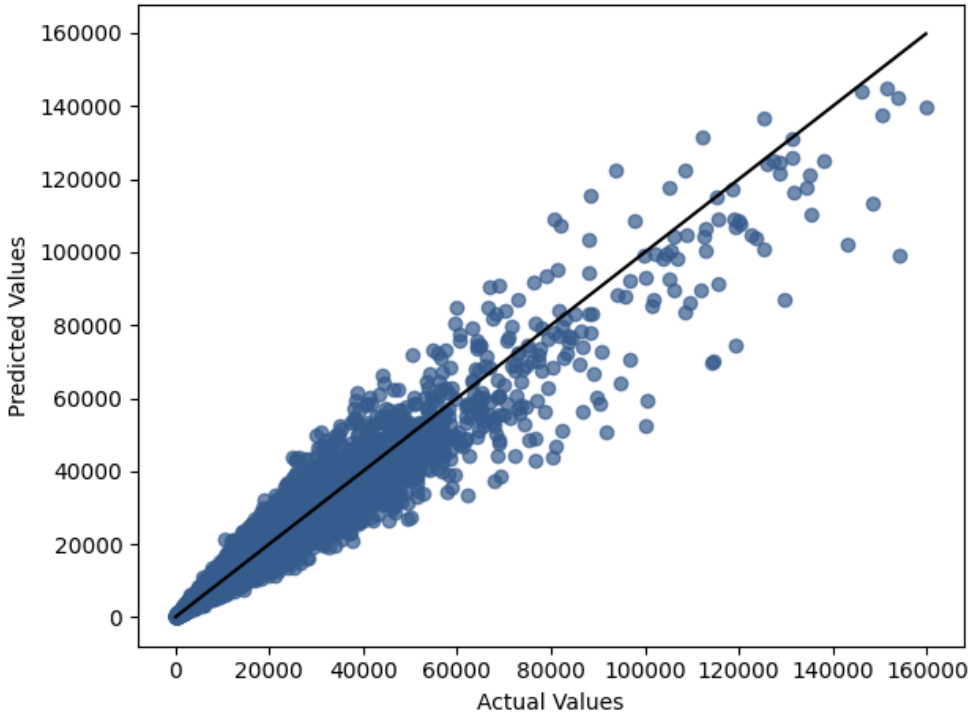

Figure 20. Random forest regression of all numerical features for price prediction

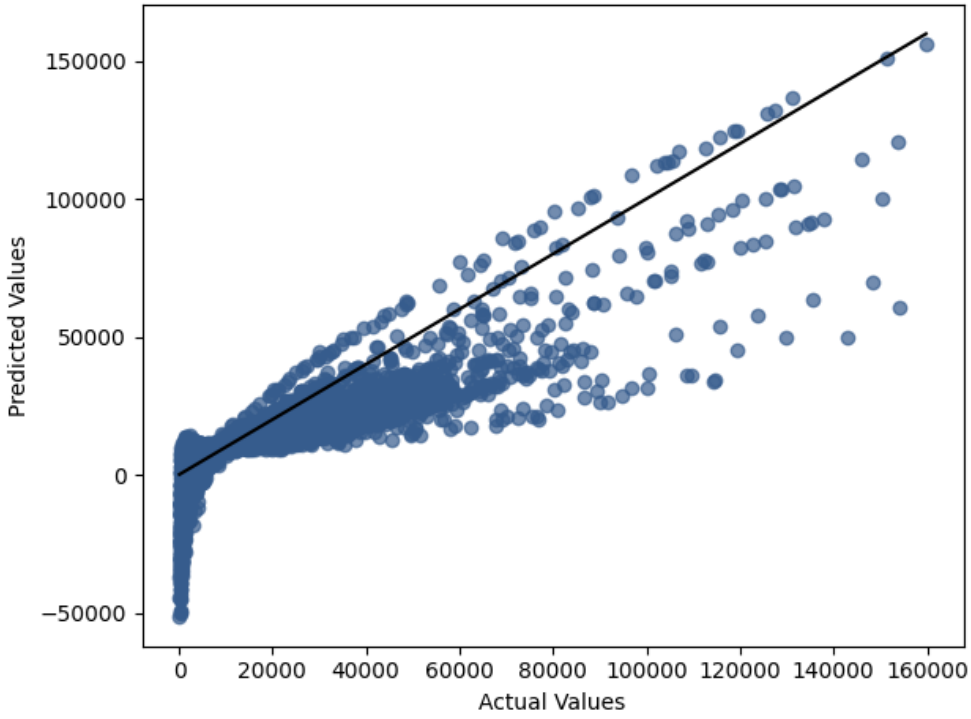

Figure 21. Support vector regression of all numerical features for price prediction

