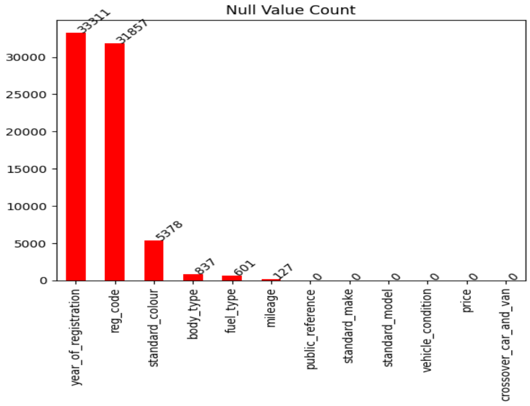
SWATHI ASHOKKUMAR

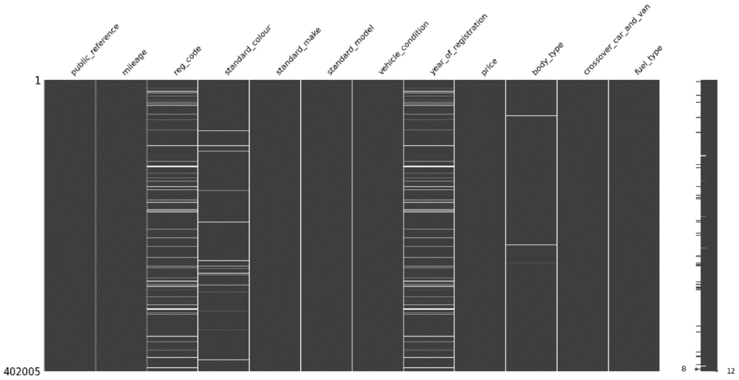
23623012

1.Data Processing for Machine Learning:

We are assuming that the Adverts.csv file contains details about cars sold by AutoTrader in the year 2021. Auto Trader is one of the UK's largest automotive marketplaces, it enables buying and selling of new and used vehicles by private sellers and marketplaces.

1.Dealing with Missing Values: The dataset contains several missing values across six columns, with the 'reg_code' and 'year_of_registration' columns exhibiting the highest count of null entries. The Null value count bar chart shows the count of missing values of all the features in the dataset.



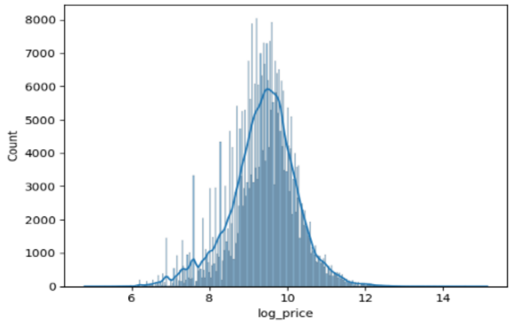


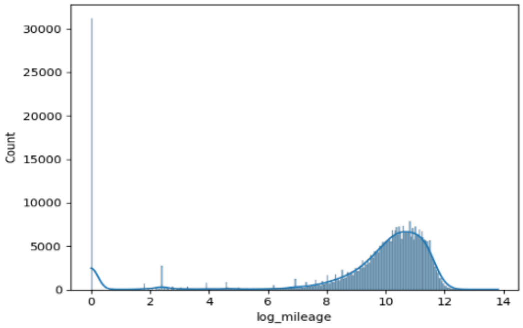
Absences in 'reg_code' and 'year_of_registration' primarily occur for new cars, presumed unregistered. For new cars, 2021 is assumed for registration, and a standardized registration code '21' is assigned.

A logical mapping fills gaps for used cars, applying historical norms to align 'reg_code' with 'year_of_registration.' Remaining missing values are treated with the mode using simple Imputer, consistent with the dataset's central tendencies.

2.Dealing with Outliers, and Noise:

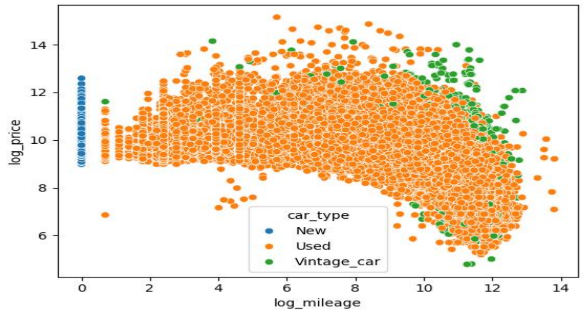
- Excluded cars outside the 1892–2021-year range to eliminate anomalies.
- Applied logarithmic transformations to 'mileage' and 'price' to normalize distributions and reduce outlier impacts.
- Estimated zero-mileage for 355 used vehicles based on average yearly usage, ensuring no new vehicles erroneously reported mileage.

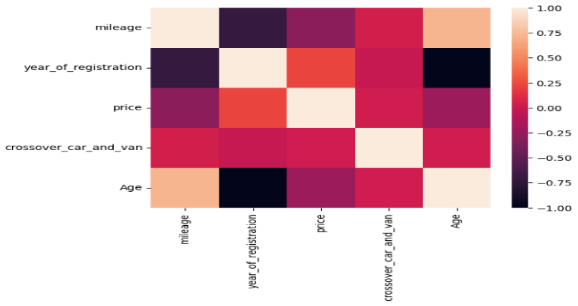




2.Deriving informative features:

We created 'Age' from 'year_of_registration' to quantify vehicle lifespan. 'Car_type' classifies vehicles into new, used, or vintage, with a 20-year threshold for vintage, reflective of price trends. A composite feature, 'Make_Model', merges 'Standard_make' and 'Standard_model' for model precision. 'Standard_colour' sees infrequent colours consolidated under 'other' to streamline categories and aid model performance.



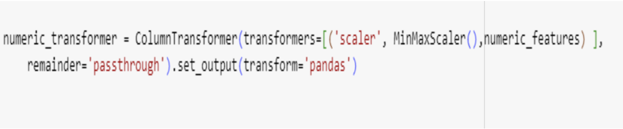


2. Feature Engineering:

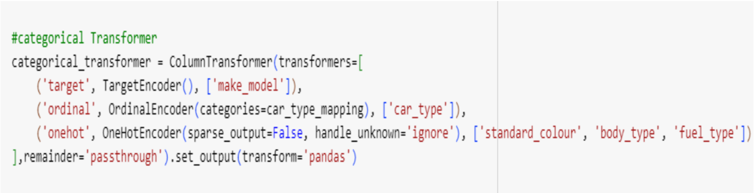
Feature Selection:

The dataset was refined by consolidating 'standard_make' and 'standard_model' into 'make_model' and removing 'vehicle_condition' in Favor of 'car_type', which categorizes vehicles by age. 'Year_of_registration' was dropped for redundancy, replaced by 'Age', and we transitioned to log_transformed 'log_price' and 'log_mileage' for better data normalization. Infrequent colors were grouped into a new column, and irrelevant features like 'crossover_van_and_car' were discarded due to their minimal impact on the target variable.

Numerical Transformation and Categorical Encoding: Numeric features were scaled using MinMaxScaler to ensure uniformity, essential for models sensitive to feature scaling. Categorical variables were converted to numeric formats: 'Make_model' received target encoding to manage its categories efficiently, 'car_type' was ordinally encoded to reflect the natural order of vehicle valuation, and attributes like 'standard_colours', 'body_type', and 'fuel_type' were one-hot encoded due to their lower cardinality.



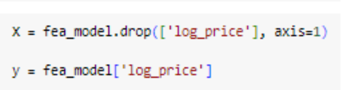
```
numeric_transformer = ColumnTransformer(transformers=[('scaler', MinMaxScaler(),numeric_features) ],
    remainder='passthrough').set_output(transform='pandas')
```



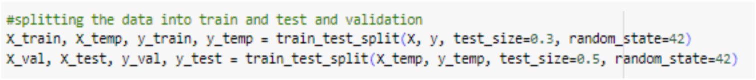
```
#categorical Transformer
categorical_transformer = ColumnTransformer(transformers=[
    ('target', TargetEncoder(), ['make_model']),
    ('ordinal', OrdinalEncoder(categories=car_type_mapping), ['car_type']),
    ('onehot', OneHotEncoder(sparse_output=False, handle_unknown='ignore'), ['standard_colour', 'body_type', 'fuel_type'])
],remainder='passthrough').set_output(transform='pandas')
```

A Preprocessor was created embedding both numerical and the categorical transformer.

