UNIVERSITÀ
DEGLI STUDI
DI MILANO

Ridge Regression - Spotify Project

Statistical Methods for Machine Learning

Anno 23/24

Alessio Bernardini

23151A

Contents

1 Assignment **3**

2 Introduction **4**

3 Dataset Analysis **5**

4 Main Steps **7**
4.1 Import Libraries 7
4.2 Data Preprocessing 7
4.3 Data Visualization 7
4.4 Feature Engineering & Transformation 7
4.5 Feature Selection 8
4.6 Model Training 8
4.7 Performance Metrics 8

5 Regression **9**
5.1 Standard Linear Regression 9
5.2 Ridge Regression 10
5.3 Kernel Regression 12

6 Features Engineering **13**
6.1 Numerical Features 13
6.2 Categorical Features 14

7 Results Obtained **15**
7.1 Ridge Regression Metrics 16
7.2 Kernel Regression Metrics 18

8 Conclusions **20**

9 Declaration of Authenticity and Conformity **21**

1 Assignment

Download the Spotify Tracks Dataset and perform ridge regression to predict the tracks' popularity. Note that this dataset contains both numerical and categorical features. The student is thus required to follow these guidelines:

1. First, train the model using only the numerical features.
2. Second, appropriately handle the categorical features (for example, with one-hot encoding or other techniques) and use them together with the numerical ones to train the model.
3. In both cases, experiment with different training parameters.
4. Use 5-fold cross validation to compute your risk estimates.
5. Thoroughly discuss and compare the performance of the model.

The student is required to implement from scratch (without using libraries, such as Scikit-learn) the code for the ridge regression, while it is not mandatory to do so for the implementation of the 5-fold cross-validation.

Optional: Instead of regular ridge regression, implement kernel ridge regression using a Gaussian kernel.

2 Introduction

This report aims to analyze in detail the phases and methodologies for developing a predictive model in Python, aimed at estimating the popularity of music tracks. The dataset used for training the model is provided by the Spotify platform.

To achieve this goal, two distinct predictive models will be developed and compared: one based on Ridge Regression and the other on Kernel Regression. These approaches will be implemented and evaluated to determine which offers the best performance based on the available data.

In the next section, we will summarize the main steps taken to achieve the set objective. We will begin with an in-depth analysis of the dataset, examining in detail all the features present and their possible issues. This will allow us to better understand the nature of the data and identify any problems that could affect the model's performance.

After the dataset analysis, we will explore in detail the two chosen regression models. A theoretical description of Ridge Regression and Kernel Regression will be provided, explaining the underlying mathematics and their operation. We will also discuss the advantages and disadvantages of each approach, as well as the situations in which one might be preferable to the other.

The next section will be dedicated to data preprocessing. We will discuss the importance of encoding categorical variables and the challenges associated with numerical data, such as handling missing values and identifying outliers. This step is crucial to ensure that the model works with clean and accurate data, minimizing the possibility of errors.

Once preprocessing is complete, we will proceed with the implementation of the models. The techniques used to tune the hyperparameters of the regressors and the use of cross-validation techniques to evaluate model performance will be described. The results obtained will be presented, comparing the performance of the two predictive models in terms of accuracy, precision, and other relevant metrics.

Finally, we will discuss the results of the comparison, highlighting which models performed better and the reasons behind their performance. We will also offer suggestions for possible future improvements and further research. The main goal of this study is to develop an accurate and reliable predictive model for forecasting the popularity of music tracks on Spotify, addressing and resolving the most complex steps through the use of the two regression models.

3 Dataset Analysis

The dataset used contains the following features:

