Predicting Track Popularity from Spotify Library using Ridge Regression and Kernel Ridge Regression

Hiyab Negga

February 2024

Abstract

The project is based on the Spotify library and applies ridge regression extensively and briefly explores kernel ridge regression to predict the 'popularity' of a song. We explored numerical predictors and subsequently adding categorical predictors by using 'leaveoutone-encoding and hotone-encoding to measure and asses the performance of these models. Model performs better when quantitative and qualitative predictors were included. However, the encoding method for qualitative features and regularization strength affected model's performance as complexity in terms of features increased.

Contents

1 Introduction

In this day and age, platforms like Spotify provide users with a variety of audio content in the form of digital music and podcasts allowing users to stream, create playlists and discover new hits or popular and upcoming podcasts. The services offered have enabled Spotify to collect data on millions of users to extract insight about their listening habits, preference, user's interaction with the platform and so much more all in the hopes of improving the user's experience by granting more personalization and expanding the exposure to discover new audio.

As a result of the surge of data and the computational power increasing, platforms like Spotify benefit from creating recommendation systems allowing users to curate personalised playlists of music, share and see other users' playlists. This increases the engagement and overall satisfaction the application has delivered to its customer base. The platform's large audio library coupled with advanced algorithms is what solidifies the platform as a leader in the digital music streaming industry. It is thus imperative for platform's like Spotify to utilise their libraries based on popularity and other factors to enact the best recommendation systems to their users. The objective is to only focus on a facet that uses machine learning algorithms such as Ridge Regression to accurately predict the popularity of a track based on the features provided within the dataset provided in Kaggle.

The initial stages of the project will consist of conducting cleaning, inspecting missing values or any discrepancy that may be present within the data set. Moreover, we will be using visualisation techniques to get a sense of any interesting insight that may be useful for our analysis. Consecutively, a brief discussion on the modeling including the mathematical derivation to implement our algorithm is undertaken. Finally, we will comment on the results for evaluation and comparisons.

2 Dataset

The data set was downloaded from Kaggle, with 114,000 rows and 21 features, of which, 15 are quantitative and 5 are qualitative, see Appendix 9.1. We stored the dataset in our local drive for access. In the initial stages of inspecting the data we dropped an unwanted column called 'Unnamed: 0' and we dropped one missing value that was present in 'artists', 'album_name' and 'track_name'. Furthermore, we inspected for duplicates in the entire data set and found none. However, when inspecting duplicates by 'track_id', we found a total of 24,259 track that had been stored more than once. Their main difference emanates from the column 'track_genre', where a track with multiple genre was stored multiple times using different characteristics. The issue was handled by grouping by 'track_id' and creating a new column to store the genres of a track in a

colon separated column.

Subsequently we randomly split our entire data set with 80% consisting of the training set and 20% consisting of the test set. We used the training set for data exploration and visualisation.

3 Data Exploration and Visualisation

Before beginning our data analysis we conduct a preliminary quantitative and qualitative exploration in order to garner insight about how our data is distributed, and structured. Our findings indicate that 'duration_ms' has larger values with a larger range. In addition to 'durartion_ms' features such as 'speechiness', 'acousticness', 'instrumentalness' and 'liveness' indicate that the predictors are right skewed, see Appendix 9.2

A correlation plot indicates that the target variable and the numeric predictors have a weak linear relationship. However there exists a strong linear relationship between 'loudness' and 'energy' and a somewhat weak positive relationship between 'danceability' and 'valence'. On the other hand we notice a strong negative correlation between 'acoutsicness' and 'energy' and a somewhat 'negative' significant correlation between 'loudness' and 'acousticness' and a weaker negative correlation between 'loudness' and 'instrumentalness'.

In Table 3.1 we represent the unique values within the data set, for artists, we considered the unique artists on their own and not the combination of different artists as unique values when conducting the count. We also inspected the different 'track_genre' and the different popularity. In addition we saw how there is an imbalance with the explicitness of a track and how it behaves with respect to 'popularity'. For further, annotated description of data set please review this exploratory document.

Feature Name	**Unique Values Count**
Track Name	56891
Album Name	37843
Artists	25816
Track Genre	114

Table 3.1: Qualitative features unique values count.

4 Data Preprocessing

Exploratory analysis enabled us to make the appropriate choice for how to treat the variables. When considering handing the numerical features, it is was imperative to convert the 'duration_ms' which uses milliseconds as seconds to avoid having values that are overly exaggerated. Moreover, handling the right skewed features by conducting a $\log(x+1)$ transformation is crucial, this includes the new converted variable 'duration_s' and the rest of the features that were right skewed. Then we proceed by standardising the numerical features except for 'key', 'mode', 'time_signature' and 'explicit' as they are label encoded and binary predictors.