Train-Test-Split: We have split the fea_model dataframe into X and Y, with X Containing the features and Y containing only the target variable price. And we have further split the X and y into train, test and validation set with training set containing 70 percent of the data and validation and the test set each containing 15 percent of the data.



```
X = fea_model.drop(['log_price'], axis=1)

y = fea_model['log_price']
```

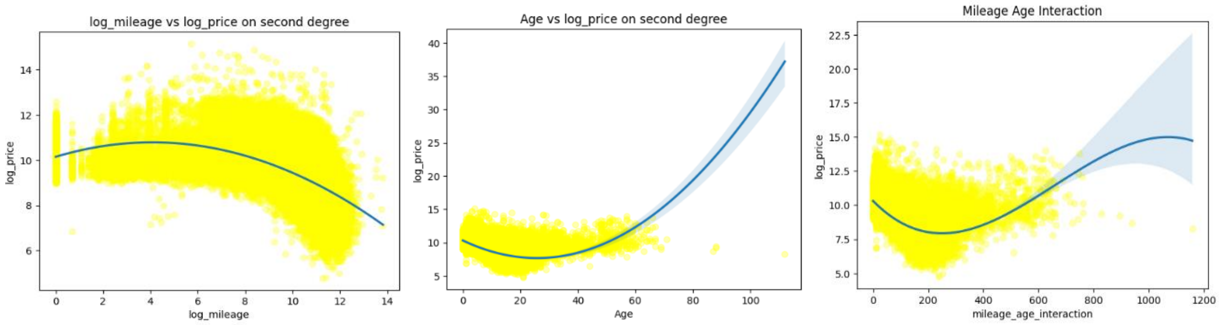


```
#splitting the data into train and test and validation
X_train, X_temp, y_train, y_temp = train_test_split(X, y, test_size=0.3, random_state=42)
X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp, test_size=0.5, random_state=42)
```

We have fitted and transformed the X_train and y_train in the preprocessor and the transformed columns are listed below.

Polynomial and Interaction Features:

I have introduced polynomial interactions to capture non-linear relationships between features and the target variable **log_price.** Specifically, interactions between the features Age and log_mileage were considered, given their potential combined effect on the log_price of vehicles.



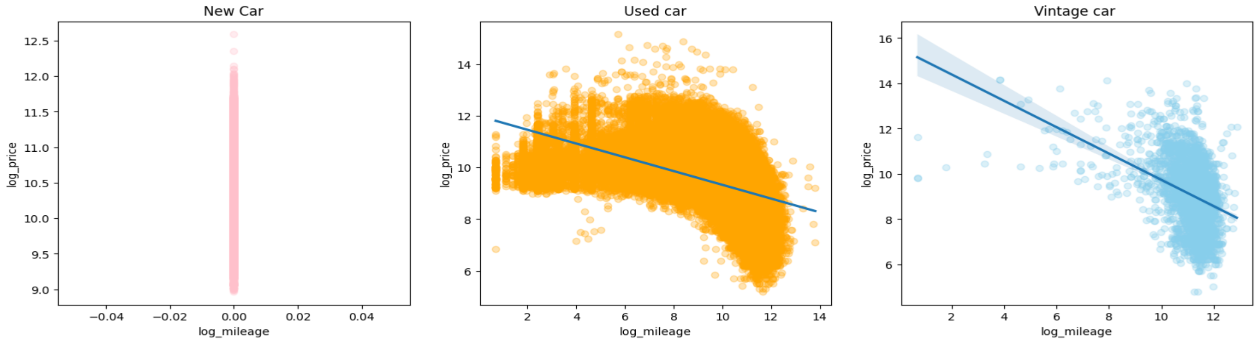
The series of plots above illustrates the preliminary exploration of our dataset. We observe the relationship between vehicle age, mileage, and their interaction with the log_price. The second-degree polynomial regression lines highlight non-linear trends:

Mileage and Price: The curve indicates that as mileage increases, the price of a vehicle decreases, but at a decelerating rate. This trend is particularly relevant for used cars, where high mileage significantly affects price. However, for vintage cars, which are often valued for their rarity rather than utility, high mileage may not substantially diminish value, necessitating a separate category in our analysis**.**

Age and Price: We notice an inflection point where the price starts to increase with age, likely reflecting the collector's value of vintage vehicles. New cars would not follow this trend; they typically depreciate as soon as they leave the dealership.

Mileage-Age Interaction: The interaction plot showcases the compounded effect of age and mileage. For new cars, even a small increase in mileage can lead to a sharp price drop. In contrast, the value of vintage cars may rise despite higher mileage, reflecting their collectability and potential as investment pieces. The shape of the curve for used cars likely falls between these extremes, with price affected by mileage but not as dramatically as with new cars.

New cars are less impacted by mileage, used cars show a standard depreciation curve, and vintage cars' value is not solely tied to their usage. This segregation allows for more accurate modelling and valuation by considering the unique factors that affect each category's pricing. This distinct pattern observed across the three categories highlight the need for a segregated analysis.



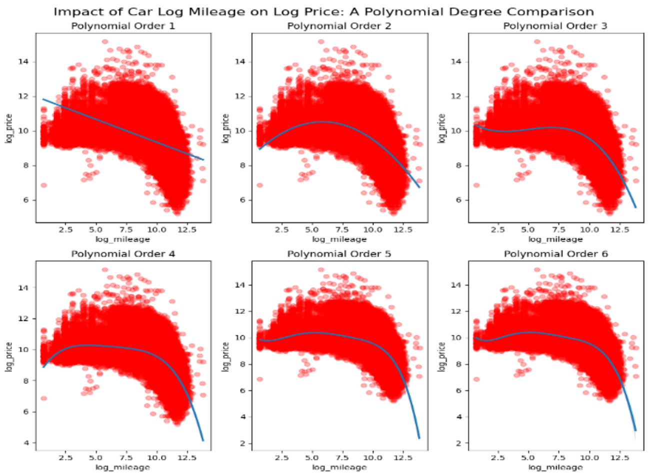
Given that new cars have negligible mileage and are current-year models (2021), introducing polynomial interactions to capture non-linear effects between age and mileage is unnecessary. These factors have zero variance and therefore zero impact on the price, making polynomial interaction unnecessary for this category.

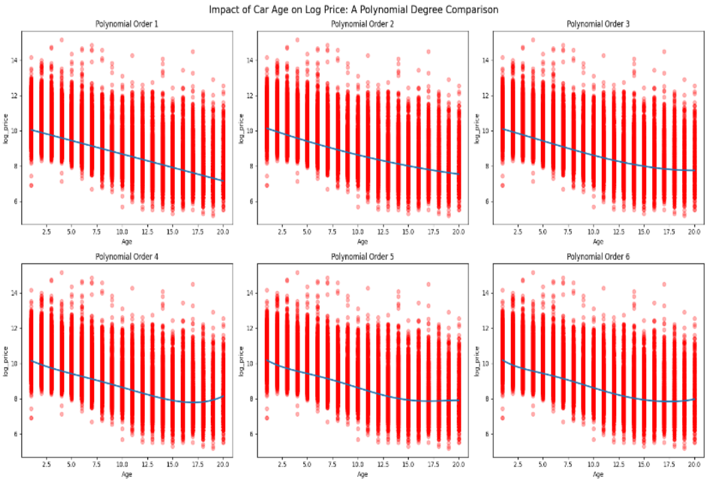
Used Car Polynomial Interaction: The set of graphs presents how used car prices are influenced by age and mileage, examined through polynomial regression models with degrees ranging from 1 to 6.

Polynomial Order 1: A linear relationship showcases the initial decline in price with increased age or mileage. This model is simple but may not capture more nuanced trends.

Polynomial Order 2 to 3: These graphs show that the price drop is sharper for newer cars and slows down for older ones. This likely means that buyers pay a lot less for slightly used cars, but once cars get older, their prices don't fall as quickly anymore.

Polynomial Order 4 to 6: These models begin to fit unusual patterns, suggesting overfitting.

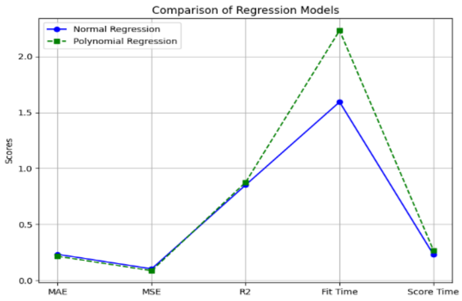


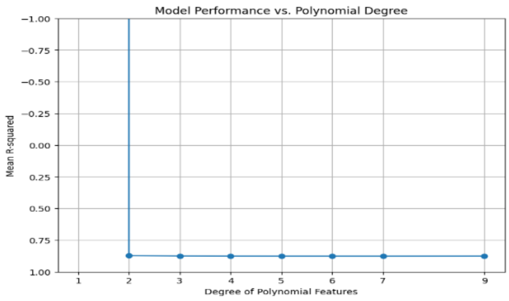


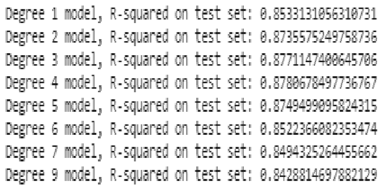
In the initial phase of model building, I applied both a standard linear regression (normal_reg) and polynomial regression (poly_reg) to understand how each would perform on the dataset. Using cross-validation, the models were compared on metrics like Mean Absolute Error (MAE), Mean Squared Error (MSE), and the coefficient of determination (R²)

In comparing linear and polynomial regression, the linear approach was steady across all our tests. However, when we added polynomial features, things got a bit more unpredictable. The model did better with some polynomial features, but when we added too many (a high degree), it started to do worse, likely overfitting the data.