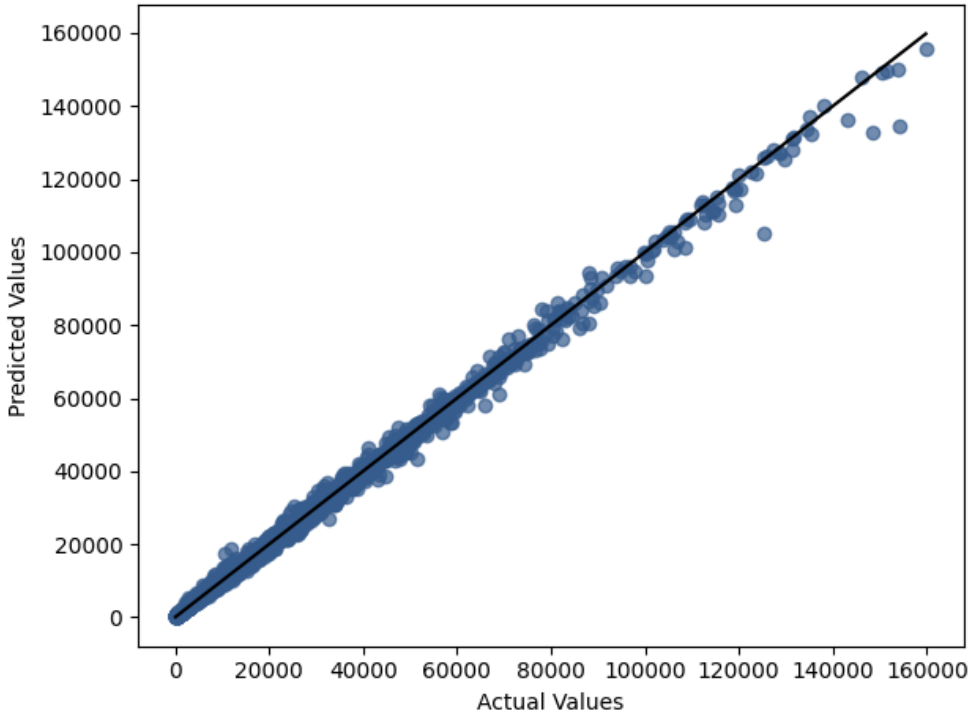

Figure 22. Random forest regression of all features (numerical and categorical) for price prediction