5 Modeling

Ridge regression, is a variant of a linear regression that is more revered for being resilient and its adaptable technique. Although basic linear regression is still useful, accounts of its performance on complicated and real worlds dataset has made it less preferable compared to ridge regression. This is usually true, especially with data sets that exhibit high correlation among predictors, also known as multicollinearity is a phenomenon in the data set, or overfitting which entails the model performing well on the training set but poorly on the test set hindering generalisation. In addition, ridge is also preferred in instances where there the data sets exhibits high dimensionality data with vast of predictors exceeding the number of training examples.

Ridge regression includes a bias in the model by regularising the magnitude of the coefficient to prevent them from becoming large, thereby mitigating the risk of, overfitting, outliers influences and multicollinearity. This capacity provides ridge regression to generalise models by trading-off bias for variance making it a preferred choice in a wide variety of applications. Similarly, kernel ridge regression further extends the ridge by undertaking a transformation, commonly referred as the kernel trick, from a linear algorithm to a non-linear one. This would enable us to model complex, non linear relationship in the data.

This project will enable an assessment of how the machine learning application performs on predicting 'popularity' of a track based on Spotify's music library. When modeling we took gradual steps, starting from only numerical[2] predictors and one boolean predictor, 'explicit' and subsequently adding the categorical predictors, namely, 'track_genre', 'artists', and 'album_name'. This enabled us to see how the model performed as more predictors were added. When handling the categorical predictors experimented with one hot encoding and leave one out target encoding.

[2]The numerical predictors include, duration_s, danceability, energy, key, loudness, mode, speechiness, acousticness, instrumentalness, liveliness, valence, tempo, time_signature

In the following sections, the project reviews the mathematical derivation of ridge regression and kernel ridge regression using the closed form (normal-equation). Relying on the custom ridge and kernel ridge regression developed we will be assessing how the different models perform on the different data sets containing different predictors. Since our data set is large in some instances we have resorted to sub-sampling techniques.

6 Mathematical Derivation

Ridge Regression is a technique that extends to linear regression by adding a regularisation technique to correct over fitting on training data in machine learning models. In addition to reducing errors caused by overfitting on training data, Ridge also corrects multicollinearity when conducting the regression analysis. Ridge adds a penalty term that is proportional to the square of the magnitude of the coefficients shrinking the coefficients towards zero by never exactly to zero. Consider we have a case where we have given N input data points x_n, n= 1,...,N, in D-dimensional space, and the corresponding outputs y_n. Converting our case into a matrix based formulation, we stack the input into a an $N \times D$ matrix $\mathbf{X}$ and the output data into a vector $\mathbf{y}$. Then the regularized leas-square problem consist in seeking $\mathbf{w} \in \mathbb{R}^{D\times 1}$ solving :

$$\min_{\mathbf{w}} ||y - \mathbf{X}\mathbf{w}||^2 + \lambda||w||^2 \tag{1}$$

The Ridge Regression constraints on the parameters w_j by adding the penalty term λ multiples by the squared norm of the weights. The penalty λ is a pre-chosen constant where we use nested cross-validation to select the value yielding the smallest cross-validation prediction error.[3] The equation below represents the solution Ridge in its matrix form.

$$\hat{\mathbf{w}} = (\mathbf{X^T}\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X^T}\mathbf{y} \tag{2}$$

Taking a step further we can obtain the kernel-based of version of Eq.(2), by transforming the data and the solution of the feature space.[4]

$$\min_{\phi(\mathbf{w})} \ ||y - \phi(\mathbf{X})\phi(\mathbf{w})||^2 + \lambda||\phi(\mathbf{w})||^2 \tag{3}$$

The notation $\phi(\mathbf{X})$ refers to the data matrix containing the transformed data, stacked as rows, $\phi(\mathbf{X} = [\phi(\mathbf{x}_1), \cdots, \phi(\mathbf{x}_n)]$

The solution $\phi(\mathbf{w})$ of this problem can be expresses as a linear combination of the training data in particular

[3]Ridge Regression: Applied Data Mining and Statistical Learning `https://online.stat.psu.edu/stat857/node/155/`