- **track_id**: The Spotify ID for the track.
- **artists**: The names of the artists who performed the track. If there are multiple artists, they are separated by a semicolon (;).
- **album_name**: The name of the album in which the track appears.
- **track_name**: The name of the track.
- **popularity**: The popularity of a track is a value between 0 and 100, with 100 being the highest popularity. Popularity is calculated by an algorithm primarily based on the total number of plays the track has had and how recent those plays are. In general, songs that are played a lot recently will have higher popularity compared to those played a lot in the past.
- **duration_ms**: The length of the track in milliseconds.
- **explicit**: Indicates whether the track contains explicit lyrics (true = yes; false = no or unknown).
- **danceability**: Describes how suitable a track is for dancing based on a combination of musical elements including tempo, rhythm stability, beat strength, and overall regularity. A value of 0.0 indicates low danceability, while 1.0 indicates high danceability.
- **energy**: A measure from 0.0 to 1.0 that represents a perception of intensity and activity. Typically, energetic tracks are fast, loud, and noisy.
- **key**: The key of the track, mapped to integers according to the standard pitch class notation. If no key is detected, the value is -1.
- **loudness**: The overall volume of a track in decibels (dB).
- **mode**: Indicates the modality (major or minor) of a track. 1 represents the major mode, while 0 represents the minor mode.
- **speechiness**: Detects the presence of spoken words in a track.
- **acousticness**: Measures the confidence with which a track is acoustic.
- **instrumentalness**: Predicts whether a track contains no vocals.
- **liveness**: Detects the presence of an audience in the recording.
- **valence**: A measure that describes the musical positiveness conveyed by a track.
- **tempo**: The estimated overall tempo of a track in beats per minute (BPM).
- **time_signature**: The estimated time signature.
- **track_genre**: The genre to which the track belongs.

As we can see, the dataset presents a variety of features that we can divide into two main categories: numerical features and categorical features.

Numerical features are those that assume a numerical value, such as popularity, duration, and many others. Categorical features, on the other hand, are those that have a textual value, such as the artist's name, the music genre, or the song title.

To use these features in a regression model, it is necessary to convert all features into numerical values. Therefore, we need to adopt strategies to encode categorical features.

Numerical features can also present challenges, such as the presence of outliers or skewed distributions. Outliers can distort the model results, while a skewed distribution can negatively impact the performance of the regression model.

To address these issues, we can apply data normalization or standardization techniques and use methods to identify and handle outliers.

In the following sections, we will discuss and delve into the strategies adopted to solve these problems, exploring how to effectively manipulate the features present in the dataset.

We will analyze various data preprocessing techniques, providing practical examples and evaluating the impact of each technique on model performance. This will allow us to obtain a clean and well-structured dataset, ready for training the Ridge and Kernel regression models.

4 Main Steps

In this section, we will analyze the steps necessary to achieve our goal of creating linear regression models using ridge and kernel regression, and discuss the various sections present in the code.

The main steps can be summarized into seven sections:

1. Import Libraries
2. Data Preprocessing
3. Data Visualization
4. Feature Engineering & Transformation
5. Feature Selection
6. Model Training
7. Performance Metrics

4.1 Import Libraries

In the first section, we will import all the necessary libraries for our code to function, such as numpy, pandas, matplotlib, seaborn, and other libraries specific to the implementation of the regression models.

4.2 Data Preprocessing

The second section is dedicated to the data preprocessing phase. In this phase, after displaying the dataset on the screen, we will clean the dataset by removing null values and duplicate rows. Subsequently, we will separate the features based on their nature (numerical or categorical) to handle them appropriately. Numerical features will be further divided into continuous and discrete.

4.3 Data Visualization

After cleaning the dataset, we will proceed to data visualization. This section will include a collection of charts to analyze the dataset composition. We will visualize individual features through histograms and count plots, and analyze correlations between features using scatter plots and heatmaps.

4.4 Feature Engineering & Transformation

This section is dedicated to feature transformation. First, we will identify and correct outliers, which can distort the predictions of regression models. One method to handle outliers is to remove them if they are few. Alternatively, we can transform the features to obtain a normal distribution. Additionally, we will encode categorical features using BaseNEncoder and apply Standard Scaling to numerical features to standardize the scale.

4.5 Feature Selection

In this section, we will split the dataset into two parts: training set and test set, with a proportion of 70% for training and 30% for testing. This division is crucial to evaluate the model's accuracy on unseen data during training.

4.6 Model Training

Once the data is prepared, we will proceed to define and train the models. We will train ridge regression and kernel regression models using various hyperparameters. The results will be compared graphically to determine which model offers the best performance.

4.7 Performance Metrics

In the final section, we will compare the performance of the trained models. We will use metrics such as mean squared error (MSE), R-squared (R^2), and other relevant metrics to evaluate the accuracy and reliability of the predictive models.

5 Regression

Regression is a fundamental statistical method used to explore and model the relationships between variables. It focuses on analyzing the dependence of a variable of interest, known as the dependent variable or response, on one or more independent variables, called predictors. In other words, regression helps to understand how and to what extent a variable can be influenced by one or more predictive variables.

The most common form of regression is linear regression, which assumes a linear relationship between the dependent variable and the independent variables.

