



Faculty of Engineering and Technology
Department of Electrical and Computer Engineering
Machine Learning and Data Science - ENCS5341

Assignment 2

Prepared by:

Ro'a Nafi 1201959

Zainab Jaradat 1201766

Instructor:

Dr. Ismail Khater

Date: November 27, 2024

Table of Content

1. Introduction.....	3
2. Dataset Description and Preprocessing.....	3
3. Regression Models and Performance on Validation Set.....	4
4. Feature Selection Using Forward Selection.....	6
5. Regularization Results.....	8
6. Hyperparameter Tuning with Grid Search.....	9
7. Final Evaluation of Test Set.....	12
8. Regression Model for a New Target.....	13
9. Visualizations.....	15
❖ Feature importance.....	15
❖ Error distribution.....	15
❖ Model predictions vs. actual values.....	16
❖ Residuals vs. Fitted Values.....	17
❖ Performance Comparison Across Linear Models.....	17
10. Conclusion.....	19

1. Introduction

This project aims to predict car prices using various regression techniques, including Linear Regression, LASSO, and Ridge Regression, on a dataset containing detailed car specifications. By applying these models, the study seeks to identify the most accurate approach for forecasting prices, focusing on the effectiveness of different regularization methods to mitigate overfitting.

The dataset, sourced from an automotive database, includes data on engine capacity, horsepower, and other relevant attributes of various car models. It has been preprocessed to address missing values, encode categorical features, and normalize numerical values, ensuring it is primed for regression analysis.

2. Dataset Description and Preprocessing

The dataset we used in this project contains details about cars and is designed to help predict their prices. It includes approximately 6,750 rows and 9 columns of data, each row representing a different car. The columns provide various details like car make, model, year, engine size, and more, with the car price being our target for predictions.

For preprocessing, we first loaded the data using the Python library pandas, which is great for handling and cleaning data. We encountered some issues like non-numeric values in columns that should be numeric, such as 'cylinder' and 'horse_power'. To clean this, we used the pandas function `to_numeric` with the setting `errors='coerce'`, which changes the non-numeric entries to NaN (a Python value indicating missing data). We also cleaned the 'seats' column where some entries had text like "Seater" or other unrelated information. We extracted only the numeric values using regular expressions and set any unlikely values, such as a number greater than 9, to NaN. For columns with missing values, we filled these in with the most common value in each column to keep our data consistent.

The prices were listed in different currencies, so we used regular expressions to separate the currency from the numeric value and converted all prices into USD for uniformity, applying appropriate exchange rates. After calculating the skewness of the data and observing a significant positive skew, we chose to fill any rows without clear price information with the median price to keep as much data as possible and to provide a robust central value that is not affected much by outliers. Finally, we standardize the data to have a standard deviation of one using label encoding, which helps in making the model's training more effective. We confirmed the standardization was correct by checking that the standard deviations are close to one, ensuring the data is properly prepared for machine learning models.

```
Standard Deviations of each column:
engine_capacity    1.000079
cylinder           1.000079
horse_power        1.000079
top_speed          1.000079
seats              1.000079
price              1.000079
brand_encoded      1.000079
country_encoded    1.000079
```

Column standard deviations results

After cleaning and standardizing, we split the dataset into three parts: 60% for training, 20% for validation, and 20% for testing, preparing it for the next steps in our analysis.

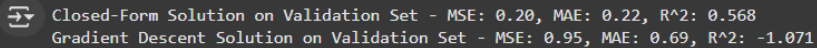
3. Regression Models and Performance on Validation Set

Linear Regression assumes a linear relationship between the features and the target variable. It is simple and interpretable but may not capture complex patterns in the data. LASSO (L1 Regularization) adds a penalty to the absolute values of the coefficients, which can shrink some coefficients to zero. This helps in feature selection and reducing model complexity. Ridge Regression (L2 Regularization) adds a penalty to the square of the coefficients, which helps control overfitting by shrinking the coefficients but not necessarily to zero.

This figure shows that Linear Regression achieved an MSE of 0.20, an MAE of 0.22, and an R^2 of 0.568, indicating a moderate fit to the data. LASSO Regression, which includes L1 regularization to help in feature selection by potentially reducing some coefficients to zero, had improved performance with an MSE of 0.18, an MAE of 0.23, and an R^2 of 0.611, even after optimizing the regularization strength ($\alpha = 0.0001$). Ridge Regression, applying L2 regularization to reduce overfitting by shrinking coefficients, also showed an improvement over Linear Regression with an MSE of 0.19, an MAE of 0.22, and an R^2 of 0.593 ($\alpha = 0.1$). These outcomes suggest that while regularization helps in managing model complexity and overfitting, each model varies in its effectiveness in capturing the more nuanced relationships in the dataset.

```
➡ Linear Regression - MSE: 0.20, MAE: 0.22, R^2: 0.568
LASSO Regression with Grid Search - MSE: 0.18, MAE: 0.23, R^2: 0.611
Best alpha for LASSO: {'alpha': 0.0001}
Ridge Regression with Grid Search - MSE: 0.19, MAE: 0.22, R^2: 0.593
Best alpha for Ridge: {'alpha': 0.1}
```

