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

Assignment #2—Regression Analysis and Model Selection

Prepared by:

Ahmad Qaimari —1210190

Yazan AbuAloun —1210145

Instructor: Ismael Khater
Date: November 28, 2024

Abstract

This assignment compares various regression models using a dataset of cars from the YallaMotor website. The dataset includes features such as car name, price, engine capacity, cylinder count, horsepower, top speed, number of seats, brand, and country of sale. Min-Max scaling was applied to normalize the features. One-hot encoding was used for the country column, and target encoding was applied to the price column. The goal is to evaluate the effectiveness of different regression techniques in predicting car prices and performance metrics, providing insights into the most suitable models for this dataset.

Contents

1 Introduction and Dataset Overview

The dataset contains comprehensive information about cars listed on the **YallaMotor** website, a popular platform for buying and selling vehicles in the Middle East. This dataset includes various attributes that provide detailed insights into the specifications and features of each car. The columns in the dataset are as follows:

Table 1.1: Dataset Columns and Descriptions

Column	**Description**
Car Name	The name of the car
Price	The price of the car
Engine Capacity	The car's engine capacity
Cylinder	The car's cylinder power
Horse Power	The car's horsepower
Top Speed	The car's top speed
Seats	The number of seats in the car
Brand	The car's brand
Country	The country where the site sells this car

This dataset can be used for various analyses, including price prediction, performance comparison, and market segmentation, providing valuable insights for car buyers, sellers, and enthusiasts.

2 Data preprocessing

This section outlines the steps taken to prepare the dataset for analysis. It includes documentation of missing values and the data cleaning strategy.

2.1 Documenting Missing Values

Table 2.1 summarizes the missing values of the dataset. It is shown that cylinder has 624 missing values, which is about %10 of the dataset. The other values are not missing, but have other inherent issues to be discussed later.

Table 2.1: Car Specifications Overview

Feature	**Missing Count**
Car Name	0
Price	0
Engine Capacity	0
Cylinder	624
Horse Power	0
Top Speed	0
Seats	0
Brand	0
Country	0

2.2 Data Cleaning Strategy

To prepare the dataset for analysis, a series of cleaning steps were undertaken. These steps are grouped into the following categories:

2.2.1 Removing Duplicates and Missing Data

- **Remove Duplicates:** All duplicate rows were dropped to ensure unique entries in the dataset.
- **Handle Missing Values:**
  - *horse_power:* Rows with missing or non-numeric values were dropped.
  - *top_speed:* Non-numeric values were converted to NaN for further cleaning.
  - *seats:* Entries with 'N A' were replaced with NaN and imputed using the median grouped by *brand*.
  - *price:* Non-numeric values were converted to NaN and imputed using the mean grouped by *car_name*. Remaining null values were imputed

using the median grouped by *brand*.

- *cylinder:* Missing values were inferred based on the *engine_capacity*.

2.2.2 Standardizing and Cleaning Values

- **Whitespace Removal:** Leading and trailing whitespaces in all columns were removed.
- **Currency Conversion:** Prices were standardized to USD using a predefined conversion function.
- **Unit Standardization:**
  - *engine_capacity:* Non-standard units were converted to liters.
  - *seats:* Numbers were extracted from strings containing 'Seater' and converted to integers.
  - *cylinder:* Rows with 'N/A, Electric' were replaced with '0'.

2.2.3 Correcting Erroneous Data

- **Exchange Erroneous Top Speed Values:** Rows where *top_speed* contained seater values were corrected.
- **Mode Imputation for Entry Errors:** For duplicate entries with conflicting feature values, the mode was used for imputation, grouped by *car_name*.

2.2.4 Handling Outliers

- **horse_power and top_speed:** Outliers were handled using median imputation grouped by *brand*.

2.2.5 Feature Engineering

- **Extract Year:** The year was extracted from the *car_name* column, which was subsequently dropped.

This comprehensive cleaning strategy ensures that the dataset is consistent, free

of missing values, and has outliers handled appropriately, making it ready for further analysis and modeling. Table 2.2 shows the data after the cleaning steps are completed.