5.1 Standard Linear Regression

In a multivariate linear regression model, the relationship between the dependent variable Y and the independent variables $X_1, X_2, ..., X_N$ is expressed by the equation:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_n X_n + \varepsilon$$

Where:

- Y represents the dependent variable (predicted output).
- β_0 is the constant term or intercept.
- $\beta_1, \beta_2, ..., \beta_n$ are the regression coefficients representing the effect of the independent variables $X_1, X_2, ..., X_n$ on Y.
- $X_1, X_2, ..., X_n$ are the independent variables (input).
- ε is the residual error.

The objective of linear regression is to find the coefficients $\beta_0, \beta_1, \beta_2, ..., \beta_N$ that minimize the sum of the squared residual errors, that is, the difference between the observed and predicted values by the model.

The ordinary least squares (OLS) method finds the coefficients β by minimizing the sum of the squared errors:

$$\min_{\beta} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2$$

where:

- y_i are the observed values,
- $\hat{y}_i$ are the values predicted by the model,
- N is the number of observations in the dataset.

However, when the independent variables are highly correlated with each other, the obtained regression coefficients can be very large and unstable, causing high variance in the coefficients and, consequently, poor generalization capability of the model.

5.2 Ridge Regression

Ridge Regression, also known as L2 regularization, is a statistical technique used to analyze data with multicollinearity, i.e., when the predictive variables are highly correlated with each other.

This methodology is an extension of Linear Regression that introduces a penalty term in the cost function, denoted by α. This regularization term has the effect of shrinking the model coefficients, thereby reducing the model variance and improving its generalization ability.

The objective thus becomes to minimize the function:

$$RSS_{\text{ridge}} = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{N} \beta_j^2$$

Where:

- λ is the penalty parameter (also known as the regularization parameter).
- β_j are the regression coefficients for $j = 1, 2, ..., N$.

The explanatory formula for Ridge Regression can be written as:

$$\hat{\beta}_{\text{ridge}} = \text{argmin}_\beta \left\{ \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{N} \beta_j^2 \right\}$$

The penalty term $\lambda \sum_{j=1}^{N} \beta_j^2$ helps to reduce the magnitude of the regression coefficients. This term penalizes larger coefficients, pushing them towards smaller values. The parameter λ controls the intensity of the penalty:

- If $\lambda = 0$, Ridge Regression reduces to Standard Linear Regression.
- If λ is very large, the coefficients will be very close to zero.

For a more detailed understanding, consider the design matrix X (where each row represents an observation and each column represents an independent variable) and the coefficient vector β. Ridge Regression seeks to solve:

$$\hat{\beta}_{\text{ridge}} = \text{argmin}_\beta \left\{ \|y - X\beta\|_2^2 + \lambda \|\beta\|_2^2 \right\}$$

Here, $\|y - X\beta\|_2^2$ represents the sum of squared errors and $\|\beta\|_2^2$ represents the sum of squared coefficients (L2 norm of the coefficients).

The solution for the coefficients in Ridge Regression can be obtained by solving:

$$\hat{\beta}_{\text{ridge}} = (X^T X + \lambda I)^{-1} X^T y$$

Where:

- $X^T X$ is the transpose of the design matrix X.
- I is the identity matrix.
- λ is the penalty parameter.

When λ is large, the penalty term becomes predominant and the regression coefficients are strongly penalized, tending towards zero. When λ is small, the penalty term has little effect and ridge regression approaches standard linear regression.

Advantages:

- **Reduction of Variance**: Ridge regression reduces the variance of the regression coefficients, making the model more stable and less sensitive to small variations in the data.
- **Control of Multicollinearity**: Ridge regression is effective in handling multicollinearity, as it prevents the coefficients from becoming too large when the variables are highly correlated with each other.
- **Prevention of Overfitting**: Ridge regression helps to prevent overfitting, especially when working with high-dimensional data.

5.3 Kernel Regression

Kernel Regression is a non-parametric regression technique that allows for estimating a regression function without assuming a specific model form. This flexible approach is particularly useful when dealing with complex data that do not follow simple linear or polynomial relationships.

Unlike traditional parametric methods, Kernel Regression does not impose rigid assumptions on the relationship between the independent and dependent variables. Instead, it uses a function called a kernel to assign weights to the data surrounding a point of interest, allowing for a more accurate estimate of the local regression function.