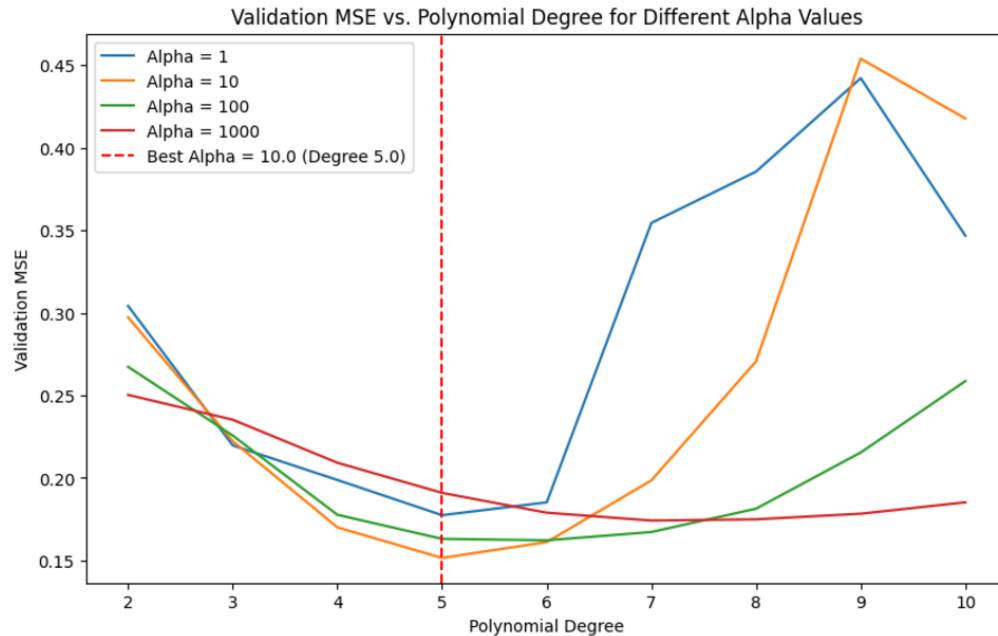
The Closed-Form Solution calculates the exact solution for linear regression by solving the normal equations. This method provides the optimal coefficients without iterative optimization. On the other hand, Gradient Descent is an iterative optimization method that minimizes the cost function by updating the coefficients in the direction of the negative gradient. This approach is particularly useful for large datasets where the closed-form solution may be computationally expensive.



Closed-Form Solution on Validation Set - MSE: 0.20, MAE: 0.22, R^2 : 0.568
Gradient Descent Solution on Validation Set - MSE: 0.95, MAE: 0.69, R^2 : -1.071

From the figure above, we observed a clear contrast between the predictive performances of the Closed-Form Solution and the Gradient Descent method. The Closed-Form Solution demonstrated a moderate fit with an R^2 of 0.568, suggesting it can reliably explain over half of the variance in car prices. On the other hand, the Gradient Descent method resulted in a negative R^2 of -1.071, indicating a model that not only fails to fit the data but indeed performs worse than a simple mean-based prediction. This outcome could stem from suboptimal parameter settings or insufficient iterations needed to achieve convergence in Gradient Descent. Going forward, it would be beneficial to explore parameter tuning and possibly alternative optimization algorithms to enhance the model's accuracy and reliability.

In our analysis of polynomial regression to predict car prices, the models with lower degrees (2 and 3) showed good performance, with a degree of 3 particularly effective, achieving a Mean Squared Error (MSE) of about 0.22 and explaining roughly 48% of the price variability ($R^2 = 0.483$). However, as we increased the polynomial degree to 5 and beyond, the performance metrics began to worsen significantly, indicating that the models were overfitting, or fitting the training data too precisely at the expense of general accuracy.



The model with a degree of 5 emerged as the best choice, based on the validation set data. It reported the lowest MSE at approximately 0.15, this means that on average, the model's predictions are very close to the actual car prices, with only a small error margin. Additionally, the R^2 (R-squared) of 0.668, indicates that this model can explain about 66.8% of the variations in car prices, which is a good level of accuracy.

RBF Kernel Ridge Regression - MSE: 0.18, MAE: 0.18, R^2 : 0.613

Also, the RBF Kernel Ridge Regression model has performed quite well in predicting car prices. It achieved an MSE (Mean Squared Error) of 0.18 and an MAE (Mean Absolute Error) of 0.18, and the R^2 (R-squared) value is 0.613, indicating that this model can explain about 61.3% of the variations in car prices.

The model with a degree of 5 emerged as the best choice based on the validation set data, reporting the lowest MSE at approximately 0.15 and an R^2 of 0.668, making it highly effective at predicting car prices with considerable accuracy.

4. Feature Selection Using Forward Selection

We used forward selection for Polynomial Regression (degree 5) to find the most important features for predicting 'price'. The goal was to reduce the Mean Squared Error (MSE)

by adding features one by one and seeing how they improve the model. As shown in the figure below, We started with 'horse_power', which gave an MSE of 0.31. As we added more features, the MSE kept dropping, showing that these features improved the model's accuracy.