From the Model Performance across different polynomial degree chart suggest that at degree 3, the model captures essential patterns effectively. However, beyond this degree, the model's performance begins to degrade, indicating an overfit to the noise in the data rather than the underlying trend.







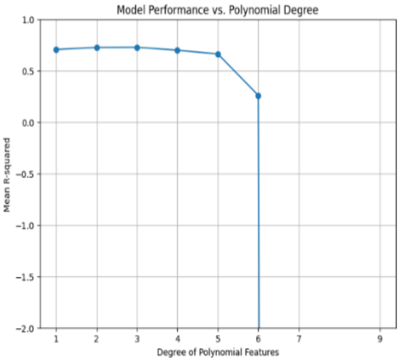
```
Degree 1 model, R-squared on test set: 0.8533131056310731
Degree 2 model, R-squared on test set: 0.8735575249758736
Degree 3 model, R-squared on test set: 0.8771147400645706
Degree 4 model, R-squared on test set: 0.8780678497736767
Degree 5 model, R-squared on test set: 0.8749499095824315
Degree 6 model, R-squared on test set: 0.8522366082353474
Degree 7 model, R-squared on test set: 0.8494325264455662
Degree 9 model, R-squared on test set: 0.8428814697882129
```

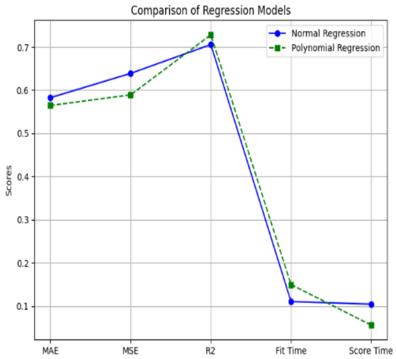
Vintage Car Polynomial Interaction: Similar to the Used car, Different degrees of polynomial interaction is evaluated for the feature Age and the log_mileage to the target value log_price. As expected, the pricing of vintage cars diverges from that of newer models, with unique factors like historical value and rarity coming into play.

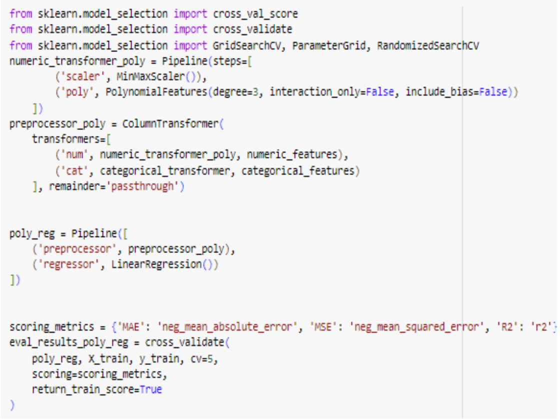
For vintage cars, a simple linear model doesn't quite capture their value. When we look at second or third-degree polynomials, we start seeing a pattern that fits better; mileage doesn't hit their price as hard as for other cars. For higher degrees, the model just starts to see patterns that aren't there which causes overfitting.

The below charts illustrating the model's R-squared performance across different polynomial degrees suggests optimal performance up to a cubic polynomial(3rd Degree). Beyond this point, model accuracy begins to decline, indicating potential overfitting as the model complexity increases.

In the comparison chart for normal versus polynomial regression, the inclusion of polynomial features seems to offer improved performance across various metrics. However, this comes with a trade-off in computation time.

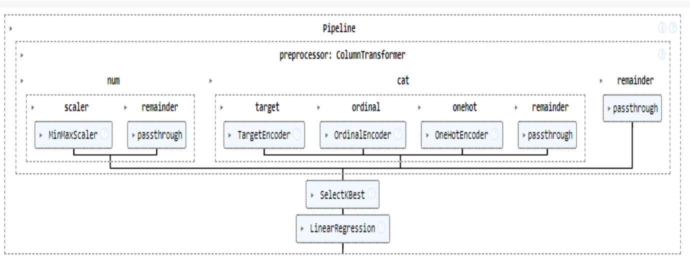


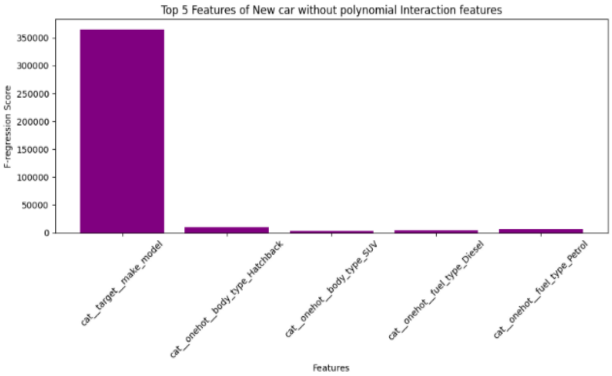


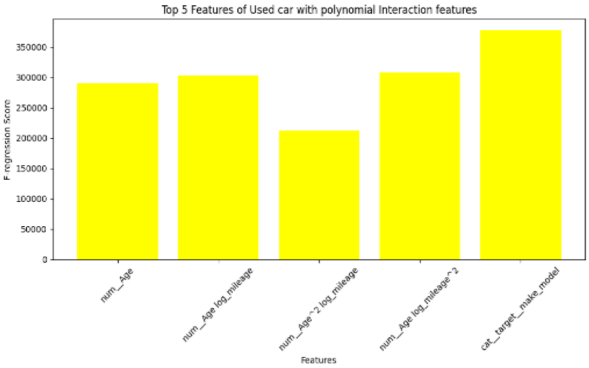


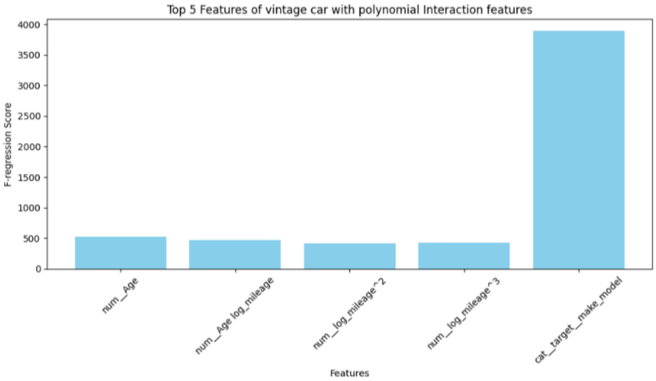
```
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import cross_validate
from sklearn.model_selection import GridSearchCV, ParameterGrid, RandomizedSearchCV
numeric_transformer_poly = Pipeline(steps=[
        ('scaler', MinMaxScaler()),
        ('poly', PolynomialFeatures(degree=3, interaction_only=False, include_bias=False))
    ])
preprocessor_poly = ColumnTransformer(
    transformers=[
        ('num', numeric_transformer_poly, numeric_features),
        ('cat', categorical_transformer, categorical_features)
    ], remainder='passthrough')


poly_reg = Pipeline([
    ('preprocessor', preprocessor_poly),
    ('regressor', LinearRegression())
])


scoring_metrics = {'MAE': 'neg_mean_absolute_error', 'MSE': 'neg_mean_squared_error', 'R2': 'r2'}
eval_results_poly_reg = cross_validate(
    poly_reg, X_train, y_train, cv=5,
    scoring=scoring_metrics,
    return_train_score=True
)
```

3.Feature Selection and Dimensionality reduction:









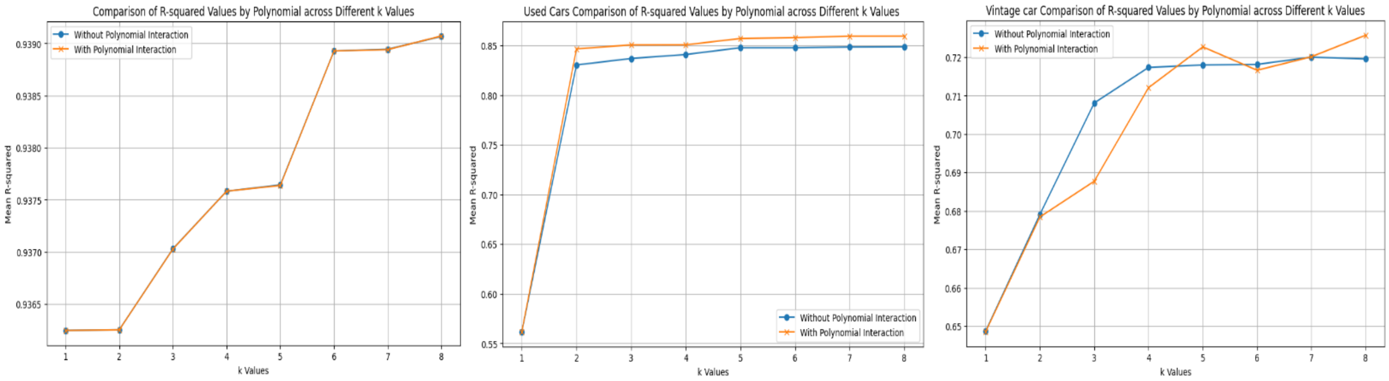
Feature selection and dimensionality reduction not only enhance model accuracy but also significantly reduce computational costs, making the modelling process more efficient, especially when dealing with large datasets for various vehicle types.