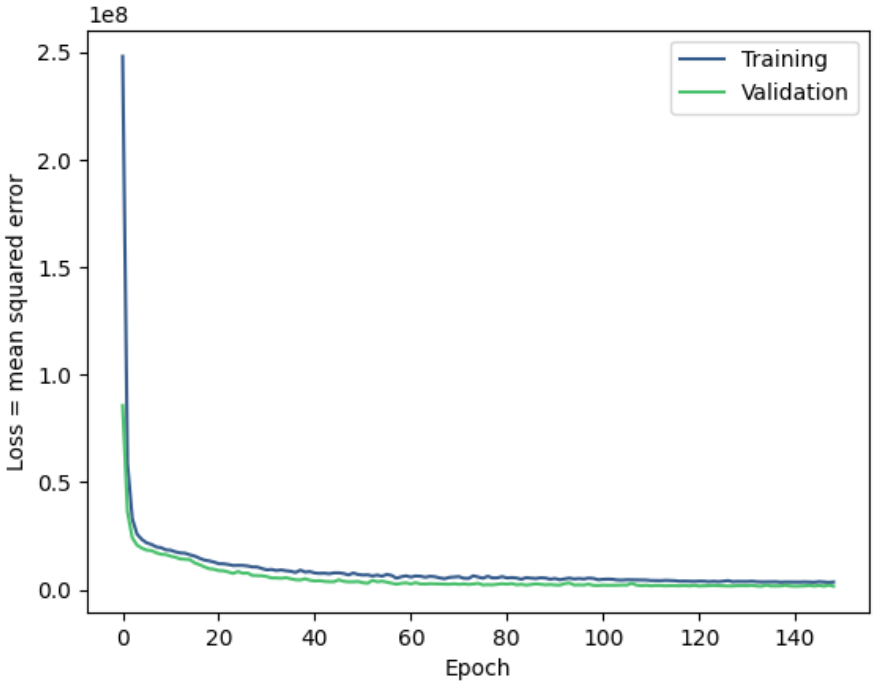

Figure 23. Mean squared error loss by epoch for optimal ANN

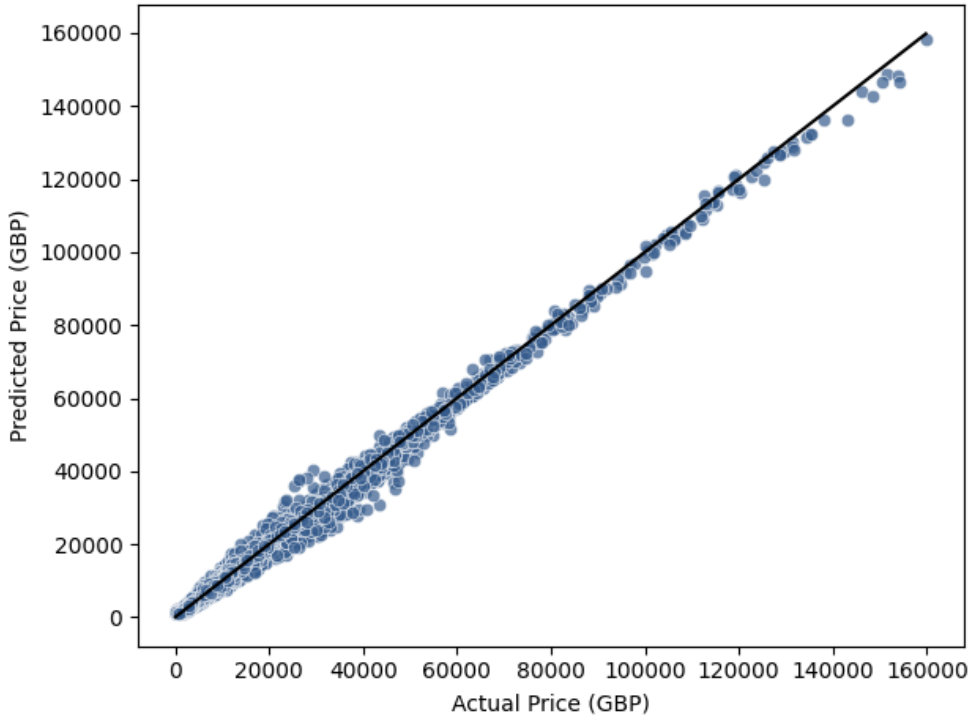

Figure 24. Actual vs predicted price for optimal ANN (R^2 = 0.995)

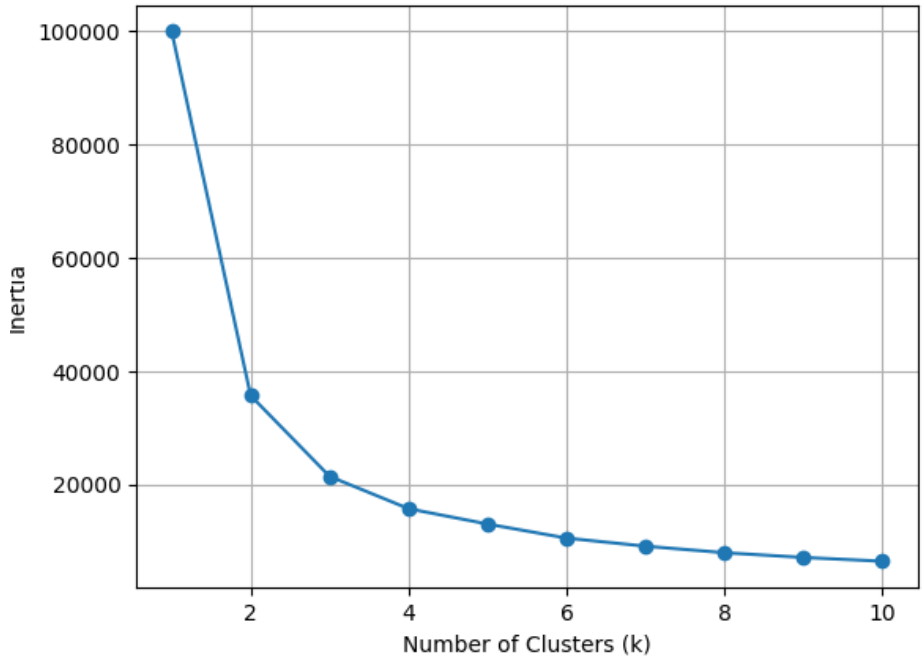

Figure 25. Inertia for optimal number of clusters (k) for year of manufacture and mileage (k-Means clustering)

