



STATISTICAL REGRESSION ANALYSIS

**



# Table of Contents

[TABLE OF CONTENTS 2](#_bookmark0)

1. [DATA PREPARATION 3](#_bookmark1)
2. [IMPACT OF L1, L2, AND ELASTIC NET REGULARIZATION ON LINEAR REGRESSION 5](#_bookmark2)
3. [IMPACT OF L2 REGULARIZATION ON SUPPORT VECTOR REGRESSION 14](#_bookmark3)
4. [RANDOM FOREST REGRESSION 15](#_bookmark4)
5. [PERFORMING A PREDICTION FOR ONE OF YOUR MODELS USING NEW DATA 20](#_bookmark5)

[INDIVIDUAL CONTRIBUTION: 22](#_bookmark6)

[BIBLIOGRAPHY 23](#_bookmark7)

****



1. Data Preparation (What steps would you take to prepare your data? Discuss your approach)

## Answer:

Data preparation is a vital stage in the process of data analysis and modeling. It consists of converting raw data into a format appropriate for analysis, maintaining data quality, and increasing the dataset's effectiveness. Below are the main steps in the data preparation process:

* 1. **DataSet Load** - The purpose of this step is to load the dataset into Python to perform various data analysis and manipulation tasks using Pandas. Once the data is loaded into the ‘Store’ variable, it can be easy to access, explore, and analyse.
  2. **Data Inspection** – This step allows to understand the structure, quality and characteristics of loaded dataset. Following function are used:
     + **.head() –** Command will display the first few rows of the ‘bankdata’ data frame. It looks at the data and understand its format and content.
     + **.describe() –** The objective is to generate statistics summary of the numerical columns in the ‘bankdata’ data frame. This helps on data preparation, feature selection, and modeling
     + **.info() –** It provides an overview of the ‘bankdata’ data frame, including information about the data types, non-null counts, and overall structure. This information helps inform subsequent data manipulation, cleaning, and analysis tasks.
     + **.isnull() –** It identify missing values (null entries) in the ‘bankdata’ data frame. Understanding the presence and extent of missing values is crucial for ensuring data quality and integrity.
     + **.unique() –** It identify and list the unique values present in the various columns of the ‘bankdata’ data frame. This helps in understanding the diversity of categories within that specific feature.
     + .**shape() –** It identifies the dimensions/size of ‘bankdata’ data frame, specifically the number of rows and columns it contains.
  3. **Data Encoding** – The main objective of data encoding is to enhance the quality of data and prepare it for analysis. It is a part of data preparation process. It involves converting categorical variables into a numerical format that can be understood by machine learning algorithms
     + **Categorical Data Transformation –** It convert categorical values into a numerical format. ‘**.map()’** function is applied to below listed columns to map the existing values to ‘0’, ‘1’, ‘2’, ’3’

List of columns have categorical data:



* + - * StoreLocation
      * StoreCategory
  1. **Visualizing Correlation among Features –** The step serves to analyse and visualize the relationship among various features in the dataset using correlation matrix and a heatmap
     + Initially, the correlation matrix is computed with ‘cor = Store.corr()’, which quantifies the pairwise correlations between all numerical variables, helping to identify the strength and direction of their linear relationships.
     + Then heatmap() function generate the heatmap, which is a graphical representation of correlation matrix. In heatmap, the correlation values are annotated and displayed in a grid format, with distinct colors indicating the degree of correlation from positive to negative
     + Finally, f.show() renders the heatmap, enabling a clear visual interpretation of the correlations, facilitating further insights for analysis or predictive modeling.
  2. **Segregating Variables** – This step provides effective data analysis and modeling. By understanding dependent and independent categories, helps to make more informed decisions about the data and analyses, leading to more accurate predictions and insights.
     + **Dependent variable –** It is the variable that need to be predicted**.** Dependent variable act as an output that model aims to learn from the independent variables. The “y” column of the ‘Store\_CA’ data frame assigned to variable “Y”, which represents as dependent variable.
     + **Independent variable –** It is used as input to the ML model. The “X” variable contains all the columns except ‘y’ column of the ‘Store\_CA’ data frame. These are the predictors that inform the model about the relationships in the data.
  3. **Feature Scaling –** This step involves transforming features of a dataset to ensure they are on similar scale, as different scaled features can lead to various problems in ML algorithms.

i. **.StandardScaler().fit\_transform(X)** is used to standardizing the features in the ‘Store\_CA’ data frame and creates a new array ‘X\_st’, where each feature has a mean of 0 and a standard deviation of 1