SelectKBest: After Preprocessing, we have applied SelectKBest to identify key features that have the most impact on predicting prices. This approach simplifies our model by focusing on the most informative data, improving speed and performance without compromising on the quality of prediction. The Above charts illustrates the top 5 feature of each vehicle type New, Used and Vintage cars.

The New cars bar chart prioritizes 'Make and Model' as the dominant feature influencing new car prices, far outstripping other variables like body type and fuel type. This indicates that brand recognition and model specifications are key factors in new car valuation, while the impact of age and mileage is negligible.

The Used car bar chart indicates that, both age and mileage significantly impact price, especially when combined as an interaction term. Additionally, the make and model remain crucial in determining a used car's worth.

For vintage cars, the chart indicates that 'Make and Model' considerably outweighs other factors such as age and mileage in terms of impact on price. This pattern aligns with the collector's market where a car's make and model, due to rarity and desirability, are often more important than usual wear-and-tear indicators.



The above graph displays a comparison of the R-squared values for Different car type with and without the addition of polynomial interaction features across a range of k values.

The chart for the new car shows that, including polynomial interactions does not significantly affect R-squared values. This suggests that simpler models are just as effective for predicting prices of new cars, where age and mileage vary minimally.

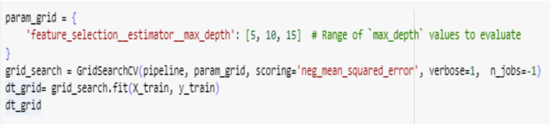
In the case of used cars, incorporating polynomial interactions improves model accuracy, as shown by higher R-squared values. This demonstrates the impact of age and mileage on pricing for used vehicles, which polynomial features can capture effectively.

The Vintage car chart shows how the model's accuracy levels off after including the best 4 or 5 features, suggesting additional features don't add predictive value for vintage car prices.

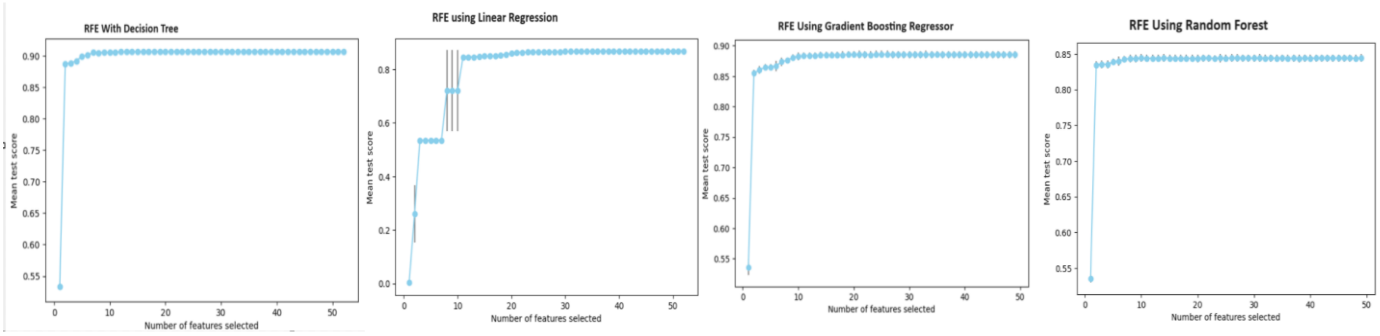
Recursive Feature Elimination: In our methodical approach to refine the predictive model, RFE was employed across various algorithms: Linear Regression, Decision Tree, Random Forest, and Gradient Boosting. The RFE was not just applied to the dataset as a whole but also segmented by car types—new, used, and vintage—aiming to capture the nuanced influences of features within each category.

1.Pipeline Creation: A pipeline was constructed for each model with RFE as a core component, ensuring systematic feature selection alongside model training. This automation facilitated the examination of feature importance across models and car types.

2. Model Evaluation: By using cross-validation within RFE, we were able to assess the models' performance stability, eliminating features until only the most influential remained. This was key in preventing model overfitting. And further, we have applied GridSearchCV for refining the optimal parameters.


```
param_grid = {
    'feature_selection__estimator__max_depth': [5, 10, 15]  # Range of `max_depth` values to evaluate
}
grid_search = GridSearchCV(pipeline, param_grid, scoring='neg_mean_squared_error', verbose=1,  n_jobs=-1)
dt_grid= grid_search.fit(X_train, y_train)
dt_grid
```

3.Comparative Analysis on Entire Dataset: The **Decision Tree model** achieved the highest R-squared score of 0.9078 using 21 features, indicating robust predictive capability with a moderate feature set. **Linear Regression** identified 40 influential features, scoring an R-squared of 0.8692, suggesting good fit but potential for overfitting. **Random Forest and Gradient Boosting,** limited to 10% data, showed promising efficiency, with R-squared scores of 0.8410 and 0.8830 respectively, and fewer features.



4. Comparative Analysis on New Car: Our feature selection analysis reveals distinct differences between methods. Using SelectKBest with Linear Regression, we pinpointed five crucial features that achieved a strong R-squared score of 0.9390, suggesting a highly effective yet simple model. In contrast, RFE identified 15 key features across various models. Except for Linear Regression, where RFE scored only 0.0467, all other models scored above 0.93. This discrepancy hints at the complexity of interactions in our dataset, which RFE captures well with more sophisticated models like Decision Trees and Gradient Boosting.

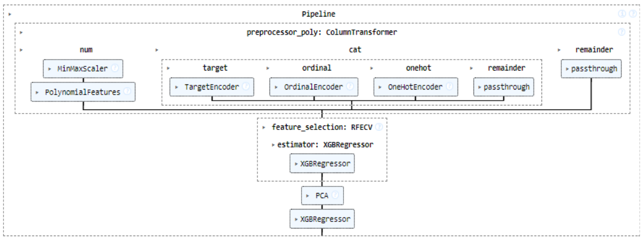
Our table summarizes these findings succinctly, providing a clear view of the feature counts and corresponding model performances.

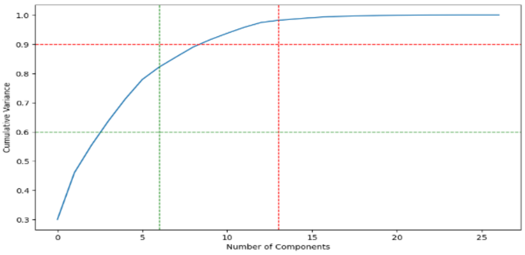
Model	Selected Features	Feature Count	Mean Test Score
Linear Regression SKBest	make_model', 'body_type_Hatchback', 'body_type_SUV', 'fuel_type_Diesel', 'fuel_type_Petrol'	5	0.9390 (R-squared)
Linear Regression RFE	'Age', 'log_mileage', 'Age^2', 'Age log_mileage', 'log_mileage^2', ...	15	0.0467 (R-squared)
Decision Tree RFE	'Age', 'log_mileage', 'Age^2', 'Age log_mileage', 'log_mileage^2', ...	15	0.9365 (R-squared)
Random Forest RFE	'Age', 'log_mileage', 'Age^2', 'Age log_mileage', 'log_mileage^2', ...	15	0.9365 (R-squared)
Gradient Boosting RFE	'Age', 'log_mileage', 'Age^2', 'Age log_mileage', 'log_mileage^2', ...	15	0.9345 (R-squared)

A similar was approach applied to used and vintage car categories, revealing patterns and feature relevance unique to each car type's market behaviour.

RFE effectively narrowed down key features across car types and models, with the Decision Tree standing out for its high R-squared score. For new cars, SelectKBest combined with Linear Regression excelled, identifying just five features with a strong predictive score. This suggests that simpler methods like SelectKBest can be competitive, especially in less complex scenarios. The comparative table consolidates these results, showcasing the strategic value of each method in feature selection.