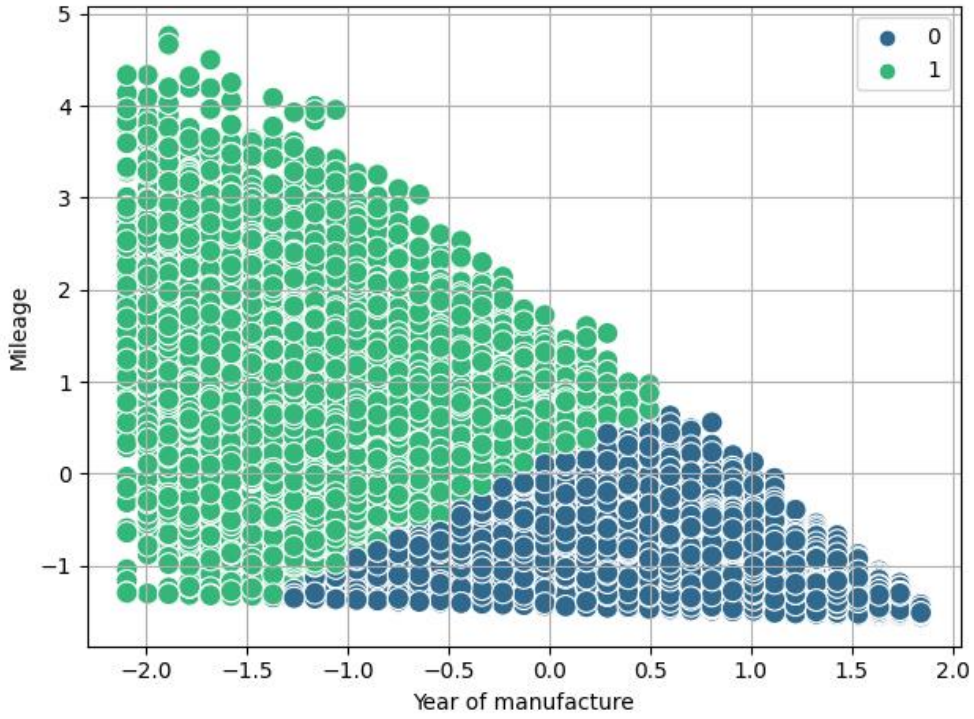

Figure 26. K-Means clustering for year of manufacture and mileage (k=2)

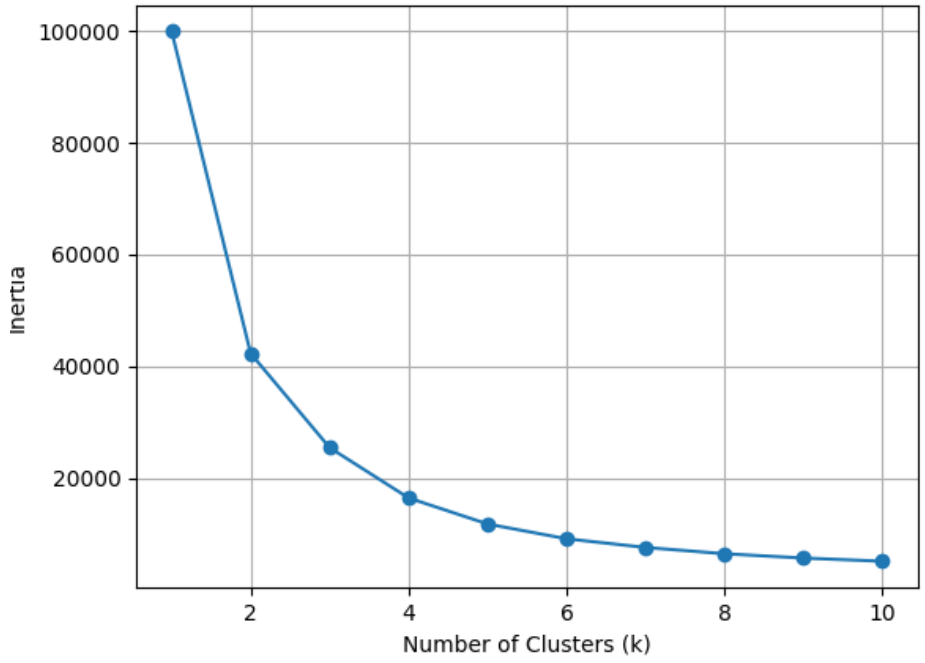

Figure 27. Inertia for optimal number of clusters (k) for year of manufacture and price (k-Means clustering)