Table 2.2: Car Specifications Data

Engine Capacity (L)	Cyl.	HP	Top Speed (km/h)	Seats	Brand	Country	Price (USD)	Year
2.0	4	180	205.0	8	Peugeot	KSA	37 955.25	2021
1.5	4	102	145.0	4	Suzuki	KSA	26 671.95	2021
2.3	4	420	173.0	4	Ford	KSA	53 460.00	2021
1.8	4	140	190.0	5	Honda	KSA	28 179.98	2021
1.8	4	140	190.0	5	Honda	KSA	25 740.45	2021

2.3 Encoding Categorical Values

Encoding is a crucial preprocessing step in machine learning that involves converting categorical data into a numerical format that can be used by machine learning algorithms. Different encoding techniques are used based on the nature of the categorical data and the requirements of the model.

- **One-Hot Encoding**: One-hot encoding transforms categorical variables into a series of binary columns, each representing a unique category. This method is particularly useful for nominal categorical variables with a small number of unique values, as it allows the model to treat each category independently without implying any ordinal relationship.

- **Target Encoding**: Target encoding, also known as mean encoding, replaces each category with the mean of the target variable for that category. This method is effective for high-cardinality categorical variables, as it reduces the dimensionality of the data while preserving the relationship between the categorical variable and the target variable. However, it requires careful handling to avoid overfitting.

In our dataset, the following encoding techniques were applied:

- **One-Hot Encoding for Countries**: The *country* column was one-hot encoded. This transformation created binary columns for each unique country, allowing the model to treat each country independently.

- **Target Encoding for Price**: The *price* column was target encoded. This

transformation replaced each category with the mean of the target variable for that category, effectively reducing the dimensionality of the data while preserving the relationship between the price and other features.

The table below shows pandas dataframe after encoding these columns.

Table 2.3: Car Specifications and Availability in Countries

Engine Capacity (L)	Cyl.	HP	Top Speed (km/h)	Seats	Brand	Price (USD)	Year	Egypt	KSA	Kuwait	Oman	Qatar	UAE
2.0	4	180	205.0	8	Peugeot	37 955.25	2021	0	1	0	0	0	0
1.5	4	102	145.0	4	Suzuki	26 671.95	2021	0	1	0	0	0	0
2.3	4	420	173.0	4	Ford	53 460.00	2021	0	1	0	0	0	0
1.8	4	140	190.0	5	Honda	28 179.98	2021	0	1	0	0	0	0
1.8	4	140	190.0	5	Honda	25 740.45	2021	0	1	0	0	0	0

2.4 Scaling in Machine Learning

Scaling is an essential preprocessing step in machine learning that involves transforming the features of a dataset to a common scale. In this dataset, Min-Max scaling was used to normalize the features.

Min-Max Scaling

Min-Max scaling transforms the features by scaling them to a fixed range, typically [0, 1]. The formula for Min-Max scaling is:

$$X' = \frac{X - X_{\min}}{X_{\max} - X_{\min}}$$

where:

- X is the original feature value.
- $X_{\min}$ is the minimum value of the feature.
- $X_{\max}$ is the maximum value of the feature.
- X' is the scaled feature value.

The table below shows the dataset after scaling was applied.

Table 2.4: Normalized Features of Car Specifications and Country Availability

Index	Engine Cap.	Cyl.	HP	Top Speed	Seats	Brand	Year	Egypt	KSA	Kuwait	Oman	Qatar	UAE
0	0.275	0.313	0.017	0.456	0.188	0.055	0.929	0.000	0.000	0.000	0.000	1.000	0.000
1	0.250	0.313	0.029	0.657	0.188	0.225	0.929	0.000	0.000	0.000	0.000	0.000	1.000
2	0.250	0.250	0.032	0.571	0.188	0.098	0.929	0.000	0.000	1.000	0.000	0.000	0.000
3	0.175	0.250	0.007	0.491	0.125	0.023	0.929	0.000	0.000	0.000	0.000	1.000	0.000
4	0.375	0.375	0.057	0.688	0.188	0.284	0.929	0.000	0.000	0.000	0.000	0.000	0.000

3 Regression Models

3.1 Linear Models

3.1.1 Linear Regression

Linear regression is the easiest form of regression analysis. It finds the straight-line relationship between the input features (like car features) and the target variable (car price). The goal is to draw a line that best fits the data by minimizing the sum of squared residuals between the actual car prices and the predicted prices. It provides a clear and straightforward way to predict car prices based on the given features.