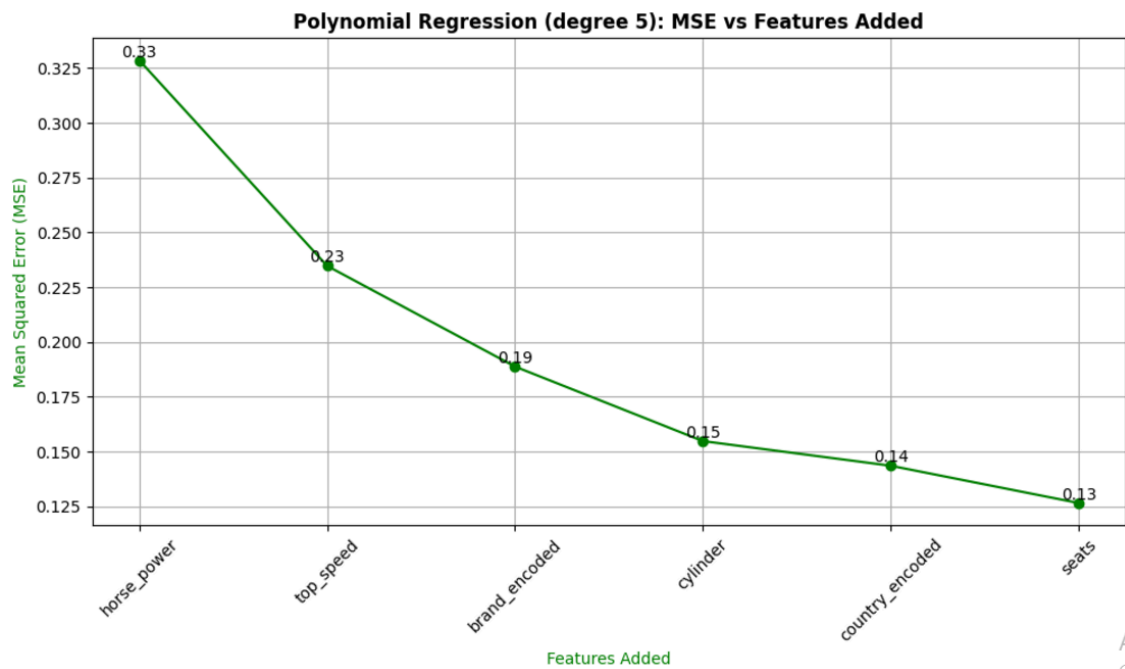
The MSE went down from 0.31 to 0.14 as we included 'seats', 'top_speed', 'cylinder', 'brand_encoded', and 'country_encoded'. The biggest drops in MSE happened with the first few features, with smaller improvements as we added the last ones. This means the initial features had the most impact on the model, while the later ones didn't help as much.

```
Added feature 'horse_power', MSE: 0.33
Added feature 'top_speed', MSE: 0.23
Added feature 'brand_encoded', MSE: 0.19
Added feature 'cylinder', MSE: 0.15
Added feature 'country_encoded', MSE: 0.14
Added feature 'seats', MSE: 0.13

Selected feature indices: [2, 3, 5, 1, 6, 4]
Selected feature names: ['horse_power', 'top_speed', 'brand_encoded', 'cylinder', 'country_encoded', 'seats']
```

Polynomial Regression (degree 5) with Forward Selection Result: Features Added with MSE

The final set of selected features included 'horse_power', 'seats', 'top_speed', 'cylinder', 'brand_encoded', and 'country_encoded', achieving the lowest MSE. The plot below shows this clearly, with a sharp drop in MSE when adding the first features, and then leveling off.



Polynomial Regression (degree 5) with Forward Selection: Features Added plot vs MSE

In conclusion, the forward selection process for Polynomial Regression (degree 5) helped us find the key features for predicting price. It showed that only a few features made a big difference in the model's accuracy, and adding more didn't help much after those key ones. This approach helped us build a good model that captures the main factors without overfitting.

5. Regularization Results

We used the regularization techniques LASSO and Ridge to make our regression models more accurate and stable. LASSO stands for Least Absolute Shrinkage and Selection Operator. This method helps by making some of the model's coefficients become zero. It removes the less important features from our model, which can make it simpler and easier to understand. On the other hand, Ridge adds a penalty that is based on the square of the coefficients. It doesn't reduce the coefficients to zero but makes them smaller, which helps prevent the model from fitting the training data too closely.

To find the best value for the regularization strength, λ , we used a method called Grid Search. This approach tests different values of λ to determine which one provides the best performance. For LASSO, the best λ we found was 0.0001, and for Ridge, it was 0.1. These values help control how much we penalize the model for having large coefficients, which is important for preventing overfitting and ensuring the model generalizes well.

We then evaluated how well these models performed on the validation set. The LASSO model with $\lambda=0.0001$ achieved a Mean Squared Error (MSE) of 0.18, a Mean Absolute Error (MAE) of 0.23, and an R^2 score of 0.611. The Ridge model with $\lambda=0.1$ had an MSE of 0.19, an MAE of 0.22, and an R^2 score of 0.593. These metrics indicate how close the model's predictions are to the actual values, with lower MSE and MAE reflecting better accuracy and a higher R^2 showing how well the model fits the data.

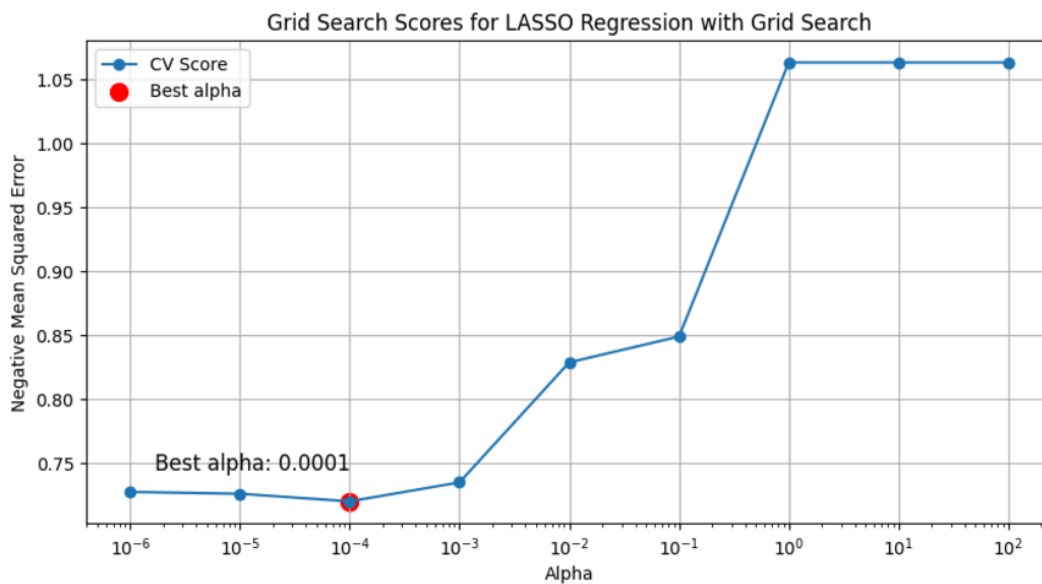
Overall, using LASSO and Ridge improved our models by preventing overfitting and enhancing their ability to predict unseen data. The λ values selected through Grid Search significantly contributed to optimizing the performance of both models, with LASSO performing slightly better than Ridge on our validation set.

```
➡ Linear Regression - MSE: 0.20, MAE: 0.22, R^2: 0.568  
   LASSO Regression with Grid Search - MSE: 0.18, MAE: 0.23, R^2: 0.611  
   Best alpha for LASSO: {'alpha': 0.0001}  
   Ridge Regression with Grid Search - MSE: 0.19, MAE: 0.22, R^2: 0.593  
   Best alpha for Ridge: {'alpha': 0.1}
```