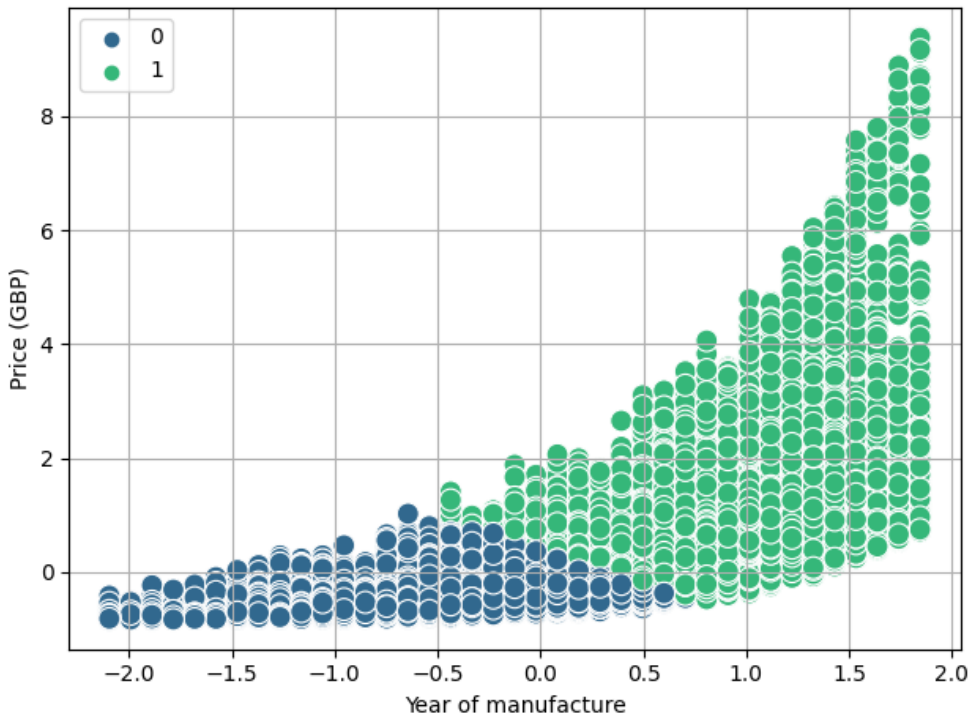

Figure 28. K-Means clustering for year of manufacture and price (k=2)