[4]Online Regression with Kernels by Steven Van Vaerenbergh and Ignacio Santamaría University of Cantabria March 2014 `https://gtas.unican.es/files/pub/ch21_online_regression_with_kernels.pdf`

$$\phi(\mathbf{w}) = \sum_{n=1}^{N} \alpha(n)\phi(\mathbf{x}_n) = \phi(\mathbf{X})^T\alpha \tag{4}$$

where $\alpha = [\alpha(1), \cdots, \alpha(N)]^T$. Substituting Eq.(4) into Eq.(5), and defining the matrix $\mathbf{K} = \phi(\mathbf{X})\phi(\mathbf{X})^T$, we obtain

$$\min_{\alpha} ||\mathbf{y} - \mathbf{K}\alpha||^2 + \lambda\alpha^T\mathbf{K}\alpha \tag{5}$$

The matrix $\mathbf{K}$ is denoted as the *kernel matrix*, and its element represents the inner products of in the feature space, calculates as kernels $k_{i,j} = \kappa(x_i, x_j)$. The final solution is given by:

$$\hat{\alpha} = (\mathbf{K} + \lambda\mathbf{I})^{-1}\mathbf{y} \tag{6}$$

We use the predicted weight to compute the predictive mean we get:

$$\hat{f}(\mathbf{x}) = \mathbf{w}^T\mathbf{x} = \sum_{i=1}^{N} \alpha_i \mathbf{x_i}^T\mathbf{x} = \sum_{i=1}^{N} \alpha_i \kappa(x_i, x) \tag{7}$$

For this project we have utilized the Gaussian kernel which which is defined as such[5]:

$$\kappa(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{||\mathbf{x} - \mathbf{x}||^2}{2\gamma}\right) \tag{8}$$

where γ represents the bandwidth. When executing the code we replaced the λ with α but the mathematical interpretation still holds.

7 Results and Evaluation

In this section we will discuss results and evaluations that rendered from running the models. We followed similar structure to assess and understand how each model and different data set perform. For all of the data set and the models, we started by running the with an $\alpha = 0.1$. We then assess the validation curve of the model inspecting the complexity of the model and the performance. This enables us to narrow our search to tune the regularization and bandwidth parameter for a five fold nested cross validation. Then we assess the performance of the model using a learning curve.

[5]Machine Learning A Probabilistic Perspective by Kevin P. Murphy

7.1 Ridge Regression (RR) Numerical

In the initial stage we included all the numerical predictors and ran the ridge regression. The results represented in Table 7.1 include the metrics, mean squares error (MSE), root mean squared error(RMSE) and R^2.

Metrics	**Results**
MSE	404.959
RMSE	20.123
R^2	0.317

Table 7.1: Results for $\alpha = 0.1$ run on numeric predictors

Result from the table indicate that on average prediction are approximately 20.12 units away from the actual values. In addition, 3.17% of the variance in the target variable can be predicted. It is evident that the model performs poorly because of its high error in prediction and its ability to explain a small percentage of the variability.

We will be using this as our base line to compare how different levels of regularization affect the model. To start out we will use two plots, specifically, the coefficient and error with respect to the regularization strength.

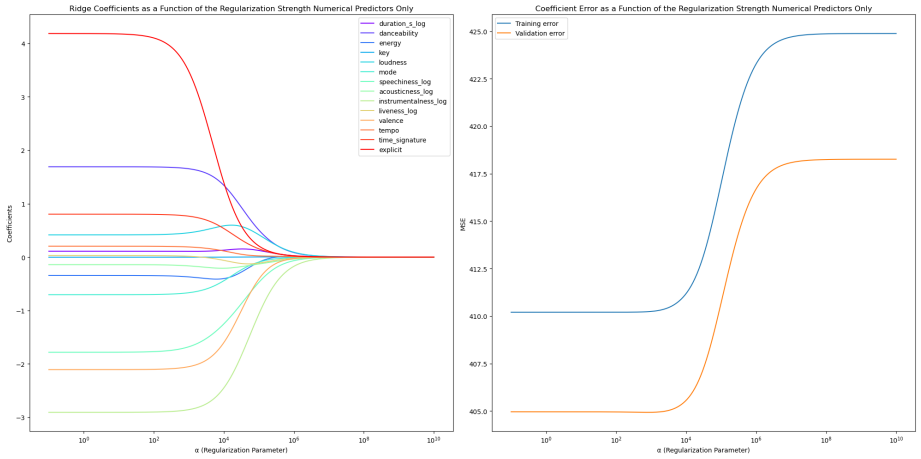


Figure 7.1: Ridge Coefficient & Error as a Function of Regularization Strength

The plot on the left in Figure 7.1 shows an increase in the regularization parameter shrinks the coefficients towards zero, gradually reducing the impacts of the features. From the plot, we can take away that 'explicit', 'duration_s_log

and 'time_signature' have positive coefficients for lower value of alpha. Furthermore, 'instrumentalness_log' and 'valence' have a negative coefficient followed by a steep incline towards zero as the value of alpha increases.