The goal of kernel regression is to estimate the value of the dependent variable y given a value of the independent variable x. The estimate is made using the values y of observations near x, where proximity is determined by a kernel function K.

A kernel function $K(x, x')$ measures the similarity or distance between two points x and x'. There are various functions to measure this, but we will use the Gaussian function:

$$K(x, x') = \exp\left(-\frac{(x - x')^2}{2h^2}\right)$$

Here, h is a smoothing parameter called the bandwidth, which controls how closely nearby points influence the estimate.

The kernel regression estimate at a point x is given by:

$$\hat{f}(x) = \frac{\sum_{i=1}^{n} K(x, x_i) y_i}{\sum_{i=1}^{n} K(x, x_i)}$$

Where:

- $\hat{f}(x)$ is the estimated value of the regression function at x.
- (x_i, y_i) are the data points (observations).
- $K(x, x_i)$ is the weight assigned to the i-th observation based on its distance from x.

Advantages:

- **Flexibility**: Does not require specifying a functional form for the relationship between x and y.
- **Adaptability**: Can capture complex and nonlinear relationships between variables.

Disadvantages:

- **Scalability**: Computationally intensive, especially with large datasets.
- **Choice of kernel and bandwidth**: Performance heavily depends on the choice of the kernel function and the bandwidth h.

6 Features Engineering

Feature management is a crucial step in training predictive models. This process requires careful analysis and manipulation of dataset features to achieve optimal results in machine learning models.

The assignment first requires training models using only the available numerical features in the dataset. Subsequently, categorical features must be included, which need to be encoded following a specific methodology.

6.1 Numerical Features

The numerical features in the dataset represent quantitative data and are divided into two different types: continuous and discrete.

The **continuous features** present in the dataset are: *popularity*, *duration_ms*, *danceability*, *energy*, *loudness*, *speechiness*, *acousticness*, *instrumentalness*, *liveness*, *valence*, *tempo*.

The **discrete features** present in the dataset are: *explicit*, *key*, *mode*, *time_signature*.

The two types of features require different treatments. For discrete features, it is necessary to ensure that they are represented by numerical values instead of boolean values like true or false. Regarding continuous features, it is essential to examine their distribution, especially when using linear regression models, for several reasons:

- **Ensure data normality**: Many regression techniques, including linear regression and ridge regression, rely on the assumption that the data follow a normal distribution. If this assumption is not met, the obtained results may be skewed and unreliable.
- **Identify outliers**: Extreme values that significantly deviate from most of the data can negatively impact the performance of the regression model. Analyzing the distribution of features allows for the identification of these outliers and deciding whether to remove, transform, or handle them differently.
- **Prevent collinearity issues**: Collinearity among continuous features can lead to stability and interpretation problems in regression models. Analyzing the distribution helps identify the presence of collinearity and adopt corrective measures.
- **Optimize model performance**: Understanding the distribution of features allows for selecting the right transformations and data preprocessing techniques to optimize the performance of the regression model.

First, we will conduct a graphical analysis of the feature distributions using histograms to identify those with skewed distributions. Next, numerical features presenting such skewness will be transformed using the following functions: Logarithmic, Reciprocal, Square-root, Exponential. The transformations we will adopt are: Square root for acousticness, and Exponential for speechiness, instrumentalness, liveness, and duration_ms.

6.2 Categorical Features

In addition to numerical features, our dataset includes categorical features, which represent non-numeric attributes such as names, categories, or labels. The categorical features in our dataset are: *track_id*, *artists*, *album_name*, *track_name*, *track_genre*.

For these categorical features to be effectively used in machine learning models, they need to be converted into a numerical form. Various techniques exist for converting these values. The technique we will use is called BaseNEncoder.

BaseNEncoder is a variant of the one-hot encoder, but instead of converting each category into a binary vector, it encodes each category into an integer value using Base-N encoding.

This method offers several advantages:

- **Dimensionality reduction**: Compared to the one-hot encoder, which creates a binary variable for each category, the BaseNEncoder converts the categories into a single numerical value. This significantly reduces the data dimensionality, especially in the presence of many categories.
- **Order preservation**: The BaseNEncoder assigns a unique numerical value to each category, implicitly maintaining the order of the categories. This can be useful in situations where the order of the categories has intrinsic meaning, such as in the case of ages or ordinal ratings.
- **Robustness to new categories**: Unlike the one-hot encoder, which requires a new binary variable for each new category encountered in the test data, the BaseNEncoder can handle new categories by assigning them a new numerical value. This makes the BaseNEncoder more robust in the presence of incomplete or evolving data over time.
- **Memory savings**: Since the BaseNEncoder converts the categories into numerical values instead of binary vectors, it requires less memory to store the data, especially when the number of categories is high.