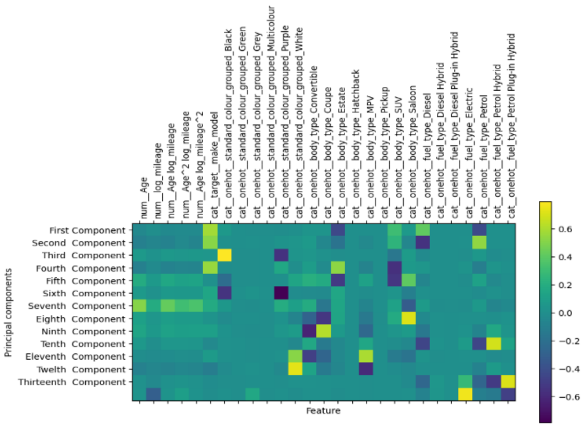
Principal Component Analysis: To Further Reduce the dimensionality of the selected features from the Automated feature selection technique, I have applied Principal Component analysis. PCA is the applied across all the best performing feature selection models to further reduce their dimensionality while preserving all the essential information.

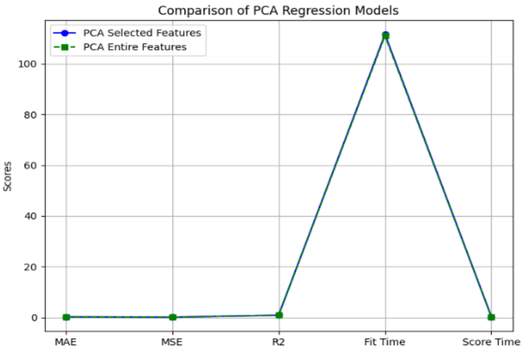




In the above example, PCA is applied after transforming both numerical and categorical columns using MinMax Scaler and other categorical encoding techniques. This is done following the selection of the best performing features using RFECV.

Upon analysis, we found that using fourteen principal components resulted in a cumulative variance of 0.98, indicating that these components capture 98% of the variability present in the original dataset. Based on this result, we chose to retain these fourteen principal components for further analysis.





The matrix above visualizes the contribution of each original feature to each principal component extracted through Principal Component Analysis (PCA). From the above matrix, we can see that the Features such as Age and Mileage, along with their interactions, and categorical features such as Vehicle Colours, Body Type, and Fuel Type, are significant contributors to the principal components.

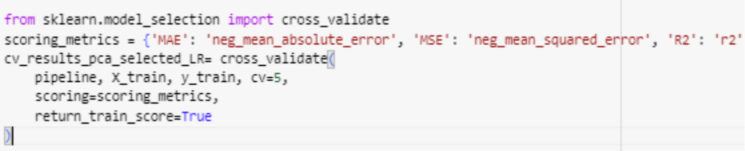
From the above comparison of regression models chart reveals that the performance using all the PCA features and the subset of 14 selected components is nearly identical across various metrics. This simplification leads to faster computation and improved model efficiency and also it reduces the chance of model overfitting. Similar steps were followed across all the models and similar results we found.

4.Model Building: In this section, we construct and evaluate regression models for predicting the car prices. We explore a variety of algorithms, each offering unique strengths and capabilities. The process involves fitting models, tuning hyperparameters, and selecting the best-performing configurations. Additionally, we consider ensemble techniques to further enhance predictive accuracy and robustness.

1.Linear Regression: This model assumes a linear relationship between the input features and the target variable, making it a suitable baseline for comparison with more complex models as our dataset contains features such as age and mileage that are highly correlated with the target variable price.

After Preprocessing the data and selecting the suitable features using Recursive feature elimination technique, dimensionality reduction was applied using Principal Component Analysis (PCA), selecting 30 components which capture the majority of the variance in the predictors while reducing computational load.

Cross Validation: To evaluate the linear regression model with PCA-reduced features, we conducted a thorough cross-validation analysis using MAE, MSE, and R² as metrics. This methodology ensures that our model's performance is not a result of overfitting but rather indicative of its ability to generalize to unseen data.



```
from sklearn.model_selection import cross_validate
scoring_metrics = {'MAE': 'neg_mean_absolute_error', 'MSE': 'neg_mean_squared_error', 'R2': 'r2'
cv_results_pca_selected_LR= cross_validate(
    pipeline, X_train, y_train, cv=5,
    scoring=scoring_metrics,
    return_train_score=True
)
```

Our model evaluation through 5-fold cross-validation has provided a detailed insight into the performance of our model.

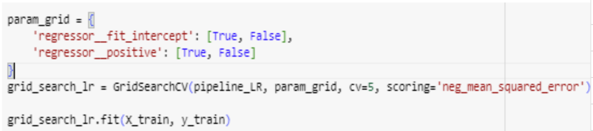
Mean Absolute Error (MAE): The models exhibited an MAE between -0.235 and -0.279 on the test sets across different folds, indicating a moderate prediction error in terms of price estimation.

Mean Squared Error (MSE): MSE values ranged from -0.108 to -0.155, showing the squared average of the errors, which points to some significant outliers affecting model performance.

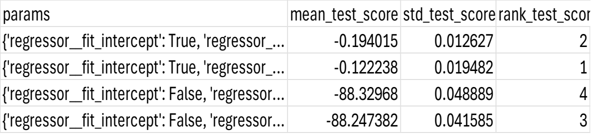
R-Squared (R²): The R² values varied from 0.789 to 0.852 on test sets, suggesting a good fit of the model to the data, although the lower end of this range may indicate potential underfitting in some folds. Hence, we increased the number of PCA components to 34 which resulted in the stable R² of 0.84 to 0.85 Across both the training and the test set.

Grid Search:

Grid search was applied to find the optimal parameters, The best-performing model configurations were those with the regressor__fit_intercept set to True and regressor__positive set to False, achieving the highest mean test score of -0.122 in MSE**.**



```
param_grid = {
    'regressor__fit_intercept': [True, False],
    'regressor__positive': [True, False]
}
grid_search_lr = GridSearchCV(pipeline_LR, param_grid, cv=5, scoring='neg_mean_squared_error')

grid_search_lr.fit(X_train, y_train)
```

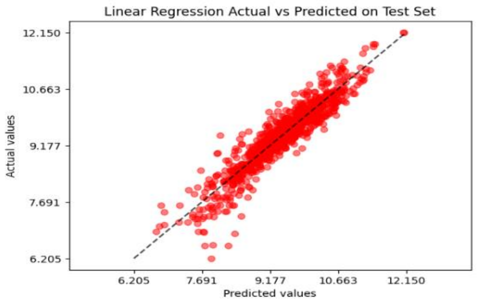


params	mean_test_score	std_test_score	rank_test_scor
{'regressor__fit_intercept': True, 'regressor_...	-0.194015	0.012627	2
{'regressor__fit_intercept': True, 'regressor_...	-0.122238	0.019482	1
{'regressor__fit_intercept': False, 'regressor...	-88.32968	0.048889	4
{'regressor__fit_intercept': False, 'regressor...	-88.247382	0.041585	3

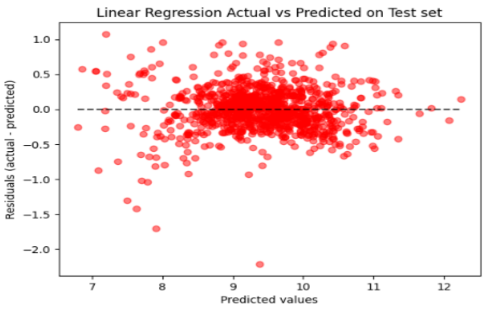
From our grid search, the optimal set of parameters yielded an improvement in the generalization of the model on unseen data, with the lowest standard deviation in test scores among the parameter sets tested.

*The Below scatter plot demonstrates the linear regression model's performance on the test set, where a closer alignment of points to the dashed line indicates higher prediction accuracy. The concentration of data points around this line suggests the model predicts well for most cases, with some scatter due to occasional outliers or model variance.

*The residuals plot assesses prediction errors, with most residuals clustering near zero, indicating generally accurate predictions. The distribution lacks any systematic pattern, suggesting that the model errors are relatively random and not dependent on the predicted values, which is desirable in a good predictive model.



Linear Regression Actual vs Predicted on Test Set



Linear Regression Actual vs Predicted on Test set

2.Boosting Tree: We Integrated XG Boost into our pipeline considering its efficiency and effectiveness in the large dataset, this model not only addresses our need for accurate predictions but also ensures computational efficiency**.**

Preprocessing: Standardization was applied to align all features on a common scale, crucial for maintaining consistency across the data which significantly affects the performance of gradient boosting models.