In the same figure from the plot on the left we can see that when alpha reaches around 10^4 mean squared error tends to increase. This enables use to narrow down our search when conducting a grid search to tune alpha using a five fold nested cross validation, in which the results for each fold are represented in the Appendix 9.3.1. So we conduct a five fold-nested cross validation to tune our regularization strength.

α	**MSE**	**RMSE**	$\mathbf{R^2}$
0.1	404.959149	20.123597	0.031744
63.89	404.954576	20.123483	0.031755
100.0	404.952220	20.123425	0.031760
1,000.0	404.933594	20.122962	0.031805
1,500.0	404.944973	20.123245	0.031778
10,000.0	405.564524	20.138633	0.030296

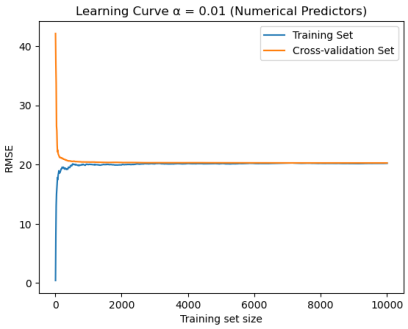
Table 7.2: Results for different values of α including the mean of 5-fold nested cross validated α highlighted in gray

In Table 7.2 we represent results from how the different model perform on different values of our regularization parameter, specifically ranging from 0.1 to 10,000. The mean of our five fold nested cross validation result is also included, which is $\alpha \approx 63.9$ and highlighted in grey. Overall it becomes clear that there is some stability in the metrics as their are no stark differences between the results.

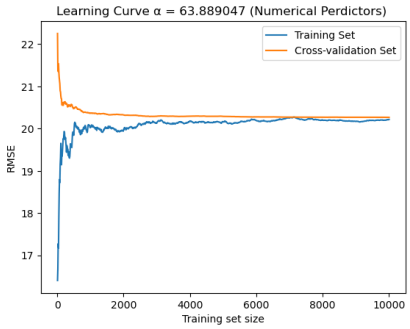
But building on this insight is critical, we further examine the learning curve to evaluate the model's performance on the size of the training set. We based our learning curve analysis on 10,000 training examples with a five fold cross-validation for $\alpha = 0.1$ and $\alpha = 63.889047$.

In Figure 7.2, the first interesting take away is the impact the regularization parameter has on the training error and the cross-validation error. For the graph with $\alpha = 0.1$, it is clear that the training error is lower and the cross-validation error is higher. This is the opposite for the learning curves with higher value of alpha. This is because the as regularization strengthens it restricts the complexity to fit the training data perfectly and enabling a better generalization on unseen data hence the higher training error and lower cross-validation error.

Generally, we are interested in the gap between the training set and the cross-validation set. Initially, we observe a large gap between the training set and the cross-validation set which typically indicates high variance or overfit-



(a) Learning Curve $\alpha = 0.1$ on 10,000 training examples



(b) Learning Curve $\alpha = 63.889047$ on 10,000

Figure 7.2: Learning Curves for Ridge Regression with Numerical Predictors

ting. As training examples increase we notice that the curves begin to plateau, eventually converging to a high error of approximately 20 RMSE. This suggests that adding more data will not improve the model's performance and we can conclude that the model is underfitted due to its high error on both the training and test set for both of the plots. This potentially means that the model is simplistic in nature or that the features included do not provide enough information to perform a good prediction, which is supported by the low value of the R^2.

For the ridge regression with numerical predictors adding more data or increasing the regularization and reducing the flexibility in terms of α will not resolve in an improvement of performance. So in the next stage we focus on adding complexity in terms of features.

7.2 Numerical & Categorical

In the previous section we saw that the model's performance is subpar as per the evaluation metrics. In this section we will be adding the complexity of the dataset by including the categorical predictors. We will be following the same structure, with the only difference attributed to handle approach of handling categorical predictors. In the first sub-section, we will resort to a leave-one-out-encoding and in the second sub-section we will be using a one-hot-encoding.