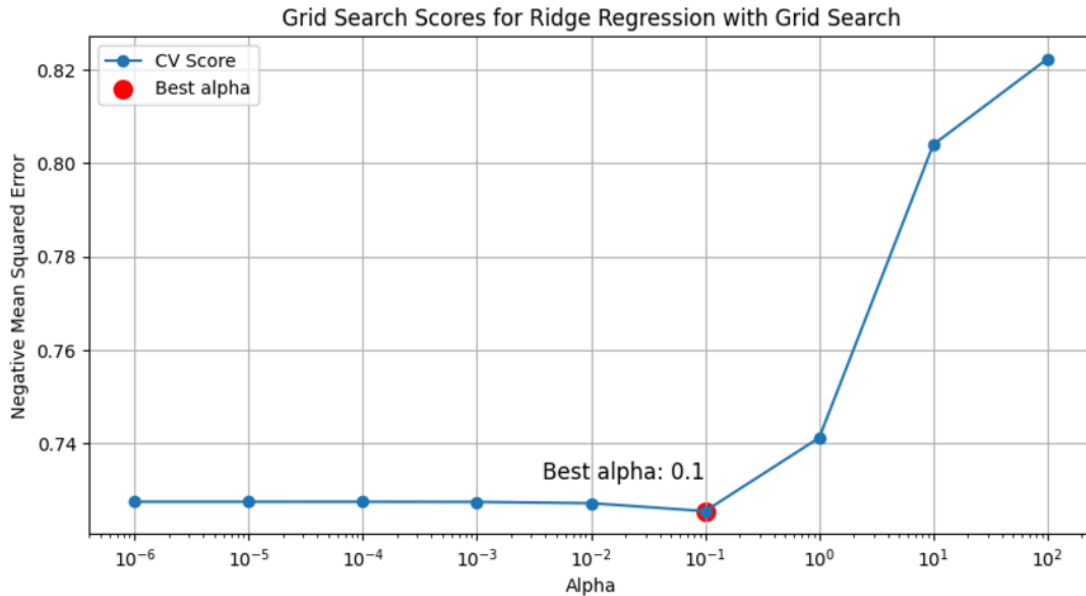

6. Hyperparameter Tuning with Grid Search

In this part, we focused on selecting the best model and tuning its hyperparameters using Grid Search. This method helps us find the best values for the model parameters by testing multiple combinations of parameter settings. We applied this to four regression techniques: LASSO, Ridge Regression, Polynomial Regression, and RBF Kernel Ridge Regression.

For LASSO regression, we used Grid Search to find the optimal value of the regularization parameter, alpha. The best alpha value was found to be 0.0001, which minimized the mean squared error. This indicates that the model performs best with this level of regularization, balancing the trade-off between bias and variance.



In Ridge Regression, we also used Grid Search to determine the best alpha value. The optimal alpha was 0.1, which provided the lowest mean squared error. This suggests that a moderate level of regularization is most effective for this dataset, helping to reduce overfitting and improve generalization.