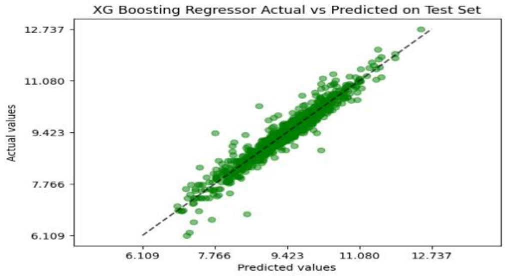
Feature Selection: Using RFECV, we narrowed down to the top 25 most significant features. This reduction in features is vital, as it decreases model complexity and enhances training speed without compromising on model accuracy.

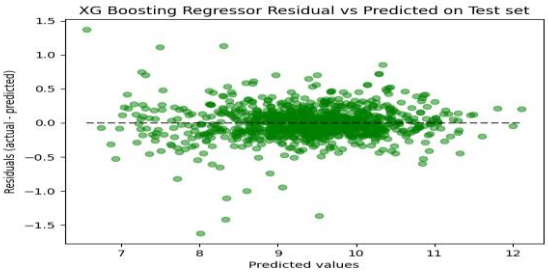
Dimensionality Reduction: We further condensed these features into 17 principal components via PCA. This reduction not only decreases the computational load but also helps in mitigating the curse of dimensionality, which is particularly problematic for tree-based models.

Cross-Validation Accuracy: Our XGBoost model demonstrated strong generalization capabilities with cross-validation scores of [0.92635552, 0.92612095, 0.91033764, 0.92743743, 0.92658348]. The consistency of these scores across different data subsets confirms the model's robustness.

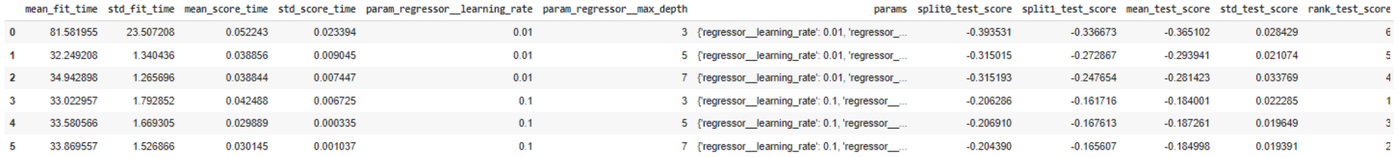
Test Performance: On unseen test data, the model achieved an R² score of 0.9292455359935805, indicating a high level of predictive accuracy, this closeness suggests that the model learns general patterns rather than memorizing the training data.

The Mean Squared Error (MSE) was 0.05141334325194698, and the Mean Absolute Error (MAE) was 0.15840937675358974, both of which underscore the model's precise predictions relative to the scale of target values.
The plots of actual vs. predicted values and the residuals confirmed the model's precision, with predictions closely aligned with true values and residuals evenly scattered around zero, indicating unbiased predictions.





Grid Search: The XGBoost model was then tuned and validated through a series of tests involving different parameter learning_rate (0.01 to 0.2), max_depth (3 to 10), n_estimators (100 to 300), and subsample (0.5 to 1.0) to find the optimal configuration. This was done to optimize the model for our specific dataset and objectives, ensuring a finely tuned balance between performance and overfitting.



	mean_fit_time	std_fit_time	mean_score_time	std_score_time	param_regressor__learning_rate	param_regressor__max_depth	params	split0_test_score	split1_test_score	mean_test_score	std_test_score	rank_test_score
0	81.581955	23.507208	0.052243	0.023394	0.01	3	{'regressor__learning_rate': 0.01, 'regressor_...	-0.393531	-0.336673	-0.365102	0.028429	6
1	32.249208	1.340436	0.038856	0.009045	0.01	5	{'regressor__learning_rate': 0.01, 'regressor_...	-0.315015	-0.272867	-0.293941	0.021074	5
2	34.942898	1.265696	0.038844	0.007447	0.01	7	{'regressor__learning_rate': 0.01, 'regressor_...	-0.315193	-0.247654	-0.281423	0.033769	4
3	33.022957	1.792852	0.042488	0.006725	0.1	3	{'regressor__learning_rate': 0.1, 'regressor__...	-0.206286	-0.161716	-0.184001	0.022285	1
4	33.580566	1.669305	0.029889	0.000335	0.1	5	{'regressor__learning_rate': 0.1, 'regressor__...	-0.206910	-0.167613	-0.187261	0.019649	3
5	33.869557	1.526866	0.030145	0.001037	0.1	7	{'regressor__learning_rate': 0.1, 'regressor__...	-0.204390	-0.165607	-0.184998	0.019391	2

3.Random Forest Regressor:

The Random Forest regressor plays a pivotal role in addressing complex regression tasks. Recognized for its robustness and ability to handle large datasets, this model forms the backbone of our predictive strategy.

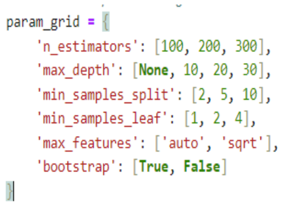
Pipeline Configuration:

Preprocessing and Feature Selection: Like the XG Boost Regressor, we start by standardizing the numerical data to ensure consistent scale and performance across the model. We employ RFECV for automated feature selection, effectively identifying and retaining only the most impactful features, streamlining our model without compromising its predictive power.

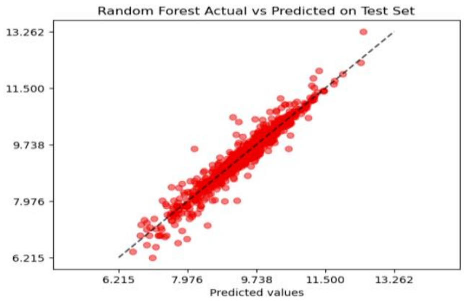
PCA Integration:

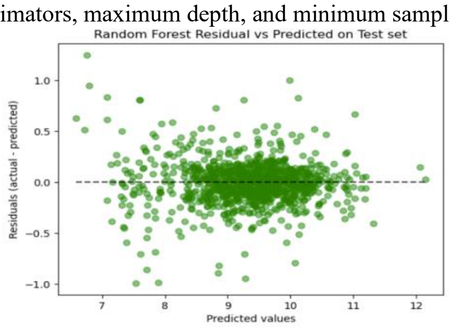
Dimensionality Reduction: By using principal component analysis (PCA), we condense features into 27 principal components, combating dimensionality's challenges and lowering computational needs. Capturing 99.54% of variance, it optimizes efficiency without sacrificing accuracy, ensuring peak model performance in speed and precision.

Model Performance and Overfitting: Random Forest regressor demonstrates impressive performance, with training and test scores of 0.9878 and 0.9220 respectively, there's a notable gap between the two, suggesting a potential for overfitting. To tackle this concern, we employ grid search for hyperparameter tuning, focusing on parameters like the number of estimators, maximum depth, and minimum samples per split.



```
param_grid = {
    'n_estimators': [100, 200, 300],
    'max_depth': [None, 10, 20, 30],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4],
    'max_features': ['auto', 'sqrt'],
    'bootstrap': [True, False]
}
```

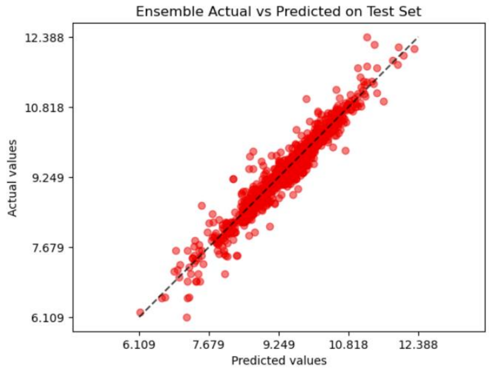


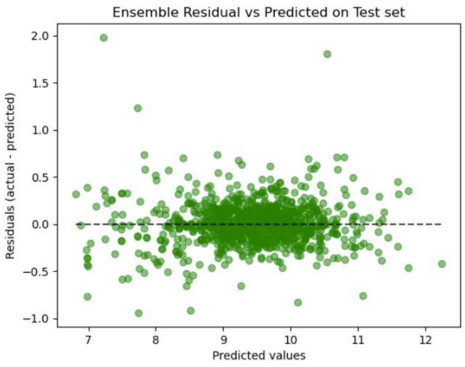


4.Stacking Ensemble:

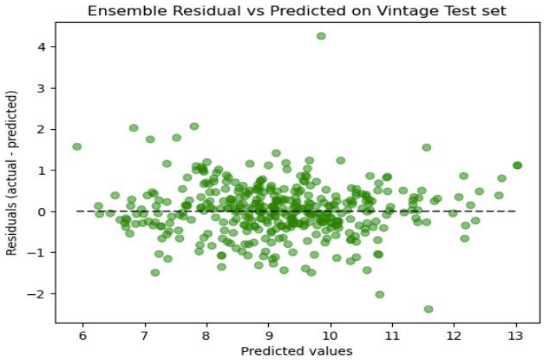
In our model-building process, we combined two different predictive models using a stacking ensemble to improve accuracy in car price predictions especially for vintage cars. We selected Linear Regression for its simplicity and effectiveness in capturing linear relationships. Linear Regression is good for understanding direct relationships between features like age or mileage and car prices and XGBoost was chosen for its powerful ability to handle complex and non-linear relationships in data.