7.2.1 RR Leave-One-Out Encoding

Running the numerical and categorical predictors using $\alpha = 0.1$ gives us a much better result than our model with only numerical predictors. The MSE dropped from 404.59 to 149.60, RMSE also drops to 12.23 and the R^2 increases to 64%.

Overall the model with the categorical predictors treated using the leave-one-out encoding has a moderate fit to the data as shown in the table below.

Metrics	**Results**
MSE	149.60
RMSE	12.23
R^2	0.64

Table 7.3: Results for $\alpha = 0.1$ run on numeric & categorical predictors using leave-one-out encoding

Assessing the coefficient and coefficient error as the function of the regularization strength are plotted in Figure 7.3. In the right plot we can see that the categorical predictors, namely, 'track_genre', 'artists' and 'album_name' to be more significant decreasing steadily and slowly than their numerical counterparts. Furthermore, it is evident that the numerical drop in significance when compared to the ridge regression run on numerical predictors only.

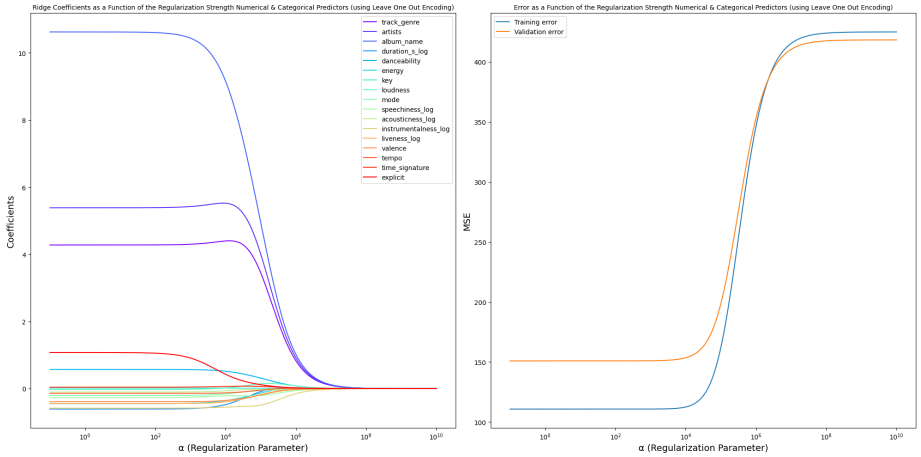


Figure 7.3: Coefficient & Error as a Function of the Regularization Strength (Numerical & Categorical Predictors using Leave-One-Out Encoding)

In the right plot we can see that the categorical predictors, namely, 'track_genre', 'artists' and 'album_name' to be more significant decreasing steadily and slowly than their numerical counterparts. Furthermore, it is evident that the numerical drop in significance when compared to the ridge regression run on numerical predictors only.

Assessing the left plot from Figure 7.3, in the initial stages with lower values

of alpha we witness that the model is over-fitting. This is due to the relatively low training error and the high validation error. As alpha increases, the penalty reduces the complexity up to a point where the learning error and the cross validation error converge at a large error rate which is a sign of underfitting.

Accordingly, we conducted the nested five fold cross-validation where the results for each fold is represented in Appendix 9.3.3. As we did before we will report the results of mean of the nested cross validation and other alpha values in order to assess the performance of the model with different regularization strength.

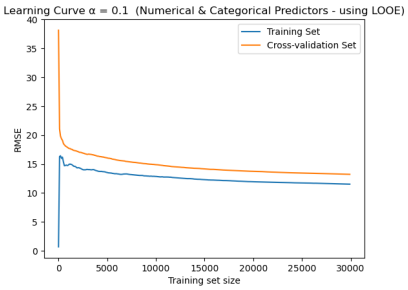
α	**MSE**	**RMSE**	$\mathbf{R^2}$
2.308417e-10	149.604903	12.231308	0.642295
0.1	149.604927	12.231309	0.642295
100	149.628541	12.232275	0.642239
1,000	149.859878	12.241727	0.641685
10,000	153.24927	12.379389	0.633581
100,000	202.748984	14.238995	0.515228

Table 7.4: Results for different values of α including the mean of 5-fold nested cross validated α highlighted in gray for ridge regression run on numerical & categorical predictors treated using leave one out encoding.