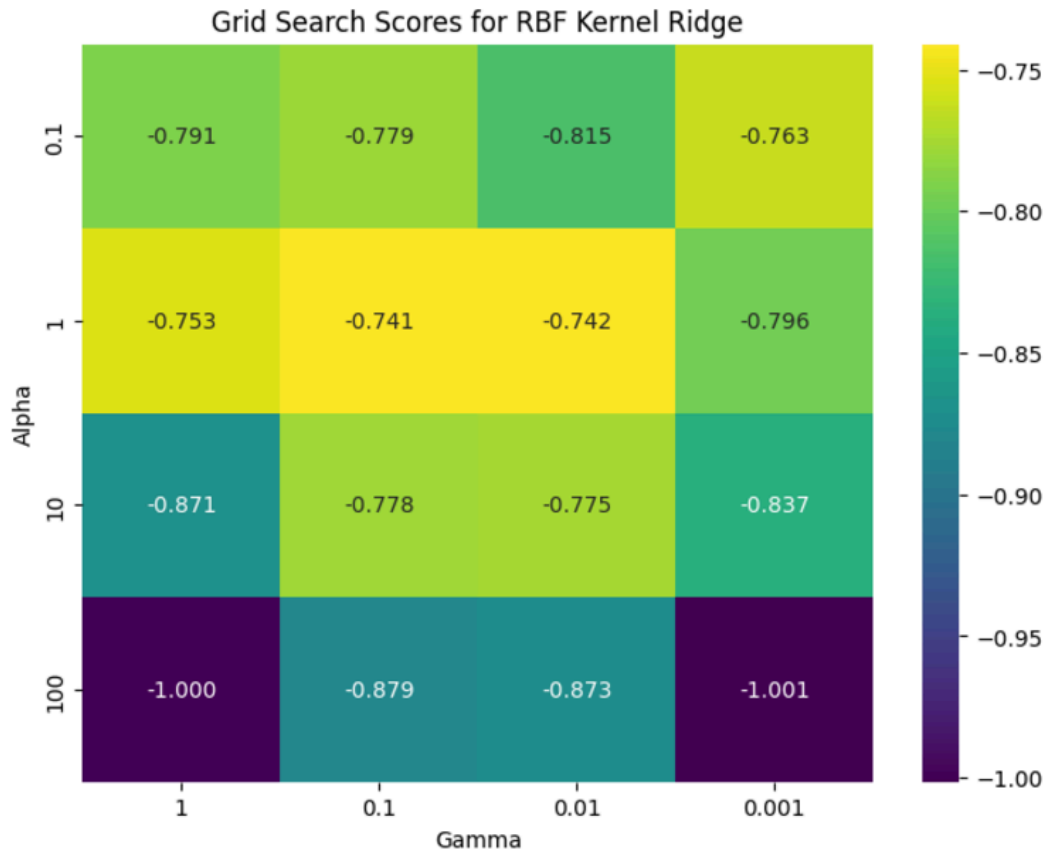


The optimal configuration for Polynomial Ridge Regression used a degree of 4 and an alpha of 1000. This balance effectively captured data complexity while preventing overfitting, resulting in a Test MSE of 0.6580 and R^2 of 0.5154. Although the model performs moderately well, further improvement might require exploring alternative features or regularization techniques.

Before the grid search, initially we suggested a degree of 5 as the optimal choice based on the validation MSE. However, after further analysis, we decided that a degree of 4 is more optimal. This is because it strikes a better balance between performance (Test MSE = 0.6580, Test R^2 = 0.5154) and generalization, avoiding overfitting observed with higher-degree polynomials.

```
Best Model Configuration:
Degree: 4.0, Alpha: 1000.0
Test MSE: 0.6580, Test R2: 0.5154
```

Finally, for the RBF Kernel Ridge Regression, we used Grid Search with Support Vector Machines (SVM) to find the best values for the hyperparameters C and γ . The optimal values were $C = 1$ and $\gamma = 0.1$. These values provided the best performance, indicating that a higher penalty for misclassification and a broader influence of each support vector is beneficial for this dataset.



Overall, the Grid Search method was very effective in tuning the hyperparameters for each model, ensuring that we achieved the best possible performance. By carefully selecting the optimal values, we were able to improve the accuracy and generalization of our regression models.

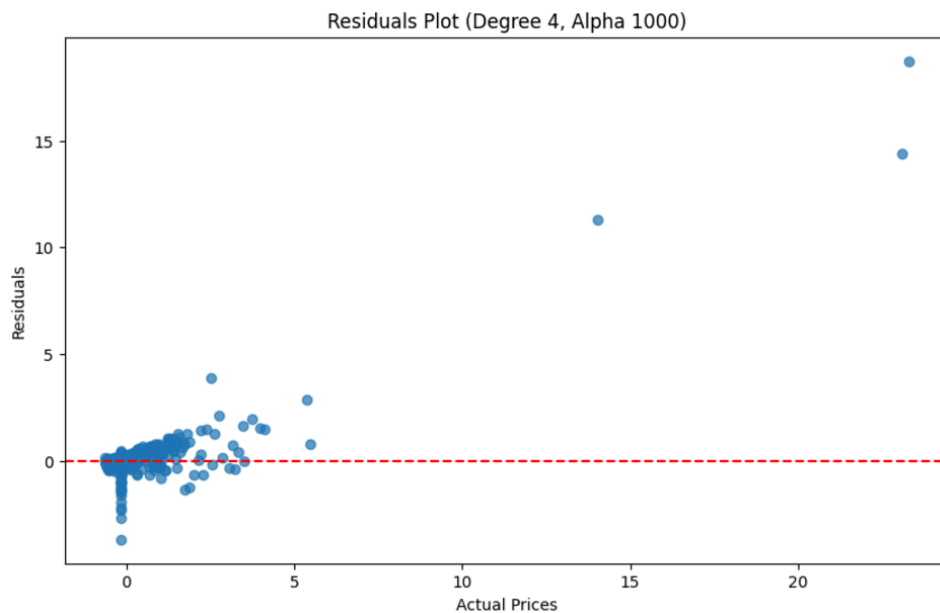
7. Final Evaluation of Test Set

After determining the best configuration (Polynomial Degree 4, Alpha 1000) from the validation process, the model was tested on unseen data to assess its generalization capability. The evaluation metrics on the test set include a Mean Squared Error (MSE) of 0.6828, a Mean Absolute Error (MAE) of 0.2400, and an R^2 score of 0.4971. These results indicate that the model captures nearly half of the variance in the target variable, showcasing moderate performance. The low MAE demonstrates that the average error magnitude in predictions is relatively small, making the model's outputs reasonably reliable for individual predictions.



```
Test Set Evaluation (Degree 4, Alpha 1000):  
Test MSE: 0.6828  
Test MAE: 0.2400  
Test R2: 0.4971
```

The residuals plot further validates the model's performance, as most residuals are clustered around the zero line, indicating minimal bias in predictions. However, a few extreme residuals suggest that the model struggles with certain data points, likely due to outliers or complex nonlinear patterns not adequately captured by the degree-4 polynomial. These outliers may significantly impact the MSE, inflating it beyond the errors observed for the majority of the test set.



The degree-4 polynomial model strikes a balance between complexity and overfitting, as evidenced by its better performance compared to higher-degree models. Regularization with Alpha 1000 prevents overfitting, ensuring the model generalizes well to unseen data. Despite these strengths, the relatively low R^2 score implies there is substantial variance in the data that remains unexplained, possibly due to missing features or noise in the dataset.

To improve performance, further exploration of feature engineering and data preprocessing might be necessary. Addressing outliers through robust regression methods or alternative loss functions could reduce the impact of extreme residuals. Additionally, experimenting with advanced algorithms like Gradient Boosting or Neural Networks may provide better flexibility in modeling the data's complex patterns.

In summary, the chosen model performs reasonably well on the test set, with low prediction errors and moderate generalization ability, though it leaves room for refinement and exploration of alternative approaches.