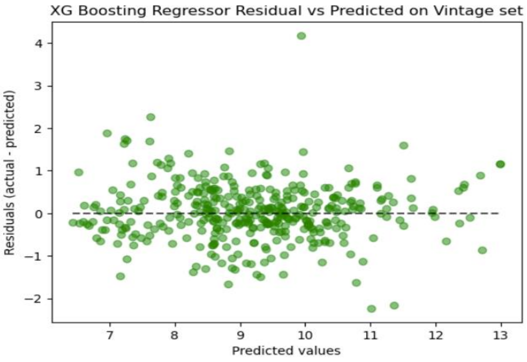
Final Estimator: To Combine the predictions from our base models, we employ a linear regression model as the final estimator. This meta-model combines the predictions from the base models using a linear combination of their outputs. By learning the optimal weights for these combinations, the final estimator produces a refined prediction that maximizes predictive accuracy.
Cross-Validation: The ensemble model performed consistently well across different data splits, as shown by the cross-validation scores: **[0.9287, 0.9235, 0.9275, 0.9255, 0.9285].** These scores tell us that the model does a good job at predicting car prices accurately across various parts of our data.





Specific Analysis on Vintage Cars: Vintage cars are a particularly challenging subset due to their unique attributes like historical significance, rarity, and varied condition. These factors complicate the predictive modelling as they do not always align with typical market dynamics that are easier to quantify.



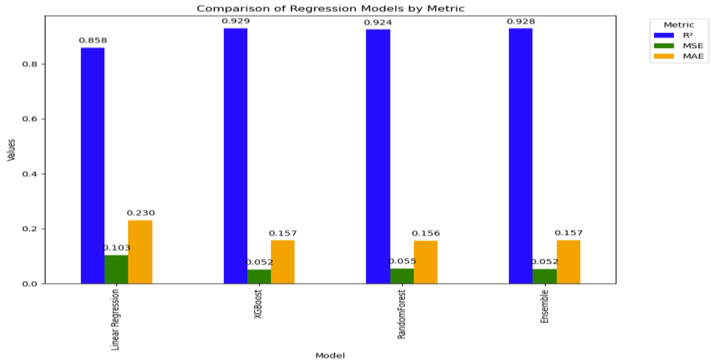


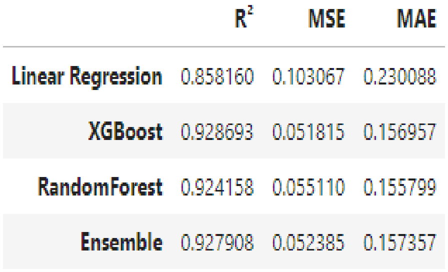
The residual plots for the ensemble model demonstrate a notable tightening of residuals compared to those from the base XGBoost model. This indicates a reduction in prediction errors, suggesting that the ensemble model is somewhat more precise in handling the complex factors influencing vintage car prices. However, the improvement is marginal emphasizing the difficulty of capturing the complex relationship of the vintage cars.

Model Enhancement: Adding the specified features—such as detailed car condition, authenticity of parts, heritage certificates, and market trends—will significantly improve the predictive model's accuracy for vintage cars.

5. Model Evaluation and Analysis:

Model Comparison: We initiated our analysis by comparing the four models using three key metrics: R², Mean Squared Error (MSE), and Mean Absolute Error (MAE). As illustrated in the comparative bar chart, the XGBoost model consistently outperformed the other models in terms of R², while showing competitive and often superior performance in minimizing MSE and MAE. This comprehensive evaluation highlighted XGBoost's effective balance between prediction accuracy and error minimization.



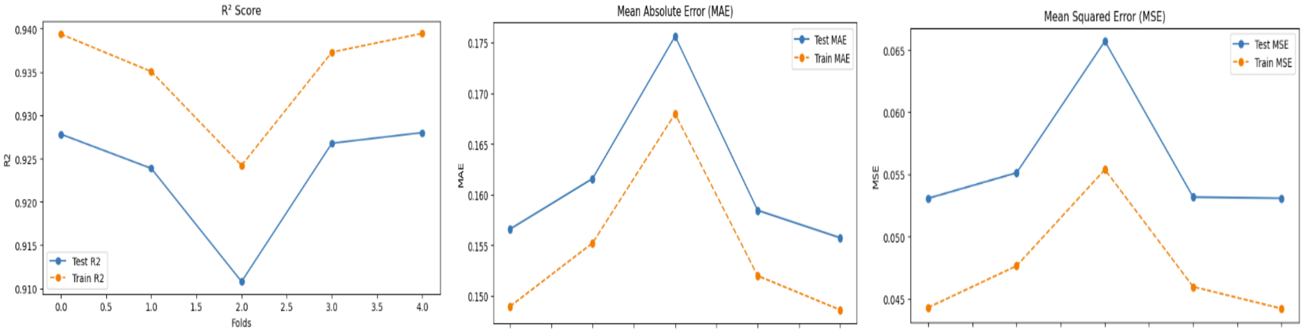


	R^2	MSE	MAE
Linear Regression	0.858160	0.103067	0.230088
XGBoost	0.928693	0.051815	0.156957
RandomForest	0.924158	0.055110	0.155799
Ensemble	0.927908	0.052385	0.157357

Cross-Validation Analysis: we employed a 5-fold cross-validation to assess the robustness and reliability of the XGBoost model. The cross-validation results demonstrated remarkable consistency in performance across all folds, reinforcing the model's stability and reliability.

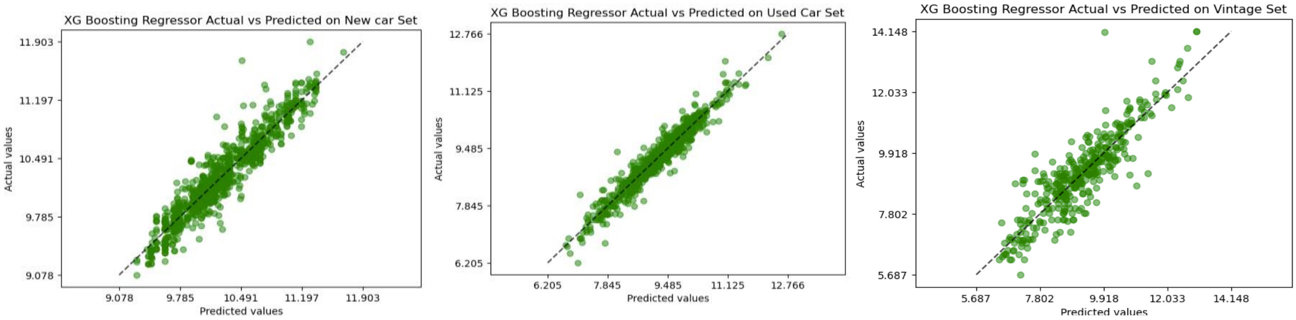
Mean Absolute Error (MAE): The test MAE scores showed slight variations across the folds, indicating a stable prediction error rate that aligns closely with the training data. **Mean Squared Error (MSE):** Similar to MAE, MSE scores remained consistent, which signifies effective handling of outliers and variance in the dataset.

R² Score: The R² values were predominantly above 0.91, suggesting a strong explanatory power and predictive accuracy of the model.



The line plots for each metric across the folds vividly depict the performance dynamics, with both training and testing scores demonstrating minimal overfitting. The consistent performance of XGBoost across multiple metrics and its computational efficiency were decisive factors in its selection.

True vs Predicted Analysis on subset of the car:



In our assessment of the XGBoost model across different car categories—new, used, and vintage—the model exhibited varying degrees of accuracy. For new cars, the model achieved an R² score of 0.9025, indicating a high level of predictability and a strong fit to the data. This high performance suggests that the model captures the majority of the variance in new car prices effectively. Used cars followed closely with an R² score of 0.93, showing excellent model performance in this broader category as well. However, the model's performance on the vintage car subset presented a lower R² score of 0.83, reflecting challenges in capturing the more nuanced factors influencing prices in this segment.

True vs Predicted plots for each category are included to visually demonstrate the model's performance, highlighting areas where the model predictions align closely with actual values and where discrepancies occur.

Recommendations:

For new cars, where simpler relationships in pricing dynamics exist, switching to a Linear Regression model using only the top five features identified by SelectKBest may be more efficient and equally effective. This shift promises reduced complexity and improved model interpretability.