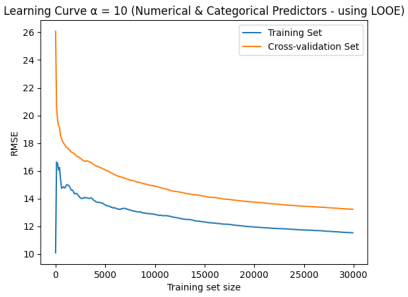
From Table 7.4 we can observe the changes to our metrics are small relative to the changes in the regularization parameter.This again shows that changing the flexibility of the model doesn't have a drastic impact on its performance. In addition, the model tends to do better on very small values of the regularization parameter see Appendix 9.3.3. Another, interesting observation is that the metrics for $\alpha = 2.308417e-10$ and $\alpha = 0.1$ are equivalent in terms of performance. Now let us turn how the model performs with respect to the training size using the learning curve. We again rely on our baseline model of $\alpha = 0.1$, which has a similar performance as our tuned regularization, and $\alpha = 10$ for comparison.

We based the learning curve2 in Figure 7.4 on 30,000 training examples. Key take away from the plots with $\alpha = 0.1$ is that as training examples are added the training error starts to stabilize a little below 15 RMSE. Where as the cross-validation set starts to stabilize a little above the training error. The gap indicates overfitting and with both curves plateauing we can conclude that adding more data will not improve the performance of the model. The plot behaves similar to the learning curve with $\alpha = 2.308417e-10$, see Appendix 9.3.4

For comparison we plotted the learning curve with $\alpha = 10$ and we can see that the gap between the training set and the cross validation set is wider.



(a) Learning Curve $\alpha = 0.1$ on 30,000 training examples



(b) Learning Curve $\alpha = 10$ on 30,000 training examples

Figure 7.4: Learning Curves for Ridge Regression with Numerical & Categorical Predictors using Leave-One-Out Encoding

Furthermore, we can see that the training error has higher error rates in the initial stages when our regularization parameter is higher. Moreover, the cross-validation error are lower when compared to the learning curve with smaller regularization because it enables the generalization on unseen data. However the model with the $\alpha = 10$ performs worse than the learning curve because the gap is wider and not shrinking.

7.2.2 RR One-Hot Encoding

In this section we apply one-hot-encoding on the categorical predictors. Instead of treating each unique occurrence as a column we just used our colon separated column of multi-genres and multi-artists as a column. Even though this has its own drawback because it exacerbates our problem by increasing the quantity of predictors and sparsity leading use to a problem known as the 'curse of dimensionality' it was necessary due to computational limitations. As we have done before we have run the ridge regression on our pre-processed features with the baseline $\alpha = 0.1$ and report the results in the Table 7.5.

Metrics	**Results**
MSE	77.94
RMSE	8.83
R^2	0.81

Table 7.5: Results for $\alpha = 0.1$ run on numeric & categorical predictors using one-hot-encoding

The results show a great promise when compared to the results we obtained

in the previous sections. With 81% of the variability in 'popularity' being explained by the model, with an average of 8.1 units of difference between the predicted and the actual values. These metrics indicates that the model is a stronger model compared to the previous ones. But let us gather more insight on the overall performance of the model.

Since we have more than 60,000 columns we will avoid assessing the coefficient as a function of the regularization strength. Instead we will be inspecting the validation curve to see how the model performs for different regularization parameter.

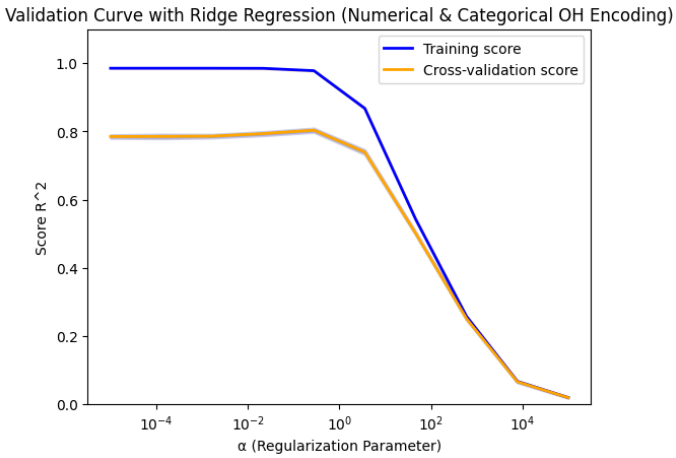


Figure 7.5: Validation Curve Numerical & Categorical Feature using One-Hot-Encoding

Validation curve for the ridge regression ran on numerical and categorical features treated with one hot encoding, represented in Figure 7.5 demonstrates for small values between 10^{-4} to around 10^{-1} the gap between the training score and cross-validation curve is wide indicating that there is overfitting, with the training score being higher than the cross-validation score. The gap starts to shrink between 10^{-1} and 10^{1} where there is peak in the cross-validation score, giving us the best trade off between bias and variance. As the regularization parameter increase, the score declines because data is over simplified making it difficult to capture underlying patterns in the data. This will help us narrow out our nested cross-validated to tune our parameter.