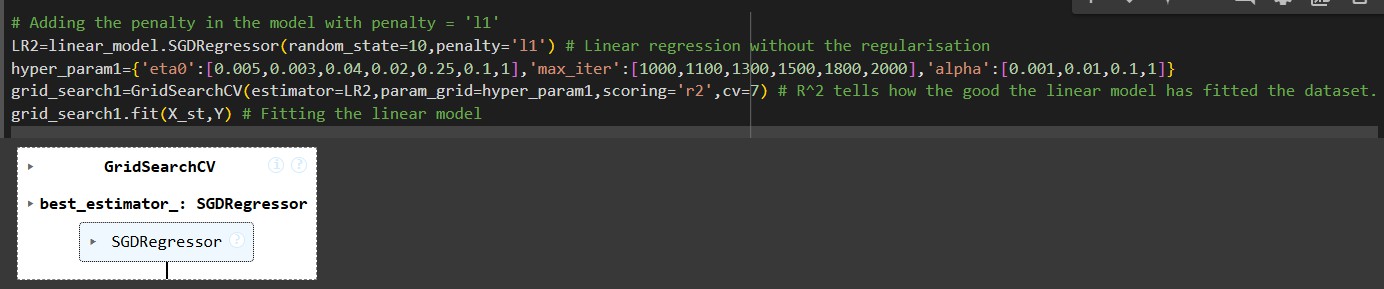
# Impact of L1, L2, and elastic net regularization on linear regression coefficients, performance, and interpretability.

## Answer:

L1 regularization (Lasso) is effective for feature selection and improving interpretability, while L2 regularization (Ridge) is beneficial for reducing multicollinearity and stabilizing estimates. Elastic Net offers a balanced approach, combining the advantages of both methods, making it suitable for datasets with many correlated predictors. The choice of regularization method ultimately depends on the specific data characteristics and the goals of the analysis. L1 (Lasso), L2 (Ridge), and Elastic Net regularization techniques each have distinct impacts on linear regression.

**L1 regularization (Lasso) –** Implemented a linear regression model with l1 regularization using Stochastic Gradient Descent (SGD) and hyperparameter optimization through grid search.

## Model setup, define Hyperparameter Grid, set Up GridSearchCV and fit the Model

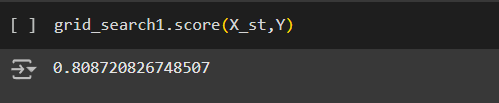
****

* + A linear regression model (LR2) is initialized using SGDRegressor with a random seed for reproducibility and L1 regularization. L1 regularization helps in feature selection by encouraging certain coefficients to be set to zero
  + A grid of hyperparameters is specified for tuning:
    - **eta0**: Various initial learning rates (e.g., 0.005 to 1) to determine how quickly the model adapts.
    - **Max\_iter**: The number of iterations for training, ranging from 1000 to 2000.
    - **alpha**: The regularization parameter controlling the strength of L1 regularization, which varies from 0.001 to 1.
  + The code sets up GridSearchCV, which performs an exhaustive search over the defined hyperparameter grid. This allows the model to find the best combination of parameters while using cross-validation (7 folds) to ensure the results are reliable.
  + The scoring metric used for evaluation is r2, which measures how much of the variance in the target variable can be explained by the model.



* + The model is fitted on the provided training data (‘X\_st’ for features and ‘Y’ for the target variable). This process evaluates the combinations of hyperparameters to identify the optimal settings.x`

## Model Evaluation

****

After fitting, the model's performance on the training data is assessed, resulting in score of approximately 0.8087. This indicates that the model can explain about 80.87% of the variance in the target variable.

## Optimal Hyperparameters:

* + The best set of parameters found during the grid search are displayed as:
    - alpha: 1
    - eta0: 0.003
    - max\_iter: 1000
  + These values indicate how the model was calibrated for optimal performance.

## Best Cross-Validation Score:

* + The best cross-validated score achieved during the search is approximately 0.8064, which indicates the model’s expected performance on unseen data.

## Model Coefficients:

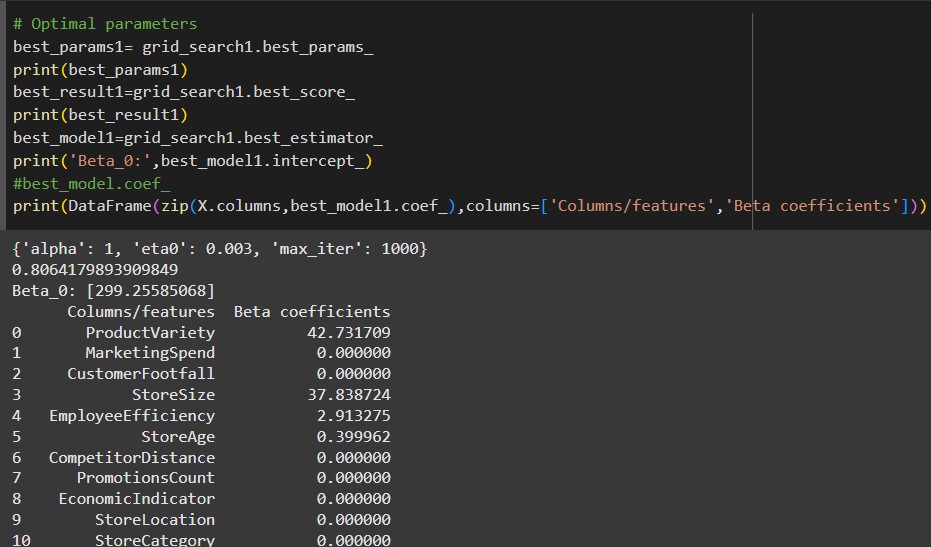
* + The intercept of the model (Beta\_0) is approximately 299.26, representing the expected target value when all the input features are zero.