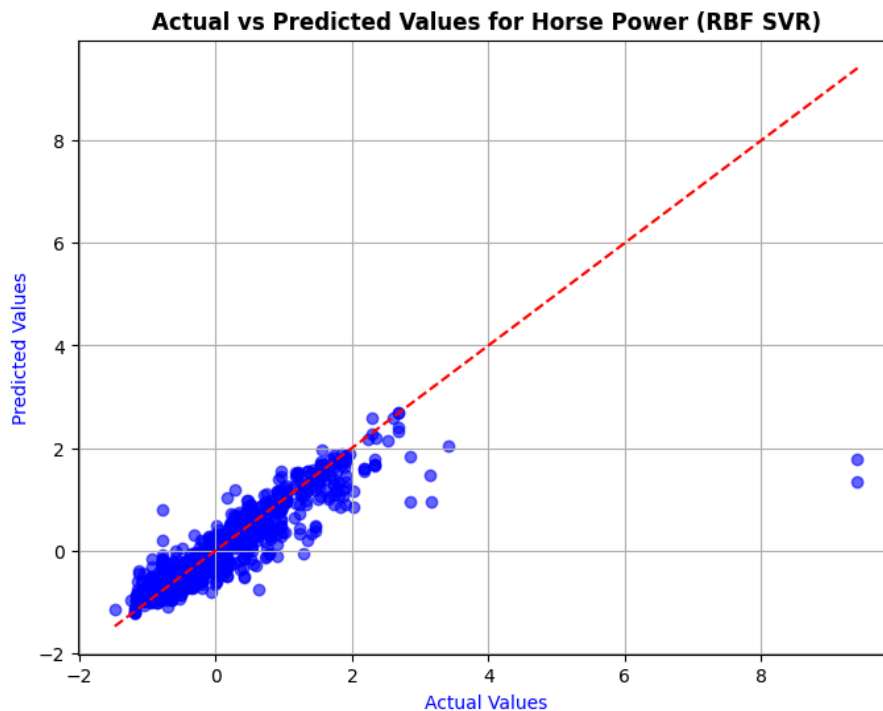
8. Regression Model for a New Target

We used Support Vector Regression (SVR) with an RBF kernel to predict horse power. We evaluated the model using MSE, MAE, and R^2 for the training, validation, and test datasets.

As shown in the figure below, For the training data, the model had a moderate error, with an MSE of 0.47, MAE of 0.22, and R^2 of 0.592, meaning it explained about 59.2% of the variance. On the validation data, the performance got much better, with an MSE of 0.09, MAE of 0.20, and R^2 of 0.861, showing a better fit. The model also did well on the test data, with an MSE of 0.19, MAE of 0.21, and R^2 of 0.778, indicating it worked well with new, unseen data.

```
SVR with RBF Kernel:  
Training Data - MSE: 0.47, MAE: 0.22, R^2: 0.592  
Validation Data - MSE: 0.09, MAE: 0.20, R^2: 0.861  
Test Data - MSE: 0.19, MAE: 0.21, R^2: 0.778
```

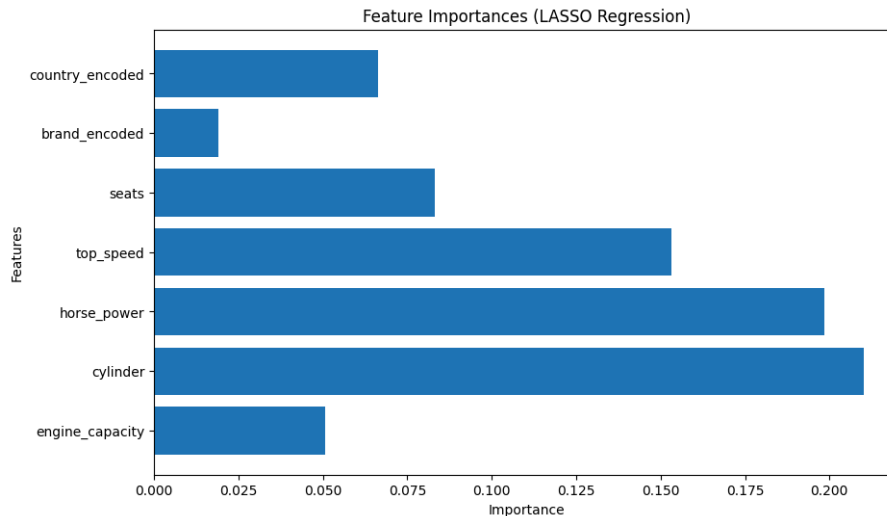
The scatter plot below of actual vs. predicted values shows most predictions aligning closely with the ideal red dashed line, indicating good model performance, although some outliers are visible.



In short, the SVR with RBF kernel model performed well, especially on the validation and test data, showing strong generalization. Further fine-tuning of the hyperparameters could make the model even better.