7 Results Obtained

During our study, we conducted a detailed analysis by comparing different models and using a wide range of hyperparameters. For each model considered, we divided the training process into two distinct phases: one phase using only numerical features and another phase including all available features.

The statistics we analyzed to evaluate the performance of the models are as follows:

- **MSE (Mean Squared Error)**: MSE is a fundamental measure for evaluating the predictive capacity of models. It represents the mean of the squared errors, calculated as the difference between the values predicted by the model and the observed values. A lower MSE indicates better predictive capacity of the model, as the prediction errors are reduced. MSE formula:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

  Where:

  - n is the number of observations.

- **ABMSE (Adjusted Bias Mean Squared Error)**: ABMSE is a metric derived from MSE that accounts for the model's bias, which is the systematic tendency of the model to overestimate or underestimate values. This index is calculated by adding the square of the bias to the MSE. Considering the bias provides a more complete evaluation of the model's accuracy. Bias formula:

$$\text{Bias} = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)$$

  ABMSE formula:

$$\text{ABMSE} = \text{MSE} + \text{Bias}^2$$

- **R2_SCORE (Coefficient of Determination R^2)**: R^2 is a statistical measure that represents the proportion of the variance in the observed data explained by the model. It ranges from 0 to 1, where a value of 1 indicates that the model perfectly explains the variance in the observed data, while a value of 0 indicates that the model does not explain any variance. R^2 provides an evaluation of the goodness of fit of the model to the data. R^2 formula:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

  Where:

  - $\bar{y}$ is the mean of the observed values.

By analyzing these metrics, we were able to gain a deep understanding of the performance of the different models considered, allowing us to draw significant conclusions regarding their effectiveness and predictive capabilities.

7.1 Ridge Regression Metrics

Let's take a look at the results of our tests on regression models. We start with our Ridge Regressor model, comparing it with a simple linear regressor and the Ridge Regression models provided by scikit-learn.

	Model	MSE	ABMSE	R2_score
0	Sklearn Linear Model - Numerical Features	480.776970	18.258630	0.026337
1	Sklearn Ridge Model(0.9) - Numerical Features	480.777025	18.258638	0.026337
2	Sklearn Ridge Model (0.5) - Numerical Features	480.777000	18.258634	0.026337
3	Sklearn Ridge Model (0.3) - Numerical Features	480.776988	18.258632	0.026337
4	My Ridge Model (0.9) - Numerical Features	480.776980	18.258654	0.026337
5	My Ridge Model (0.5) - Numerical Features	480.776975	18.258643	0.026337
6	My Ridge Model (0.3) - Numerical Features	480.776973	18.258638	0.026337

Figure 1: Numerical Metrics - Ridge Regressor

	Model	MSE	ABMSE	R2_score
0	Sklearn Linear Model - All Features	407.732180	16.606043	0.174266
1	Sklearn Ridge Model(0.9) - All Features	407.732271	16.606064	0.174266
2	Sklearn Ridge Model (0.5) - All Features	407.732230	16.606055	0.174266
3	Sklearn Ridge Model (0.3) - All Features	407.732210	16.606050	0.174266
4	My Ridge Model (0.9) - All Features	407.732478	16.606044	0.174266
5	My Ridge Model (0.5) - All Features	407.732345	16.606044	0.174266
6	My Ridge Model (0.3) - All Features	407.732278	16.606044	0.174266

Figure 2: All Metrics - Ridge Regressor

The results obtained are as follows: the values of MSE, ABMSE, and R2_score are very similar among the various models. Therefore, our model produces results comparable to scikit-learn's standards.

We can observe the difference between models trained with only numerical features and those trained with all features. The former have worse values compared to the latter, so by introducing categorical variables, we have a more performant model. Even varying the alpha parameters in Ridge models, the performance remains stable. This leads us to think that, for this particular dataset, regularization has a negligible impact on overall performance.

Moving to 5-fold cross-validation, we obtained consistent results. All models maintain similar performance even in this configuration, confirming the robustness of their performance on different partitions of the training data.