3.1.2 Gradient Descent Solution of linear regression

Gradient descent is an optimization algorithm used to minimize the cost function in machine learning models. It iteratively updates the model parameters in the direction of the negative gradient of the cost function.

The update rule for gradient descent is:

$$\theta_j := \theta_j - \alpha \frac{\partial J(\theta)}{\partial \theta_j}$$

where:

- θ_j are the model parameters.
- α is the learning rate.
- $J(\theta)$ is the cost function.

For linear regression, the cost function $J(\theta)$ is:

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

where:

- m is the number of training examples.
- $h_\theta(x^{(i)})$ is the hypothesis function.
- $y^{(i)}$ is the actual output.

The gradient of the cost function is:

$$\frac{\partial J(\theta)}{\partial \theta_j} = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

The figure below shows gradient descent iterations on this dataset.

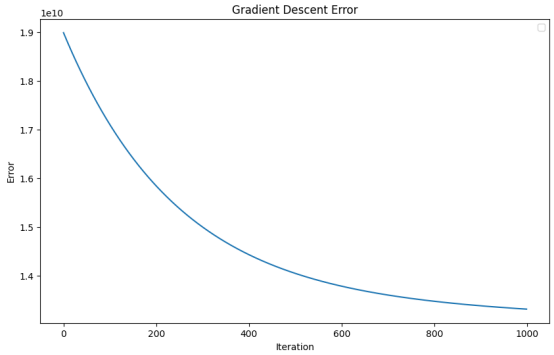

Figure 3.1: Error vs Iteration on YallaMotor dataset with learningRate = 0.01

3.1.3 Comparing gradient with regression

The table below shows the comparison between gradient descent and linear regression, without using API functions.

Table 3.1: Performance Metrics for Regression Models

Model	**MSE**	**MAE**	$\mathbf{R^2}$
Gradient Descent	8 204 837 232.47	42 132.17	0.061
Linear Regression	3 464 737 894.11	25 140.11	0.603

3.1.4 Lasso Regression

Lasso Regression tries to improve on regular linear regression by adding a penalizing term to the cost function that ought to be minimized. In Lasso Regression, the penalizing term is proportional to the absolute values of the coefficients, controlled by a hyperparameter λ (lambda). This penalty term is expressed mathematically as

$$\lambda \sum |w_i|,$$

where w_i represents the coefficients of the model. The parameter λ plays a crucial role in balancing the trade-off between minimizing the residual sum of squares (to achieve accuracy) and shrinking the coefficients towards zero (to enforce simplicity and prevent overfitting). When λ is set to zero, Lasso Regression behaves like standard linear regression, as no penalty is applied. Conversely, as λ increases, the penalty becomes stronger, and some coefficients may shrink exactly to zero, effectively performing feature selection by excluding less relevant predictors.

3.1.5 Ridge Regression

Ridge Regression is similar to Lasso Regression in that it also seeks to improve regular linear regression by adding a penalizing term to the cost function. However, the main difference lies in the nature of the penalty term. While Lasso uses the L_1-norm, Ridge Regression employs the L_2-norm, which is proportional to the square of the coefficients. The penalty term in Ridge Regression is expressed mathematically as

$$\lambda \sum w_i^2,$$

where w_i represents the coefficients of the model, and λ (lambda) is a hyperparameter that controls the strength of the penalty.

Like in Lasso, the parameter λ governs the trade-off between minimizing the residual sum of squares (to achieve accuracy) and shrinking the coefficients (to prevent overfitting). As λ increases, the penalty forces the coefficients to shrink towards zero, but unlike Lasso, Ridge Regression does not set any coefficients to exactly

zero. Instead, it retains all features while reducing the magnitude of the coefficients.