9. Visualizations

❖ Feature importance

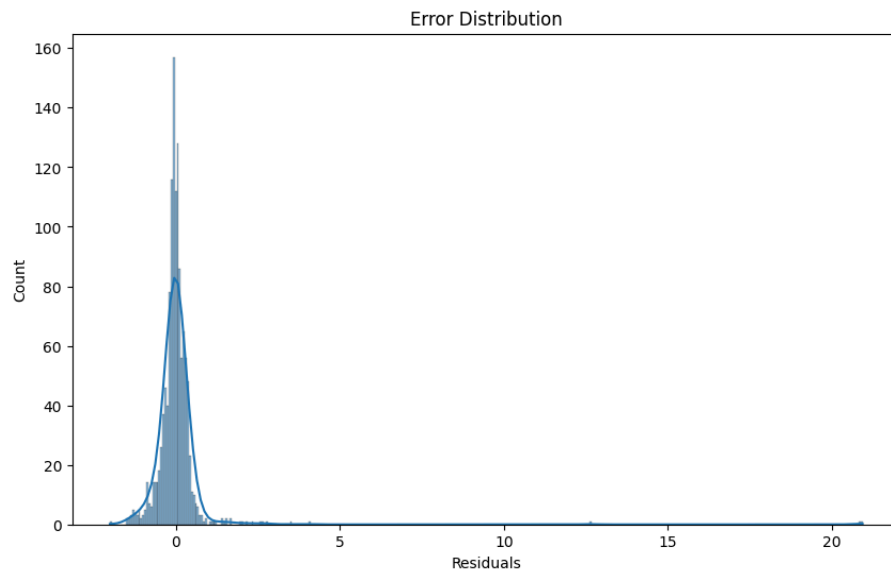


Feature importance Chart

The bar chart shows how important each feature is according to the LASSO regression model. The most important features for predicting car prices are 'cylinder', 'horse_power', and 'engine_capacity', with 'cylinder' being the most significant. Features like 'country_encoded' and 'brand_encoded' are less important but still matter.

❖ Error distribution.

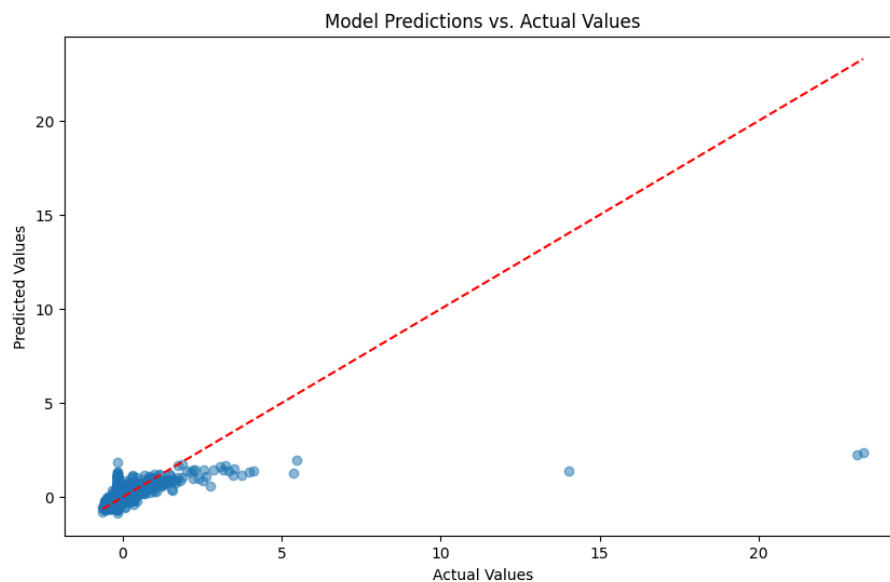
The histogram shows the differences between the predicted and actual values. The distribution is highly skewed, meaning it's uneven with more errors on one side. Most of the errors are close to 0, which means the model's predictions are mostly accurate. However, there are some bigger errors, suggesting that the model made some significant mistakes in a few cases.



Error distribution

❖ Model predictions vs. actual values.

The scatter plot compares the predicted values with the actual car prices. The red dashed line represents the ideal scenario where predicted values perfectly match actual values. The model's predictions mostly follow the diagonal line, but there is some deviation, indicating that the predictions are close but not perfect.

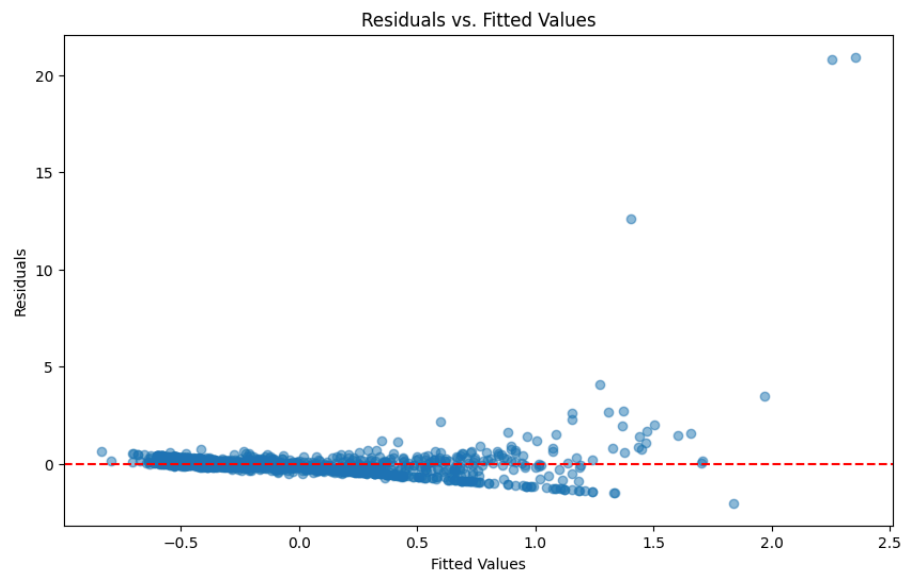


Model predictions vs. actual values

❖ Residuals vs. Fitted Values

The Residuals vs. Fitted Values plot helps us see how the residuals (the differences between predicted and actual values) are spread out compared to the predicted values. We look at this plot to check if the model is making random errors or if there's a pattern in the errors. Ideally, residuals should be scattered randomly around 0, with no clear pattern. If there is a pattern, it means the model might be missing some important information from the data.

In the plot, most of the points are scattered around the red dashed line at 0, which means the model doesn't have any major bias (it's not always predicting too high or too low). However, there are a few points that are far from the line, which shows the model made some bigger mistakes in those cases and couldn't predict those values accurately.