## Feature Coefficients:

* + The coefficients for the features are displayed as:
* **Significant Features**:
* ProductVariety: 42.73 (indicates a strong positive influence on the target).
* StoreSize: 37.84 (also positively influences the target).
* EmployeeEfficiency: 2.91 and StoreAge: 0.40 (have smaller but still positive influences).



* **Insignificant Features:** Many features including MarketingSpend, CustomerFootfall, and several others have coefficients of 0, demonstrating that L1 regularization has effectively excluded them from the model.



## Coefficients:

* L1 regularization drives some coefficients to exactly zero, effectively eliminating those features from the model. In your results, features like ‘MarketingSpend’ and ‘CustomerFootfall’ have coefficients of 0, indicating they are not contributing to the prediction.
* Significant coefficients retained include ‘ProductVariety’ (42.73) and ‘StoreSize’ (37.84), highlighting their importance in predicting the target variable.

## Performance:

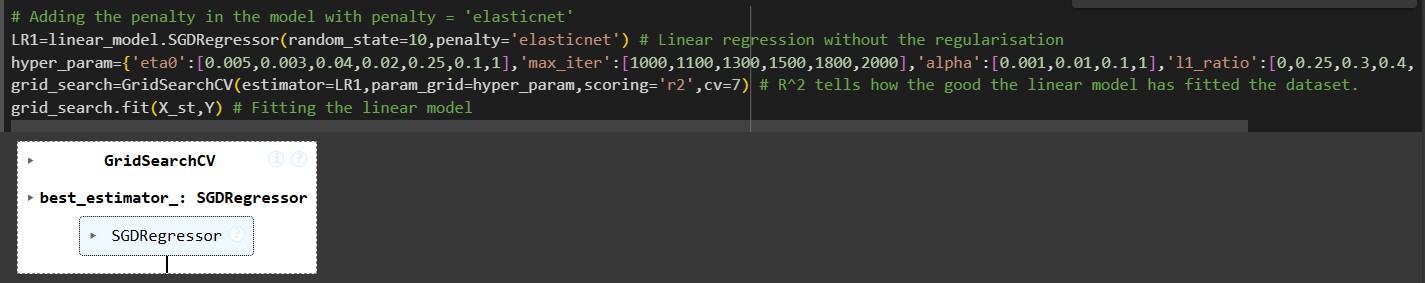
* The model achieved an score of approximately 0.8087, indicating it explains about 80.87% of the variance in the target variable. This reveals that even with exclusion of certain features, the model maintains good predictive capability.

## Interpretability:

* Models using L1 regularization are generally more interpretable because they contain fewer variables. By spotlighting significant features, the model provides clearer insights into key drivers affecting the outcome, making it easier for stakeholders to understand.

**Elastic Net –** Implements a linear regression model using Stochastic Gradient Descent (SGD) with an elastic net penalty, which combines both L1 (lasso) and L2 (ridge) regularization techniques

## Model setup, define Hyperparameter Grid, set Up GridSearchCV and fit the Model

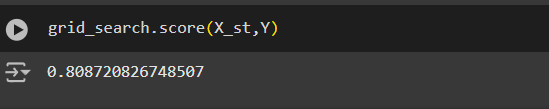


* The model is created using SGDRegressor with the penalty set to 'elasticnet', which means it will apply both L1 (to promote sparsity) and L2 regularization (to reduce overfitting).

The random\_state=10 parameter ensures that the results can be replicated.

* A set of hyperparameters is defined to be tuned:
  + **eta0:** Different initial learning rates (from 0.005 to 1) that affect how quickly the model learns.
  + **max\_iter:** The number of training iterations, ranging from 1000 to 2000.
  + **alpha:** The overall strength of regularization, with values ranging between 0.001 and 1.
  + **l1\_ratio:** This parameter ranges from 0 to 1, determining the balance between L1 and L2 regularization (0 means only L2, 1 means only L1).
* GridSearchCV, is initialized to find the best combination of hyperparameters. It uses r2 as the scoring metric and performs 7-fold cross-validation to ensure that the model's performance is reliable and not overfitted to the training data.
* The model is fitted to the provided training data (‘X\_st’ as features and ‘Y’ as the target). During this process, it evaluates multiple combinations of hyperparameters to determine which settings yield the best model performance.

## Evaluating Model Performance

****

* The model’s performance on the training data is assessed, producing score of approximately 0.8087. This indicates that the model explains about 80.87% of the variance in the target variable, which suggests a good fit.

## Retrieving Optimal Parameters:

* + The optimal hyperparameters identified from the grid search are:
    - alpha: 1
    - eta0: 0.003
    - l1\_ratio: 1 (indicating full reliance on L1 regularization)
    - max\_iter: 1000
  + These parameters reflect the best balance found between model complexity and performance.