The mean result of our nested five fold cross validation returns $\alpha = 0.1$, which is represented in Table 7.6 The result of each of the fold is represented in the Appendix 9.3.5. We used the result and other values of alpha to show how the model reacts to different regularization. It seems evident that the model

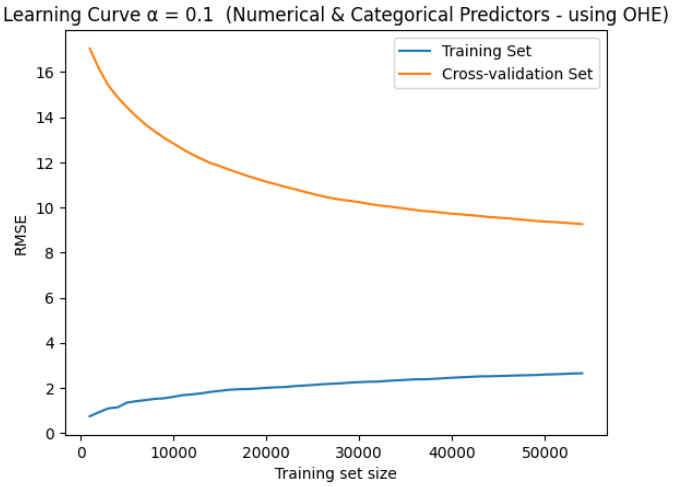


Figure 7.6: Learning Curve Numerical & Categorical Predictor using One-Hot Encoding on 55,000 training examples

with the hot-one-encoding is very sensitive to the regularization parameter, relative to the other models. An increase from $\alpha = 1$ to $\alpha = 100$, the model's MSE increases by 100 and its R^2 drops by 20%.

α	**MSE**	**RMSE**	$\mathbf{R^2}$
0.01	82.78119088	9.098416944	0.8020703797
0.1	78.0300961	8.833464558	0.8134302354
1	80.99441631	8.99968979	0.8063425532
10	132.117801	11.494251	0.684107
100	233.022499	15.265074	0.442844

Table 7.6: Results for different values of α including the mean of 5-fold nested cross validated α highlighted in gray for ridge regression run on numerical & categorical predictors treated using one-hot-encoding.

Learning curve in Figure 7.6 indicates a wide gap between the training set and the cross-validation set throughout all the training examples that are plotted. With the training error starting out with a low training error but then increases as more examples are added. In our plot we see it stabilize to a lower value of RMSE compared to our other models. Where as the cross-validation curve starting out with a high RMSE but then declines as more training examples are trained on the model. We do not see both the training set and cross validation set starting to converge and this indicates that the model is performing too well on the training error but failing to generalize well on unseen data.

In this case we can just say that the model might benefit from more data.

Nevertheless, we must not forget that when treating the model by using hot one encoding we have a high-dimensional and sparse space which means when adding more data points might be beneficial in combating against the sparsity and filling in the high-dimensional space. This is the ideal outcome, which may lead to the improvement of the model performance.

We have thoroughly inspected how the ridge regression performs on features that only include quantitative values and on both quantitative and qualitative using two treatment types. In the next section we explore how the kernel ridge regression.

7.3 Kernel Ridge Regression (KRR)

For the kernel ridge regression we decided to include all of the features using two types of treatment we are already familiar with for the categorical features. Since we had computational and memory limitation, we decided to subset our data set only using 20,000 and 3,500 data points, for training and testing, respectively. It is also important that we broke away from the previous structure. We did not tune any parameter in this section, rather we only conducted cross validation for different values of α while keeping γ constant.

7.3.1 KRR One-Hot Encoded Categorical Predictors

In order to see how the model performs on different regularization parameter we conducted leave-one-out cross validation for different values of α while keeping γ constant at 0.1. We were not able to apply the five fold cross validation due to resource limitations.