	Model	MSE_CV	ABMSE_CV	R2_score_CV
0	Sklearn Linear Model - Numerical Features	484.596040	18.310879	0.026398
1	Sklearn Ridge Model(0.9) - Numerical Features	484.596036	18.310888	0.026398
2	Sklearn Ridge Model (0.5) - Numerical Features	484.596038	18.310884	0.026398
3	Sklearn Ridge Model (0.3) - Numerical Features	484.596038	18.310882	0.026398
4	My Ridge Model (0.9) - Numerical Features	484.596033	18.310910	0.026398
5	My Ridge Model (0.5) - Numerical Features	484.596036	18.310897	0.026398
6	My Ridge Model (0.3) - Numerical Features	484.596037	18.310890	0.026398

Figure 3: Numerical Metrics CV - Ridge Regressor

	Model	MSE_CV	ABMSE_CV	R2_score_CV
0	Sklearn Linear Model - All Features	411.235702	16.669741	0.17381
1	Sklearn Ridge Model(0.9) - All Features	411.235694	16.669762	0.17381
2	Sklearn Ridge Model (0.5) - All Features	411.235698	16.669753	0.17381
3	Sklearn Ridge Model (0.3) - All Features	411.235699	16.669748	0.17381
4	My Ridge Model (0.9) - All Features	411.235697	16.669738	0.17381
5	My Ridge Model (0.5) - All Features	411.235698	16.669739	0.17381
6	My Ridge Model (0.3) - All Features	411.235699	16.669740	0.17381

Figure 4: All Metrics CV - Ridge Regressor

Even in this case, we have an increase in performance by introducing categorical features into the dataset.

The addition of Ridge regularization does not seem to lead to significant improvements compared to basic linear regression. This could indicate that the problem does not suffer from overfitting and that the simplicity of the linear model is sufficient.

Finally, we noticed a slight improvement in performance when using all features compared to only numerical features. This suggests that the additional features contain valuable information that the model can exploit to make more accurate predictions.

7.2 Kernel Regression Metrics

Let's now analyze the results obtained with Kernel Regression models. Our Kernel Regression model will be compared with the model provided by sklearn, using different hyperparameters.

Sklearn Gaussian Kernel Regressor:

	Model	MSE	ABMSE	R2_score
0	Sklearn Kernel Ridge Model gamma=0.1, alpha=1....	470.364228	17.620581	-0.086215
1	Sklearn Kernel Ridge Model gamma=0.2, alpha=1....	510.853800	18.860525	-0.179718
2	Sklearn Kernel Ridge Model gamma=0.3, alpha=1....	602.332787	20.692983	-0.390971
3	Sklearn Kernel Ridge Model gamma=0.4, alpha=1....	741.629258	23.043122	-0.712649
4	Sklearn Kernel Ridge Model gamma=0.5, alpha=1....	892.072247	25.230046	-1.060068
5	Sklearn Kernel Ridge Model gamma=0.1, alpha=0....	479.615061	17.724622	-0.107578
6	Sklearn Kernel Ridge Model gamma=0.3, alpha=0....	596.107195	20.550013	-0.376594
7	Sklearn Kernel Ridge Model gamma=0.1, alpha=0....	504.151703	18.060958	-0.164241
8	Sklearn Kernel Ridge Model gamma=0.3, alpha=0....	591.657403	20.455138	-0.366318

Figure 5: Numerical Metrics - Sklearn Kernel Regressor

	Model	MSE	ABMSE	R2_score
0	Sklearn Kernel Ridge Model gamma=0.1, alpha=1....	1370.332293	30.620543	-2.164518
1	Sklearn Kernel Ridge Model gamma=0.2, alpha=1....	1371.808590	30.639638	-2.167927
2	Sklearn Kernel Ridge Model gamma=0.3, alpha=1....	1371.838920	30.639989	-2.167997
3	Sklearn Kernel Ridge Model gamma=0.4, alpha=1....	1371.839960	30.640000	-2.167999
4	Sklearn Kernel Ridge Model gamma=0.5, alpha=1....	1371.839998	30.640000	-2.167999
5	Sklearn Kernel Ridge Model gamma=0.1, alpha=0....	1370.166285	30.618383	-2.164134
6	Sklearn Kernel Ridge Model gamma=0.3, alpha=0....	1371.838801	30.639987	-2.167997
7	Sklearn Kernel Ridge Model gamma=0.1, alpha=0....	1369.835173	30.614064	-2.163370
8	Sklearn Kernel Ridge Model gamma=0.3, alpha=0....	1371.838561	30.639985	-2.167996

Figure 6: All Metrics - Sklearn Kernel Regressor

For sklearn models, a general trend is observed towards an increase in error metrics (MSE and ABMSE) and a decrease in R2_score as the gamma parameter increases, regardless of the alpha value. Additionally, the metrics analyzed on the complete dataset provide inaccurate estimates.