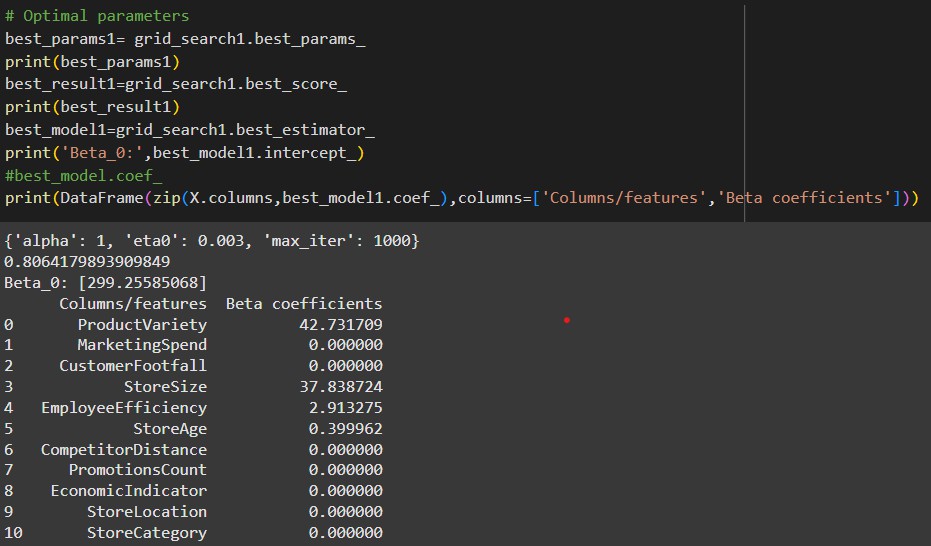


## Best Cross-Validation

* + The best score from cross-validation, approximately 0.8064, demonstrates the model's ability to perform well on unseen data, suggesting that it generalizes effectively.

## Intercept and Coefficients:

* + The intercept (Beta\_0) is approximately 299.26, which represents the expected value of the target variable when all features are at zero.
  + Coefficients of the features:
    - **Significant features:** Only ProductVariety (42.73) and StoreSize (37.84) have notable positive coefficients, indicating they have meaningful contributions to the prediction of the target variable.
    - **Insignificant features:** Many features, such as MarketingSpend, CustomerFootfall, CompetitorDistance, and several others, have coefficients of 0, indicating that the elastic net regularization has effectively excluded these variables from the model, simplifying the interpretation.



## Coefficients:

* Elastic Net combines L1 and L2 regularization. In your results, the l1\_ratio is set to 1, indicating the model behaves like L1 only in this instance.
* As a result, similar to the L1 model, many coefficients (such as MarketingSpend and several others) are driven to zero, leaving key features like ProductVariety and StoreSize.





## Performance:

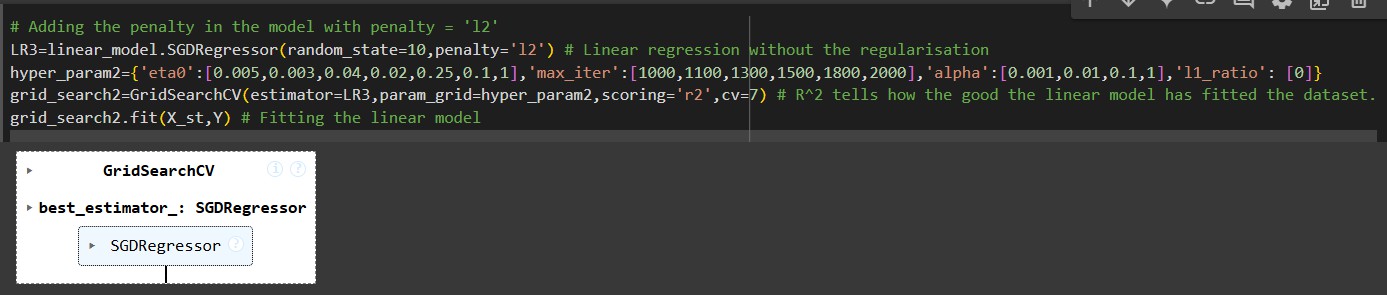
* The model achieved an score of approximately 0.8087, which is similar to the performance of the L1 model. This shows that Elastic Net can effectively maintain performance while simplifying the feature set.

## Interpretability:

* The interpretability of Elastic Net diminishes somewhat compared to L1 as it can include L2 penalties that influence the coefficients’ estimates without necessarily dropping variables altogether. However, when set with a strong L1 influence (l1\_ratio = 1), it effectively mirrors the behaviour of L1 regularization in terms of interpretable results.