3.2 Non Linear Models

3.2.1 Polynomial Regression

Polynomial regression models non-linear relationships by introducing polynomial terms ($x^2, x^3, \ldots$) to fit a curve rather than a straight line. Increasing the degree of the polynomial adds flexibility, allowing the model to capture more complex patterns in the data. However, higher degrees risk overfitting, where the model performs well on training data but poorly on unseen data.

The Bayesian Information Criterion (BIC) score is a metric used to evaluate model complexity and fit. It is calculated using the formula:

$$\text{BIC} = -2 \cdot \ln(L) + k \cdot \ln(n)$$

where L is the likelihood of the model, k is the number of parameters, and n is the number of data points. The BIC penalizes overly complex models by increasing with the number of parameters. A lower BIC score indicates a better trade-off between simplicity and fit. When deciding the degree of a polynomial, the degree corresponding to the lowest BIC score is typically selected, ensuring the model captures patterns without overfitting.

3.2.2 Support Vector Regression (with Gaussian (RBF) Kernel)

Support Vector Regression (SVR) with the Radial Basis Function (RBF) kernel is a type of Support Vector Machine (SVM) tailored for regression tasks. By utilizing the RBF kernel, SVR maps input features into a higher-dimensional space, enabling it to model complex, non-linear relationships effectively. The kernel is governed by the parameter γ, which controls the kernel's sensitivity to data points, while the regularization parameter C and the margin of tolerance ϵ help balance model flexibility and generalization. This approach leverages the core principles of SVMs, such as maximizing the margin, to provide smooth and accurate predictions, making it ideal for capturing intricate patterns in regression problems.

4 Performance Analysis of Regression Models

The following table outlines the performance of each regression model (before any hyperparameter tuning) on the validation set, defined by the mean square error, mean absolute error and the R2 score.

Table 4.1: Performance of various regression models on the validation set.

Model	MAE	MSE	R2
Lasso	24940.533572	2311072654.651835	0.685126
Ridge	23582.412437	2215829676.005766	0.698102
Linear Regression (degree=2)	24945.061158	2311301788.082292	0.685094
Polynomial Regression	26530.800876	2424462615.761249	0.669677
SVR	46022.437360	8253887438.358505	-0.124559

To further improve the performance of the regression models, Grid Search was applied to optimize the hyperparameters for each model. An exhaustive search was conducted over a specified parameter grid, with all combinations of parameters systematically tested to find the best-performing set. This process ensured that the optimal configuration for each model was selected.

For each model, different values for hyperparameters such as the regularization strength for Lasso and Ridge regressions, and the C parameter for the Support Vector Regression (SVR) model were tested. The following table outlines the Grid Search results for each model.

Table 4.2: Performance of regression models on the validation set After Grid Search.

Model	MAE	MSE	R2
Lasso	24660.134969	2298883814.979136	0.686786
Ridge	23496.891406	2345438885.180511	0.680443
SVR	30172.121708	5037649760.629855	0.313641

For the Polynomial Regression model, instead of using Grid Search, the optimal degree was determined based on the Bayesian Information Criterion (BIC) scores.

The degree of the polynomial that resulted in the lowest BIC score was selected as the optimal degree for the model. The BIC scores were plotted against the polynomial degrees, as shown in the figure below. The plot reveals that the BIC score decreases as the polynomial degree increases, but the lowest BIC score is observed at degree 2, suggesting that a quadratic model provides the optimal balance between model complexity and goodness of fit. This degree was chosen as the optimal degree for the Polynomial Regression model, as it minimizes overfitting while still capturing the underlying pattern in the data.