My Gaussian Kernel model:

	Model	MSE	ABMSE	R2_score
0	My Kernel Ridge Model gamma=0.1, alpha=1.0 - N...	470.364228	17.620581	-0.086215
1	My Kernel Ridge Model gamma=0.2, alpha=1.0 - N...	510.853800	18.860525	-0.179718
2	My Kernel Ridge Model gamma=0.3, alpha=1.0 - N...	602.332787	20.692983	-0.390971
3	My Kernel Ridge Model gamma=0.9, alpha=1.0 - N...	1242.453860	29.239963	-1.869207
4	My Kernel Ridge Model gamma=0.1, alpha=0.8 - N...	479.615061	17.724622	-0.107578
5	My Kernel Ridge Model gamma=0.3, alpha=0.8 - N...	596.107195	20.550013	-0.376594
6	My Kernel Ridge Model gamma=0.1, alpha=0.5 - N...	504.151703	18.060958	-0.164241
7	My Kernel Ridge Model gamma=0.3, alpha=0.5 - N...	591.657403	20.455138	-0.366318

Figure 7: Numerical Metrics - My Kernel Regressor

	Model	MSE	ABMSE	R2_score
0	My Kernel Ridge Model gamma=0.1, alpha=1.0 - A...	1370.332293	30.620543	-2.164518
1	My Kernel Ridge Model gamma=0.2, alpha=1.0 - A...	1371.808590	30.639638	-2.167927
2	My Kernel Ridge Model gamma=0.3, alpha=1.0 - A...	1371.838920	30.639989	-2.167997
3	My Kernel Ridge Model gamma=0.9, alpha=1.0 - A...	1371.840000	30.640000	-2.167999
4	My Kernel Ridge Model gamma=0.1, alpha=0.8 - A...	1370.166285	30.618383	-2.164134
5	My Kernel Ridge Model gamma=0.3, alpha=0.8 - A...	1371.838801	30.639987	-2.167997
6	My Kernel Ridge Model gamma=0.1, alpha=0.5 - A...	1369.835173	30.614064	-2.163370
7	My Kernel Ridge Model gamma=0.3, alpha=0.5 - A...	1371.838561	30.639985	-2.167996

Figure 8: All Metrics - My Kernel Regressor

Our model follows the same trend: as gamma increases, there is an increase in error and a decrease in the model's goodness of fit. Using different alpha values does not seem to significantly affect the performance of the models on this dataset.

In summary, all Kernel Ridge regression models, both custom and provided by Sklearn, show unsatisfactory performance on both datasets. This suggests that the Kernel Ridge model might not be the best choice for this data.

8 Conclusions

Comparing our Ridge Regression and Kernel Regression models, a significant difference in their performance and complexity emerges.

The Ridge Regression model is based on a linear penalization of the coefficients and is effective in handling multicollinearity among variables. It is a simple and interpretable model that can perform well when the relationships between features and the target are approximately linear. However, the relatively low R^2 scores indicate that the model might not optimally capture the complex relationships present in the data, though overall, it achieved good metrics.

On the other hand, the Kernel Regression model relies on the use of a kernel function to map the data into a higher-dimensional space, where the relationship between independent and dependent variables can be more complex. This model can be more flexible and capable of capturing non-linear relationships in the data.

However, the results obtained with the Kernel Regression model indicate overall lower performance compared to the Ridge Regression model, suggesting that the dataset might suffer from multicollinearity, or the relationships between features and the target are approximately linear, so we do not need a complex model like the Kernel Regressor that can capture complex relationships in the dataset.

9 Declaration of Authenticity and Conformity

I declare that this material, which I now submit for assessment, is entirely my/our own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of my/our work. I understand that plagiarism, collusion, and copying are grave and serious offences in the university and accept the penalties that would be imposed should I engage in plagiarism, collusion or copying. This assignment, or any part of it, has not been previously submitted by me/us or any other person for assessment on this or any other course of study.