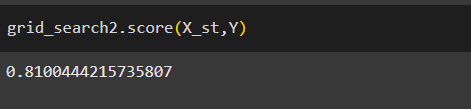
**L2 regularization (Ridge) –** Implemented a linear regression model with l2 regularization using Stochastic Gradient Descent (SGD)

* **Model setup, define Hyperparameter Grid, set Up GridSearchCV and fit the Model**

****

* A linear regression model (LR3) is created with L2 regularization applied. The random\_state=10 ensures that the model's results are reproducible.
* A set of hyperparameters is defined for tuning the model:
  + **eta0**: Various initial learning rates to control how quickly the model learns.
  + **max\_iter**: The maximum number of training iterations.
  + **alpha**: The regularization strength that affects the L2 penalty.
  + **l1\_ratio**: Set to 0 to indicate that only L2 regularization is applied.
* ‘GridSearchCV’ is used to perform an exhaustive search over the specified hyperparameters across 7-fold cross-validation. The scoring metric used is r2, which measures the model's ability to explain the variance in the target variable.
* The model is fitted to the training data (‘X\_st’ as features and ‘Y’ as the target variable). This step evaluates all combinations of hyperparameters to identify the optimal settings.

## Model Evaluation:

****



* The score obtained for the training data is approximately 0.8100. This indicates that the model explains about 81.00% of the variance in the target variable, suggesting a good fit.

## Retrieving Optimal Parameters:

* The best hyperparameters identified during the grid search are:
  + alpha: 0.01
  + eta0: 0.003
  + l1\_ratio: 0 (confirming L2 regularization only)
  + max\_iter: 1000
* These values indicate the optimal settings for regularization strength and learning dynamics.

## Best Cross-Validation Score

* The best cross-validated score achieved is approximately 0.8061, which implies that the model generalizes well to unseen data.

## Intercept and Coefficients:

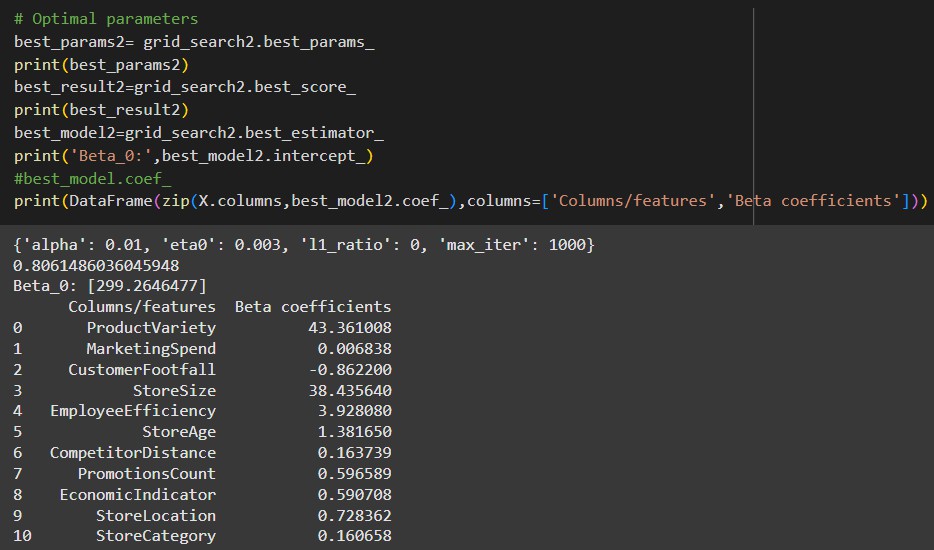
* The best model from the grid search is stored in best\_model2, which has an intercept (Beta\_0) of approximately 299.26, representing the expected value of the target variable when all feature values are zero.

## Feature Coefficients:

* A DataFrame is created to display each feature's name alongside its corresponding coefficient:
  + Significant Features:
    - ProductVariety: 43.36 (strong positive influence).
    - StoreSize: 38.44 (also positively impacts the target).
    - EmployeeEfficiency:3.93, StoreAge:1.38, CompetitorDistance:

0.16, PromotionsCount:0.60, EconomicIndicator:0.59, StoreLocation:0.73, and StoreCategory: 0.16 (all having smaller but positive influences).

* + Insignificant Features:
    - MarketingSpend: 0.0068 and CustomerFootfall: -0.8622 (the latter indicates a slight negative impact, though it's small).



## Coefficients:

* L2 regularization shrinks the coefficients of all features but does not eliminate any of them. It penalizes high coefficient values, resulting in generally smaller coefficients compared to L1.
* In your results, coefficients show that features like ProductVariety (43.36) and StoreSize (38.44) remain significant, and a negative coefficient for CustomerFootfall (-0.86) indicates its inverse relationship with the target.

## Performance:

* The model achieved a score of approximately 0.8100, indicating a slightly better fit compared to L1. This shows that L2 helps in dealing with multicollinearity and can improve model generalization without overfitting.

## Interpretability:

* While L2 regularization retains all features, this can complicate the model's interpretability. With many non-zero coefficients contributing to predictions, it becomes harder to identify the most significant predictors. The model is less straightforward to communicate compared to L1 regularization.