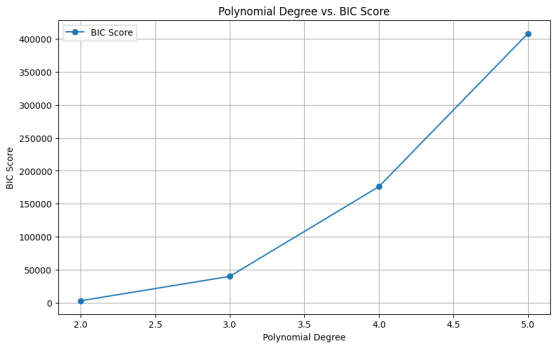

Figure 4.1: Polynomial Degree vs BIC Score Graph Plot

After determining the optimal polynomial degree using the Bayesian Information Criterion (BIC) scores, the performance metrics for the Polynomial Regression model were further evaluated across different polynomial degrees.

The results in the following table show that the MSE and MAE begin to increase significantly as the model complexity grew, suggesting that higher-degree polynomials led to overfitting.

Table 4.3: Performance metrics for polynomial regression models with different degrees

Degree	2	3	4	5
MSE	2.4245e+09	9.7742e+27	1.2787e+29	4.9171e+17
MAE	2.6531e+04	3.9448e+12	2.3445e+13	4.8896e+07
R2	6.6968e-01	-1.3317e+18	-1.7422e+19	-6.6994e+07
BIC	2.9920e+03	3.9948e+04	1.7590e+05	4.0804e+05

5 Evaluation on the Test Set

After testing each model on the validation set, the results in Tables 4.1 and 4.2 indicate that the Lasso Regression model outperformed the other models in terms of overall performance. To validate the model's performance, it was tested on the unseen test set. The test set evaluation ensures that the selected model generalizes well to new data, providing an unbiased assessment of its predictive capabilities. Key metrics such as Mean Squared Error (MSE), Mean Absolute Error (MAE), and R^2 score were computed for the test set and compared to the validation set results.

Metric	Mean Absolute Error	Mean Squared Error	R2 Score
Value	24694.988254	3560159080.590832	0.592437

Table 5.1: Performance metrics for the Lasso model on the Test Set

As table 5.1 shows, the Lasso Regression model demonstrated strong and consistent performance on the test set, achieving an MSE MAE and R2 score comparable to those observed during validation. This consistency underscores the model's ability to generalize effectively, validating the choice of Lasso Regression as the best-performing model. However, the model is not without limitations. While Lasso Regression's ability to shrink less important coefficients to zero aids in feature selection, it may inadvertently exclude features that contribute marginally to predictive performance, especially in cases of correlated predictors. Moreover, the selection of the regularization parameter (α) plays a critical role in determining the model's performance. Although cross-validation was used to optimize α, further experimentation with alternative hyperparameter tuning methods could yield better results. Additionally, Lasso assumes a linear relationship between predictors and the target variable, which may not adequately capture nonlinear patterns in the data.

6 Forward Feature Selection

Forward Feature Selection is a method used to iteratively select the most relevant features for building a regression model. The idea is to start with an empty set of features and then sequentially add the feature that improves the model's performance the most. This approach is particularly useful in situations where there are many potential features, and including all of them might lead to overfitting or unnecessarily complex models.

- **Reduces Complexity:** By selecting only the most important features, the model becomes simpler, reducing the risk of overfitting.

- **Improves Interpretability:** With fewer features, it is easier to understand the relationships between predictors and the target variable.

- **Enhances Generalization:** A model trained with relevant features is likely to perform better on unseen data.

The Lasso regression model was identified as the best performing model during the evaluation phase due to its ability to balance model complexity and predictive performance. To further refine the Lasso model, forward feature selection was applied to determine the most important features for prediction.

Table 6.1: Results of forward feature selection for the Lasso regression model

Model	Iteration 1	Iteration 2	Iteration 3
Added Feature	brand	cylinder	country_ksa
Validation Score	0.620608	0.700749	0.704041

The forward feature selection process identified three key features—cylinder (1), brand (5), and KSA country (8)—as the most relevant for predicting car prices. The cylinder feature reflects the engine size and performance, which directly impacts the car's price, with larger engines typically being more expensive. The brand of the car is crucial, as luxury and well-known brands tend to command higher prices due to their reputation, quality, and desirability. Lastly, the KSA country feature (one-hot encoded) highlights the importance of regional factors, as demand for certain types of cars in Saudi Arabia, such as luxury or large-engine vehicles, influences pricing.