For vintage cars, where predictive performance lags, incorporating additional features that capture unique attributes such as rarity, historical value, and maintenance history could enhance accuracy.

5.3 Global and Local Explanations with SHAP

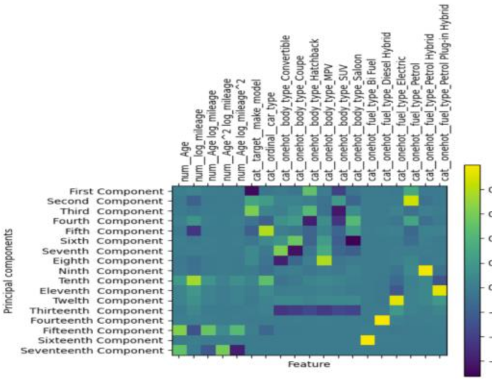
Key Findings from SHAP Analysis:

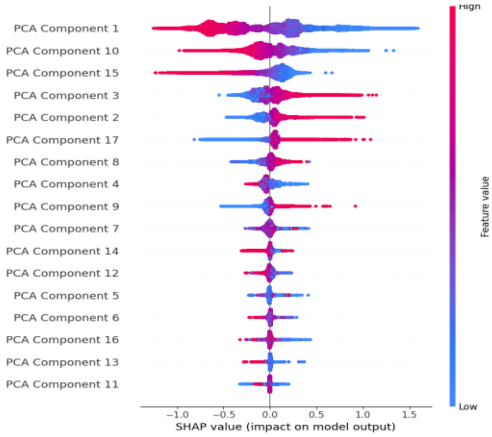
PCA Component 1: Dominantly impacted by make_model and body_type_Hatchback. This indicates that specific car makes and models either significantly increase (high positive SHAP values) or decrease (negative SHAP values) the predicted car prices. Typically, luxury or rare models are likely to elevate the predicted values, whereas more common models may decrease them.

PCA Component 10: Strongly influenced by mileage. Higher mileage consistently correlates with lower car values, reflecting the typical depreciation due to wear and use.

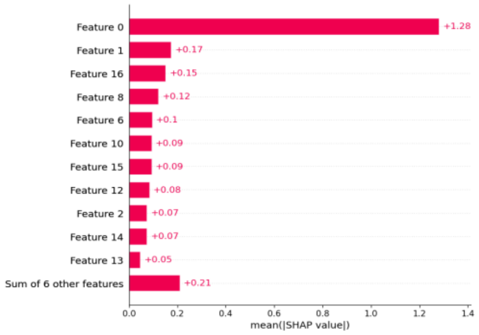
PCA Component 15: This component is primarily influenced by age and mileage, highlighting that older cars with higher mileage tend to have lower predicted values. This is intuitive as wear and age are critical depreciating factors in vehicle valuation.

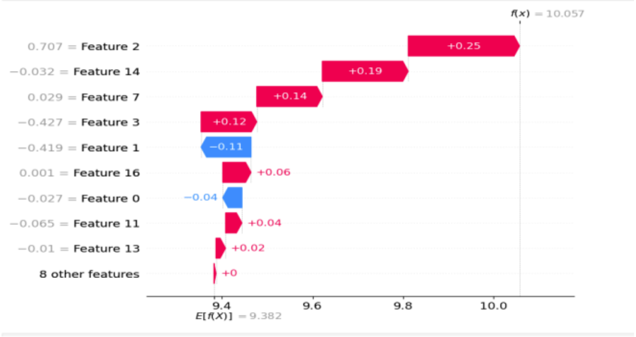
PCA Component 3: Affected by make_model and specific body types like SUVs and Hatchbacks. Certain premium models and body types enhance the vehicle's value, while others might reduce it, depending on their market perception and consumer preference.





Impact of Car Make and Model on Vintage Car Prices:

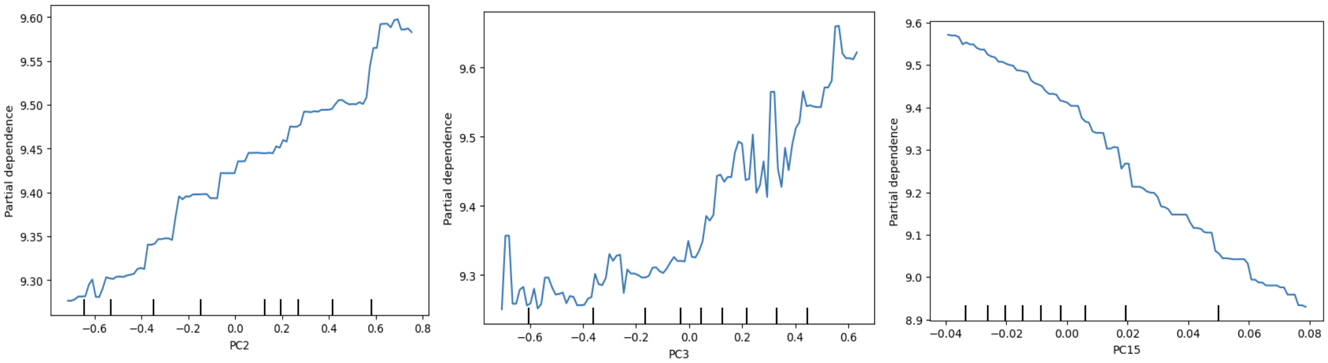




In our analysis of vintage car prices using machine learning, SHAP values revealed a notable influence of the make_model feature within PCA Component 1(Feature 0 in chart). Distinct car makes and models either significantly boost or reduce the predicted values, reflecting their market desirability and historical worth. To further refine our prediction models and capture the full spectrum of factors influencing vintage car prices, additional features such as car age, restoration level, originality of parts, and historical documentation might be integrated. These features could provide deeper insights into a car's condition and authenticity, factors that are likely to impact its valuation significantly.

5.4 Partial Dependency Plots:

To further explore these insights, Partial Dependence Plots were generated for the top five components identified by SHAP as having the greatest impact on the model. These PDPs illustrate the specific relationships between these principal components and the predicted outcomes, providing a visual representation of the effects identified through SHAP.



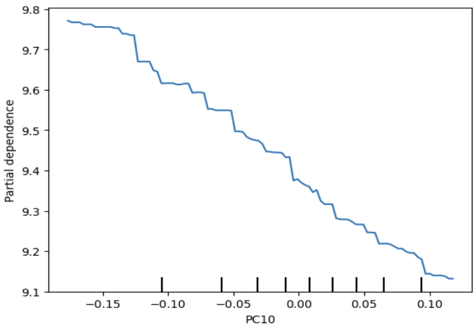
PC2 primarily reflects the impact of the vehicle's fuel type, particularly petrol, and to a lesser extent, the make and model. The Partial Dependence Plot for PC2 shows an increasing trend, indicating that vehicles with petrol as their fuel type tend to have higher predicted prices.

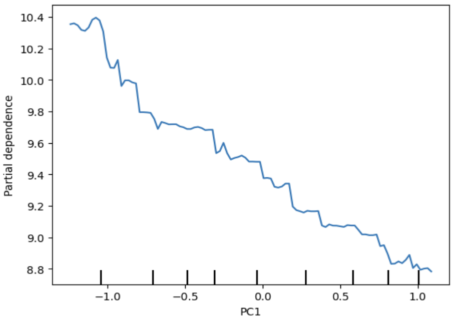
PC3, which integrates attributes from premium models and body types such as SUVs, positively influences the vehicle prices, underscoring the market preference for these vehicles.

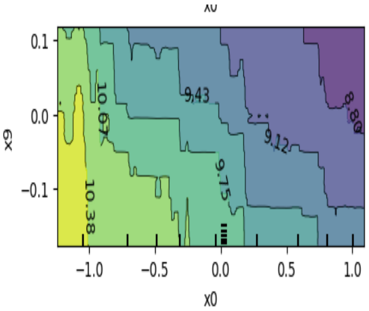
PC15 decline in relation to increasing age and mileage aligns with expected depreciation patterns.

PC10 Analysis via Partial Dependence Plots reveals that higher vehicle mileage, which predominantly defines PC10, correlates negatively with vehicle prices, confirming that increased mileage leads to lower valuations.

PC1, which strongly correlates with common car makes and hatchback body types, shows a clear negative impact on car prices, suggesting that commonality and non-luxury vehicle types are associated with lower values.


Partial dependence
9.8
9.7
9.6
9.5
9.4
9.3
9.2
9.1
−0.15
−0.10
−0.05
0.00
0.05
0.10
PC10


Partial dependence
10.4
10.2
10.0
9.8
9.6
9.4
9.2
9.0
8.8
−1.0
−0.5
0.0
0.5
1.0
PC1


10.67
9.43
9.12
8.80
9.75
10.38
x9
0.1
0.0
−0.1
−1.0
−0.5
0.0
0.5
1.0
x0