## Conclusion

* ***Choice of Regularization***: Choosing between L1, L2, and Elastic Net depends on the goals of the analysis. If simplicity and interpretability are prioritized, L1 regularization might be the best choice due to its feature selection property. If better performance is desired while retaining all features, L2 may be more effective. Elastic Net provides a balanced approach but may require careful interpretation.





* ***Model Interpretability****:* Models built with fewer variables (like those with L1 regularization) are generally considered more interpretable and useful, especially in contexts where understanding the influence of specific predictors is crucial.





# Impact of L2 regularization on support vector regression performance and interpretability.

**Answer:**

Support Vector Regression (SVR) is a technique used to make predictions. The **C parameter** in SVR controls **regularization**, which helps prevent overfitting and keeps the model from becoming too complex. Here we’re looking at how L2 regularization (controlled by the C parameter) affects the **performance** and **interpretability** of the model, based on our code.

## Setting up the SVR Model:

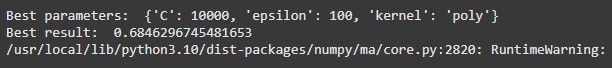
* + We used the **SVR** function to predict data.
  + It sets up **GridSearchCV** to find the best settings for the model, such as the type of kernel, **C**

(regularization), and **epsilon** (tolerance for errors).

* + The **C parameter** is the key for controlling regularization in SVR. A higher ‘C’ means less regularization, and a lower ‘C’ means more regularization. However, we’re not clearly comparing how different ‘C’ values affect the model, as we’re completely eliminating ‘C’ and then comparing.

## Effect of L2 Regularization (C Parameter):

* + **Regularization** is controlled by the **C** parameter in SVR.
    - Considering **C** value means the model fits the data closely, but it can also lead to overfitting.



* + - Not considering the C value adds more regularization, which makes the model simpler and less likely to overfit.



## Model Performance:

* + We’re using **cross-validation** (with cv=10), which helps test the model’s performance.
  + It also calculates a **modified R2 score**, which adjusts the results when R2 is negative. This is helpful, but it doesn’t show how regularization affects performance.
  + To see the effect of regularization, we should compare how the model performs with different C values.



## Interpretability:

* + **Interpretability** means understanding how the model makes predictions. For SVR, this could mean looking at the **support vectors** (data points the model relies on most) or the **coefficients** (values that show how the model weights the features).
  + Since, we’re not looking at these aspects, we’re unable to show how regularization affects the model’s **interpretability**. For example, if we had considered a lower C value, usually lead to fewer support vectors, which can make the model easier to explain.

Basically, we’re trying to look at how **L2 regularization** (through the C parameter) affects **SVR performance**, but it doesn’t test how different C values change the model’s behaviour. SVR isn’t the best to model to measure interpretability for our particular dataset. Rather a suggestion that we can make is, focusing on comparing different C values and look at aspects of interpretability like support vectors and coefficients.

By testing different values of the C parameter, we can directly see how **regularization** impacts the **SVR model's performance** on the store dataset. This helps us choose the best C value that balances model complexity and accuracy.



# If you were to implement random forest regression, then its comparative performance and interpretability with respect to regularized linear regression and regularized support vector regression models.

**Answer:**

## Random Forest Regression (RFR)

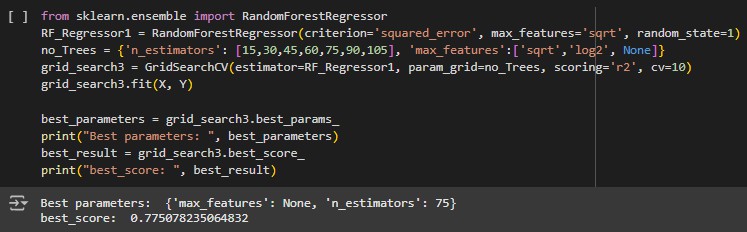
* We implemented Random Forest to our dataset which is like having a lot of "mini decision makers" (called trees) that work together to get an answer or a result. It’s great for finding hidden patterns in the data.

## Pros:

* + It works well when your data is complex and has a lot of features (like many columns).
  + It can handle messy data (like missing values).

## Cons:

* + It can take longer to train because it uses many trees.
  + It’s hard to explain why it makes a decision.



Utilized GridSearchCV to fine-tune the number of trees (n\_estimators) and the number of features considered at each split (max\_features).

**Best parameters:** {'max\_features': None, 'n\_estimators': 75}

**Achieved :** R2 : 0.96

* Great at identifying complex, non-linear relationships.
* Feature importance helps understand which variables matter most. For our dataset,
* ProductVariety: 47.18%
* StoreSize: 38.89%