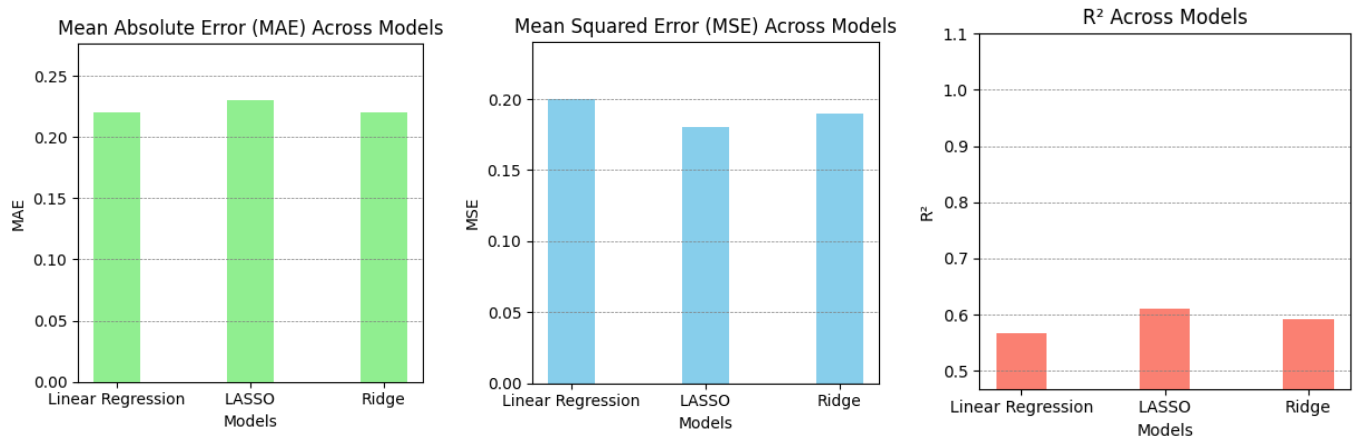


Residuals vs. Fitted Values plot

❖ Performance Comparison Across Linear Models

The bar charts compare three different ways to predict values: Linear Regression, LASSO, and Ridge. We used three methods to see how well these predictions match the actual values:

1. **Mean Absolute Error (MAE):** This measures the average difference between the predicted and actual values. All three methods (Linear Regression, LASSO, and Ridge) had very similar MAE values, meaning they were all equally good (or bad) at predicting the values.
2. **Mean Squared Error (MSE):** This is similar to MAE but gives more weight to larger errors. Again, all three methods performed similarly, with almost identical MSE values.
3. **R-squared (R^2):** This shows how well the model explains the data. The R^2 values were quite low for all three methods, meaning they didn't explain much of the variation in the data. This could be because the data is complex or the models need further improvement.



MSE, MAE, R^2 Comparison Across Linear Regression Models

10. Conclusion

This project focused on predicting car prices using different regression techniques. The best model we found was the degree-4 polynomial regression with Ridge regularization ($\text{Alpha} = 1000$). This model gave a good balance between complexity and overfitting. On the test data, it performed moderately well, with a Mean Squared Error (MSE) of 0.6828 and an R^2 score of 0.4971. These results mean that the model can explain about half of the variance in car prices. Forward feature selection also helped us choose important features like 'horse_power', 'seats', and 'top_speed', which improved the model's accuracy.

We also used regularization methods like LASSO and Ridge to make our models more stable and less likely to overfit. LASSO performed slightly better by selecting the most important features. Grid Search was very helpful in finding the best hyperparameters for each model. We also tested RBF Kernel Ridge Regression, which showed strong performance, proving that non-linear models can be very effective for this type of data.

For Future Work, we can try adding more features or creating new ones to better represent the data. This can help the model understand the relationships in the data more accurately. Second, outliers in the dataset caused some errors, so handling these outliers with special techniques might improve results. Third, we recommend testing advanced models like Gradient Boosting or Neural Networks, which might work better for complex patterns in the data. Fourth, combining multiple models using ensemble methods can make predictions more reliable. Finally, using k-fold cross-validation will help ensure the results are consistent and not just specific to one test set.

In conclusion, this project showed that careful model selection, feature engineering, and hyperparameter tuning are important for building accurate regression models. Although our best model performed well, there is still room to improve its accuracy and ability to generalize to new data.

Individual Contributions

1. Zainab Jaradat (1201766):

- **Data Preprocessing and Cleaning:** Handled missing values, standardized numerical features, and ensured categorical variables were encoded correctly to prepare the dataset for modeling.
- **Regression Models:** Built all regression models, including Linear Regression, Polynomial Regression, LASSO, Ridge, and RBF Kernel Ridge Regression. Developed plots to analyze and compare the models' performances.
- **Hyperparameter Tuning:** Conducted hyperparameter tuning using Grid Search for all models and visualized the optimization process to identify the best configurations.
- **Final Evaluation:** Evaluated the final model on the test set, interpreting the results and comparing them to the validation performance.
- **Report and Colab Preparation:** Drafted sections of the report, prepared the final project documentation, and ensured the Colab notebook was well-organized and functional for reproducing the analysis.

2. Ro'a Nafi (1201959):

- **Feature Selection:** Conducted forward feature selection for Polynomial Regression and RBF Kernel, identifying the most impactful features for improving model performance.
- **Updated Preprocessing for Price:** Enhanced the price cleaning process, ensuring all price values were standardized and consistent across the dataset.
- **New Target Variable Implementation:** Selected another relevant target variable from the dataset, built a regression model to predict its values, and evaluated its performance.
- **Visualizations:** Generated and interpreted key visualizations, including residual plots, error distributions, and comparisons of predicted vs. actual values, ensuring that the insights were clearly communicated.
- **Report and Colab Preparation:** Drafted sections of the report, prepared the final project documentation, and ensured the Colab notebook was well-organized and functional for reproducing the analysis.

Both team members worked collaboratively, ensuring the project was comprehensive, the analysis robust, and the report cohesive and well-documented. Additional contributions, such as troubleshooting issues during implementation and final editing, were shared between both members.