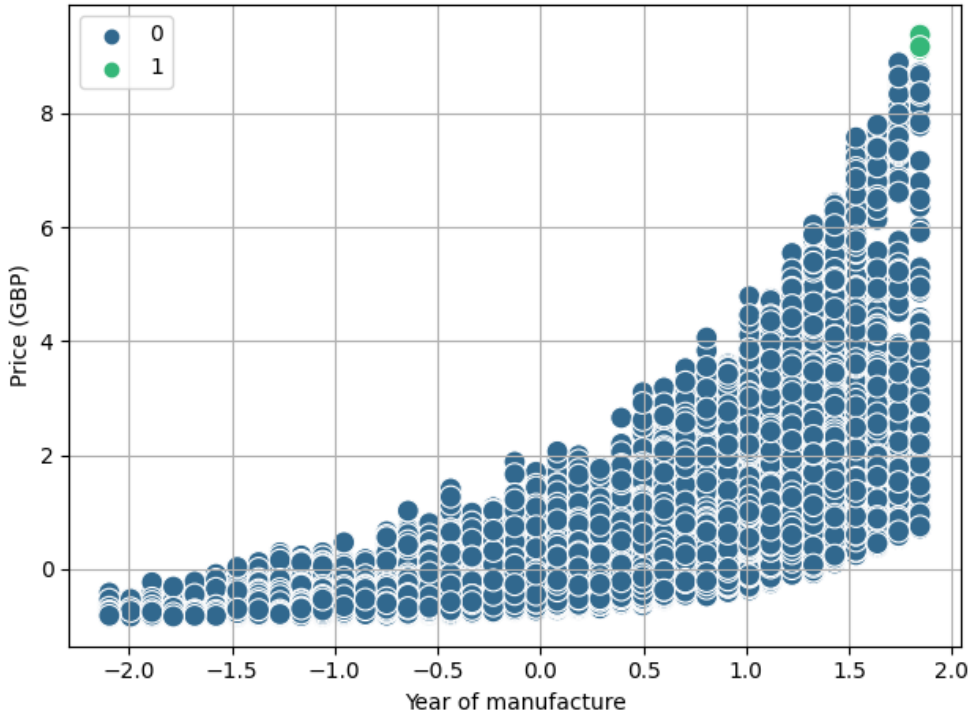

Figure 29. Complete linkage AHC for year of manufacture and price (k=2)

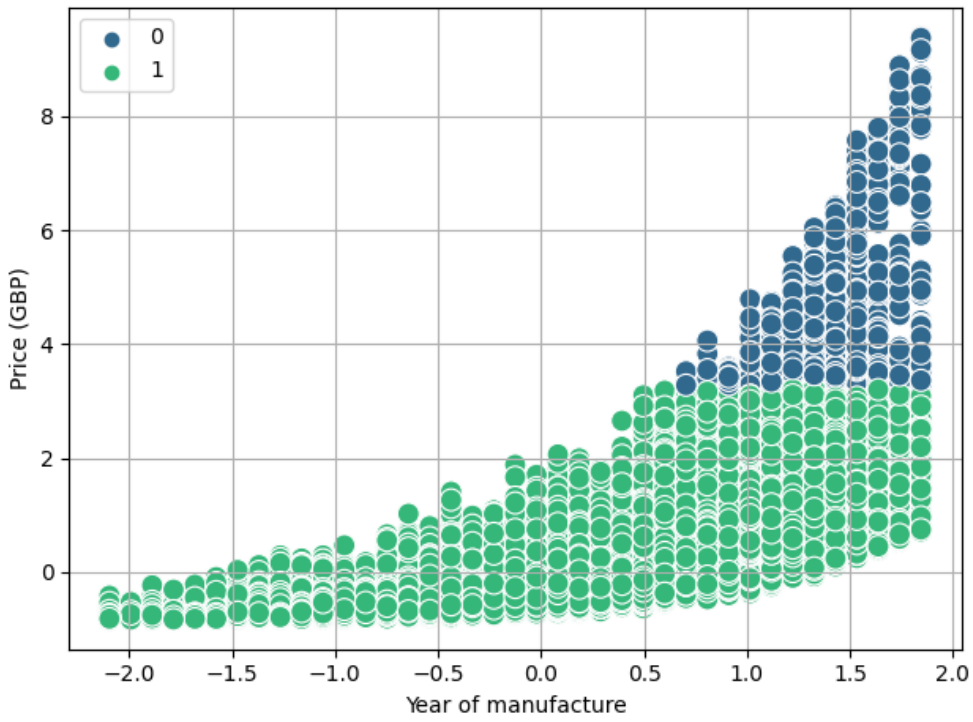

Figure 30. Complete linkage AHC for year of manufacture and price (k=2)

6.2 Tables

Table 1. ANN architecture

Layer	Type	Units	Activation function	Regularisation/ rate	Dropout
Input layer (6 features) **Hidden layer 1**	Dense	64	ReLU	L2/ 0.005	0.1
Hidden layer 2 and 3	Dense	64	ReLU	L2/ 0.005	-
Output layer	Dense	1	Linear	-	-
Epochs = 200 Validation split = 0.1 Optimiser = Adam (0.001) Early stopping, patience = 15					

Table 2. Hyperparameter tuning of regression models

Model	Feature	Min samples split	Min samples leaf	Max features	Max depth	N_estimators
Decision tree	1 num	2	4	log2	50	-
	3 num	10	2	1	30	-
Random forest	1 num	5	1	sqrt	30	100
	3 num	10	2	log2	20	50
	All	5	1	sqrt	30	100