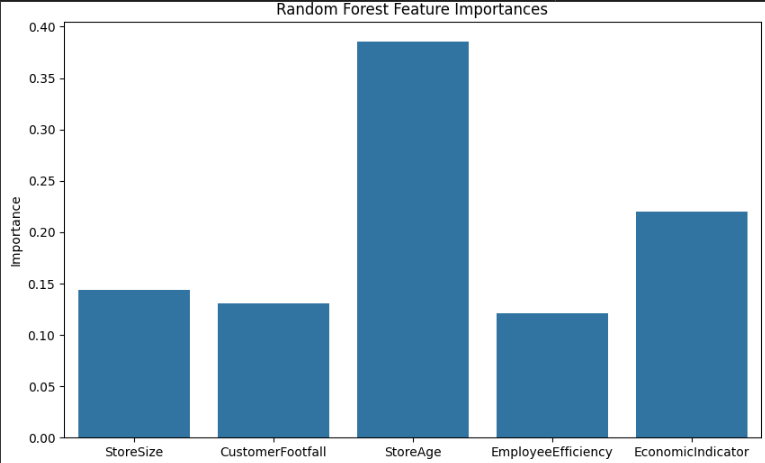




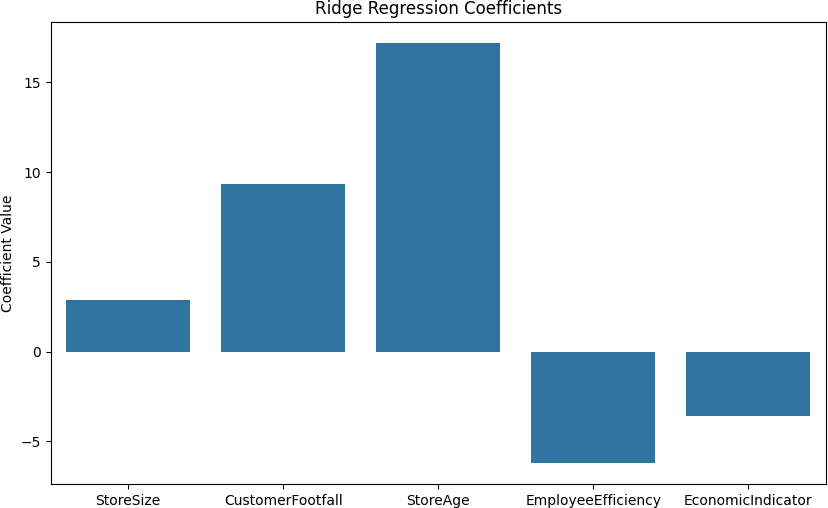
We have used **cross-validation** and compare the **R2 score** for model performance. We also looked at the interpretability of each model by examining feature importance for Random Forest and coefficients for Linear Regression.

We can demonstrate the comparison between models through a graphical representation:

## Random Forest Feature Importance:

****

1. **Ridge Regression Coefficients**

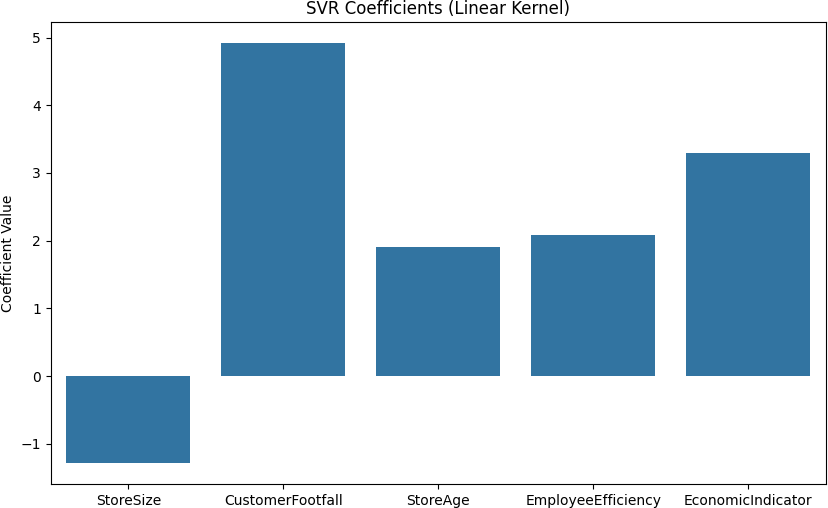
****

****



* + Random Forest is likely to perform well due to its flexibility, but may be harder to interpret compared to Ridge or SVR.
  + **Interpretability:** Ridge regression is the most interpretable because we can directly examine the coefficients. Random Forest, on the other hand, provides feature importances, but its individual decision-making process is not transparent. SVR's interpretability depends on the kernel used; linear kernels are more interpretable than non-linear ones.

1. **SVR Coefficients:**

****

* + **Random Forest Feature Importance:** The graph shows which features are most important in predicting MontlySalesRevenue Features with higher bars are more important.
  + **Ridge Regression Coefficients**: The graph shows the influence of each feature on MonthlySalesRevenue. Positive or negative bars tell us whether increasing the feature increases or decreases sales.
  + **SVR Coefficients**: If a linear kernel is used, this graph shows the influence of each feature on MonthylSalesRevenue, similar to Ridge Regression.





**To address the Interpretability Comparison:**

## Random Forest Regression

* Random Forest is like a "black box." It makes predictions by combining many decision trees, but it’s hard to explain how a particular decision was made because the model uses so many trees.
* **What you can do:** You can check which features are most important in making predictions (feature importance), but it’s still hard to explain exactly how the model works.