7 Error Visualization

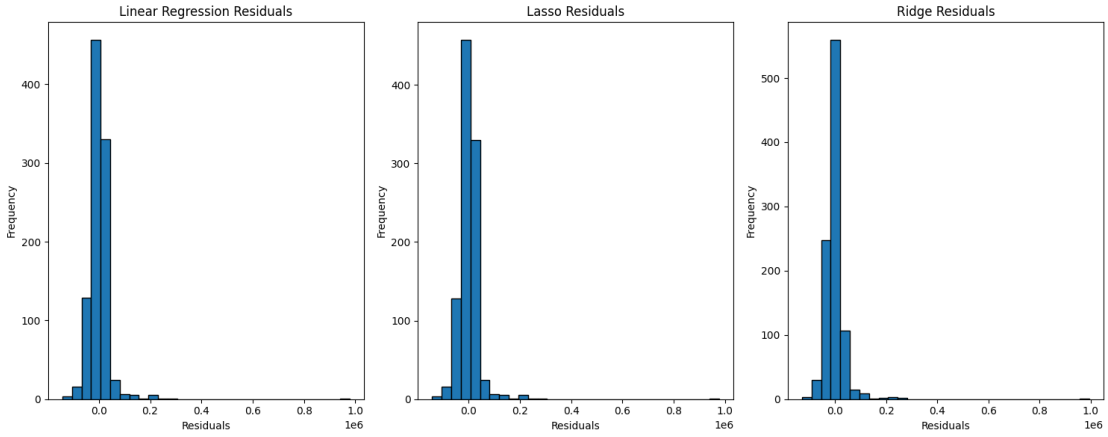

Figure 7.1: Histogram for Error Residuals

The above figure highlights the concentration of error residuals across the regular,Lasso and Ridge Linear Regression Techniques.

8 Trying a different Target Variable

In this analysis, the target variable was changed from price to top speed, with the goal of predicting a car's maximum speed based on various input features. This shift in the target variable allowed for the exploration of different patterns and relationships between the features and the new target.

For encoding categorical variables, Frequency Encoding was employed as an alternative to Target Encoding. Frequency Encoding involves replacing each category in a categorical feature with the frequency of its occurrence within the dataset. This method provides a straightforward numerical representation of the categories based on how often they appear, making it an effective way to incorporate categorical features into the regression model. Unlike Target Encoding, which involves encoding categories based on the mean of the target variable for each category, Frequency Encoding reduces the risk of overfitting by not directly involving the target variable. While it may not capture the relationships between categories as comprehensively as Target Encoding, Frequency Encoding is computationally efficient and still offers useful insights, especially when the frequency of categories correlates with the target variable. This approach ensures that categorical data is appropriately represented for regression analysis. The table below shows the

results of the different regression techniques with top speed as the target variable and frequency encoding:

Metric	Lasso	Ridge	Linear Regression	Polynomial Regression	SVR
MAE	28.979088	22.684962	21.888996	16.792339	29.030198
MSE	1324.578111	895.230076	845.736632	586.049709	1436.807663
R2	0.348435	0.559633	0.583979	0.711720	0.293228

Table 8.1: Comparison of Regression Models on top speed

9 Conclusion

In this assignment, we compared various regression models to predict car prices using the dataset from the YallaMotor website. The dataset underwent preprocessing steps, including Min-Max scaling for normalization and encoding techniques such as one-hot encoding for the **country** feature and target encoding for the **price** feature. These steps ensured that all features contributed equally to the models and that categorical variables were appropriately transformed.

The analysis demonstrated that certain regression models outperformed others in predicting car prices. Specifically, models like **Linear Regression with Lasso regularization** showed lower MSE and better performance metrics compared to other models tested. The effectiveness of these models highlights the importance of selecting suitable algorithms for regression tasks.

Overall, the results indicate that appropriate preprocessing and the choice of regression model are crucial for achieving accurate predictions. This comparison provides valuable insights into the strengths of different regression techniques when applied to complex datasets in the automotive domain.