Table 3. ANN optimisation

Layer/ Hyperparameter	Value	R^2	Comment
Hidden layers	2	0.975	No early stopping, bias at higher prices (underpredicting value).
	3	0.991	Early stopping at ~150 epochs, slight bias at very top prices. Chosen for further tuning.
Learning rate (Adam)	0.01	0.885	Reduced fit, increased bias, learning rate too fast.
	0.0001	0.945	Reduced fit, reduced bias, no early stopping.
Dropout rate	0.1	0.991	Slight bias, no early stopping. Chosen for further tuning.
Optimiser	RMSprop	0.994	Increased bias, no early stopping.
	Adagrad	0.393	Poor fit.
	Adadelta	-0.047	No fit.
Regularisation	L1	0.996	Bias minimised, best fit so far, no early stopping.
	L2	0.994	Slight bias, good fit, no early stopping.
	Elastic	0.989	Increased bias, fit reduced, early stopping.
L1 regularisation rate	0.05	0.953	Poor fit, early stopping.
	0.005	0.993	Good fit, early stopping.
Activation function	Leaky ReLU	0.993	L1, rate 0.01, no early stopping.
	eLU	0.994	No early stopping.
	SeLU	0.987	Reduced fit, no early stopping.
	eLU	0.991	L1, rate 0.005, no early stopping.
Early stopping patience	15	0.993	L1, rate 0.01, ReLU, early stopping.
	15	**0.995**	**L1, rate 0.01, ReLU, early stopping.**
Starting parameters: Optimiser = Adam (default 0.001) Early stopping, patience = 20 Dropout = 20 No regularisation (default rate 0.01)			

Table 4. Internal evaluation metrics for regression models

Figure No.	Model type	Input feature/s	MAE	MSE	RMSE	R^2
6	Linear	Year of manufacture	7031.04	132678999.95	11518.64	0.5111
7	Linear	Mileage	7964.78	162468566.87	12746.32	0.4013
8	Linear	Engine Size	10817.49	230499154.45	15182.20	0.1506
9	Poly (d=2)	Year of manufacture	5387.11	105993894.20	10295.33	0.6094
10	Poly (d=7)	Year of manufacture	5157.88	102580976.68	10128.23	0.6220
11	Poly (d=2)	Mileage	6409.91	129620312.16	11385.09	0.5224
12	Poly (d=5)	Mileage	5703.75	120554245.87	10979.72	0.5558
13	Poly (d=2)	Engine size	10807.26	230326166.00	15176.50	0.1513
14	DT	Year of manufacture	5164.72	102782548.88	10138.17	0.6213
15	DT	Mileage	5545.59	127075015.24	11272.76	0.5317
16	RF	Year of manufacture	5162.99	102815430.35	10139.79	0.6211
17	RF	Mileage	5722.09	121022629.64	11001.03	0.5540
18	Multiple linear	Year of manufacture, mileage, engine size	6091.46	89158615.76	9442.38	0.6715
19	DT	Year of manufacture, mileage, engine size	2416.15	23351109.50	4832.30	0.9140
20	RF	Year of manufacture, mileage, engine size	2162.97	16832304.52	4102.72	0.9380
21	SVR	Year of manufacture, mileage, engine size	6484.69	96481063.36	9822.48	0.6445
22	RF	All numerical and categorical	331.52	576664.59	759.38	0.9979

Table 5. Internal evaluation metrics for k-Means clustering

Input features	Optimal k	Silhouette score	DB score
Year of manufacture, engine size	3	0.460	0.753
Year of manufacture, mileage	2	0.533	0.659
Year of manufacture, price	2	0.548	0.678
Engine size, mileage	3	0.450	0.776
Engine size, price	2	0.618	0.942
Mileage, price	2	0.487	0.760

Table 6. Internal evaluation metrics for AHC for year of manufacture and price

Linkage	Optimal k	Silhouette score	DB score
Complete linkage	2	0.691	0.445
Average linkage	2	0.649	0.461
Single linkage	2	0.821	0.136

6.3 Abbreviations

Artificial Neural Network	ANN
Coefficient of determination	R^2
Davies-Bouldin	DB
Decision Tree	DT
Density-based spatial clustering of applications with noise	DBSCAN
Exponential linear unit	eLU
Machine Learning	ML
Mean Absolute Error	MAE
Mean Squared Error	MSE
Meanshift	MS
Random Forest	RF
Rectified linear unit	ReLU
Root Mean Squared Error	RMSE
Scaled exponential linear unit	SeLU
Support Vector Regression	SVR