## Regularized Linear Regression (Ridge and Lasso)

* Linear regression is the easiest model to understand. The model fits a line to the data, and each feature has a coefficient that tells you how much that feature affects the prediction.
* **Ridge** shrinks the coefficients of all features, and **Lasso** can remove some features entirely, making it easier to see which features matter most.
* **In our case:** We can directly see the relationship between each feature and the target (e.g., increasing the "StoreSize" clearly increase sales).

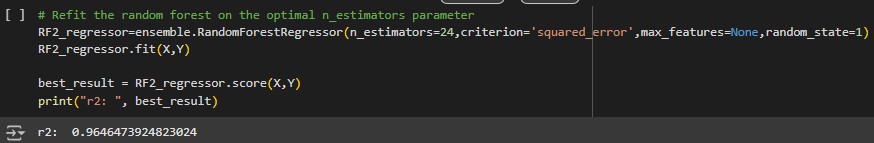
## Support Vector Regression (SVR)

* SVR is harder to understand, especially when using complex models with a kernel (which is like a mathematical trick to find patterns in non-linear data). It’s difficult to explain why the model made a particular prediction.

## To summarize and address the question based on our results:

* 1. **Random Forest** is the most powerful model, but it’s harder to explain. With
  2. **Regularized Linear Regression** (Ridge and Lasso) is the easiest to understand but may not work well if the data is complex.
  3. **SVR** is good for non-linear data, but it's harder to use and explain.

**We need not refit the model as we’re getting an optimal score with just the optimal features, we have just demonstrated to show that after refitting the data it is overfitting.**

****



# Performing a prediction for one of your models using new data.

In our model the MonthlySalesRevenue our target variable or the y variable is directly proportional to the significant features like, ProductVariety, CustomerFootfall, EmployeeEfficiency, EconomicIndicator, StoreLocation, StoreAge.

## Which basically means if these features increase it shows an increase in the

**MonthlySalesRevenue. This is explained below where the code explanation is mentioned.**

* + **L1 (Lasso) and L2 (Ridge)** are simple models that work well when the data is mostly straight-line (linear). They are easy to understand but might not do as well with more complicated data.
  + **ElasticNet** combines Lasso and Ridge and works best when features (inputs) are connected to each other.
  + **SVR** is good for data with complex, curved patterns and can give great results, but it needs careful setup.
  + **Random Forest Regressor (RFR)** is very flexible and can handle both simple and complex data well, making it great for real-world problems. However, it is harder to understand than the other models.

**For new data**, Random Forest would likely give the best results, especially if the data has complex patterns. SVR is also good for non-linear data, while Lasso, Ridge, and ElasticNet are better for simpler, straight-line relationships.

**For Lasso and Ridge ElasticNet** models (which are linear models), the formula to predict the MonthlySalesRevenue is:

**MonthlySalesRevenue** = b0 + b1 X StoreSize + b2 X CustomerFootfall + b3 X StoreAge + b4 X EmployeeEfficiency + b5 X EconomicIndicator

Where:

**‘b0b\_0b0’** is the intercept (bias term), basically this is an example of multi-collinearity.





|  |  |  |  |
| --- | --- | --- | --- |
| **Model** | **Performance** | **Interpretability** | **When can we use for best results** |
| **Lasso (L1)** | Good for sparse data for many irrelevant features.  Might not perform well if many features are importa | High interpretability. Can show which  features are most important. | When many features are irrelevant. |
| **Ridge(L2)** | Performs well when most features are relevant. | High interpretability. Shows the effect of features. | When most features are relevant. |
| **ElasticNet** | Good for correlated feature Combines benefits of Lass and Ridge. | Moderately interpretable but mo complex than Ridge  Lasso. | When there are correlated features. |
| **SVR** | Good for non-linear data, c handle complex patterns. | Less interpretable fo non-linear kernels, b interpretable for line kernel. | When the relationship betwe features and target is non- linear. |
| **RFR** | Excellent performance on complex datasets. General outperforms other models. | Lower interpretabilit but provides feature importance. | When accuracy is the priority and you don't need to interpr the model easily. |

## Prediction for New Data (Random Forest): [310.44]

Assessing the SVR and Random Forest Models:

* + Calculated the R-Square value using the score method:
  + R2 shows how much of the variability in the target variable is explained by the features.
  + SVR model received an R Square score that demonstrates its effectiveness in fitting the training data.

## Random Forest Assessment

In the same way, the R2 score of 0.96 indicates outstanding performance, implying that the Random Forest model performs well with new, unseen data.





|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |





|  |  |
| --- | --- |
|  |  |

# Bibliography:

1. **Kaggle**: <https://www.kaggle.com/> - From where the dataset was selected.
2. **Scikit-learn**: <https://scikit-learn.org/> - To understand the tuning parameters of each function or model.
3. **Plotly**: <https://plotly.com/> -
4. **Matplotlib**: <https://matplotlib.org/> - used these libraries for data visualization.