Alpha	**Gamma**	MSE_{test}	$\text{RMSE}_{\text{test}}$	R^2_{test}
0.001	0.1	234.04	15.29	0.48
0.01	0.1	233.66	15.28	0.48
0.1	0.1	273.72	16.54	0.40

Table 7.7: Results for $\alpha = 0.1$ & $\gamma = 0.1$ run on numeric & categorical predictors using one-hot encoding

Results indicate that the model is below average with the best performance attributed to when $\alpha = 0.01$ and $\gamma = 0.1$. When compared to the models in ridge regression that use different categorical treatments it is performing worse.

7.3.2 KRR Leave-One-Out Encoding

Our last model will be the kernel ridge regression by treating the categorical features using the leave-one-out encoding. In this section we have conducted a five fold cross validation using only the training set on various values of α in order to understand how the parameters impact the model's performance. In Table 10, we report the mean MSE of five fold cross validated, RMSE and R^2 for the results run on the test set using a leave-one-out validation.

Alpha	**Gamma**	Mean MSE_{CV}	MSE_{test}	$RMSE_{test}$	R^2_{test}
0.1	0.1	191.289278	182.249182	13.499970	0.571613
0.5	0.1	177.422287	171.359743	13.090445	0.597209
1.0	0.1	176.535575	170.914978	13.073446	0.598254
10.0	0.1	198.738074	191.735803	13.846870	0.549314

Table 7.8: Results for KRR $\alpha = 0.1$ & $\gamma = 0.1$ run on numeric & categorical predictors using one-hot-encoding.

Results indicate that the model is performing better than the model that utilizes one-hot-encoding as with the model being able to explain approximately 65% with an $\alpha = 0.5$ and $\gamma = 0.1$.

8 Conclusion

In this project we tried predict 'popularity' using the Spotify library using the ridge regression extensively and briefly investigated the kernel ridge regression. We gradually build our model by increasing the complexity in features and in the application of the treatment in categorical predictors. This showed us that the ridge regression ran with one-hot-encoded predictors preformed the best. Further more we saw how the regularization strength impacts the model which helped us tune the parameter.

Lastly we ran the kernel ridge regression on a sub-sampled data and found both models to be highly sensitive to the regularization parameter. As oppose to the ridge regression, the performance of the data treated with leave-one-out encoder performed better for the kernel ridge regression. There were resource limitations that did not enable us to push further especially in terms of doing more with the kernel ridge regression. Generally, inspecting other models such as the Random Forest and XGboost would also be interesting.

9 Appendix

9.1 Data Description

Variable Name	Feature Name Description	Data Type
track_id	Spotify ID for the track	categorical
artists	Artists' names who performed the track. If there is more than one artist, they are separated by a semicolon (;) .	categorical
album_name	The name of the album.	categorical
track_name	Name of the track	categorical
popularity	The popularity of a track is a value between 0 and 100, with 100 being the most popular. The popularity is calculated by the number of plays the track has and how recent the plays are.	integer
duration_ms	Track length in milliseconds	integer
explicit	True = yes if the track has explicit lyrics, false otherwise	boolean
danceability	How suitable the track is for dancing based on tempo, rhythm stability, beat strength, and overall regularity. Value ranges from 0.0 for least danceable to 1.0 for most danceable	continuous
energy	Measure from 0.0 to 1.0 representing perceptual measure of intensity and activity. Energetic tracks feel fast, loud and noisy.	numerical
key	Maps pitches using standard Pitch Class notation.	integer
loudness	Overall loudness of track measured in decibels (dB).	continuous
mode	Detects whether the track is major or minor.	integer
speechiness	Detects the presence of spoken words in the track. Value above 0.66 indicates a track made entirely of spoken words. Value between 0.33 and 0.66 describes tracks that may contain both spoken and musical elements, such as rap music.	continuous
acousticness	A confidence measure from 0.0 to 1.0, with 1.0 indicating that the track is acoustic.	continuous
instrumentalness	Predicts whether a track contains no vocals. Rap or spoken words are considered "vocal". The closer the value is to 1.0, the greater the likelihood that the track contains no vocal content.	continuous
liveliness	Detects the presence of an audience in the recording. Higher liveliness increases the probability that the track was performed live.	continuous
valence	Measures from 0.0 to 1.0 describing musical positiveness conveyed by a track.	
tempo	Overall estimated tempo of a track in beats per minute (BPM), which signifies the speed or pace for a given piece and derives directly from average beat duration.	continuous
time_signature	Estimated time signature is a notational convention to specify how many beats are in each bar. Ranges from 3 to 7.	integer
track_genre	Genre the track belongs to.	continuous

9.2 Data Exploration

9.2.1 Numerical Data Distribution

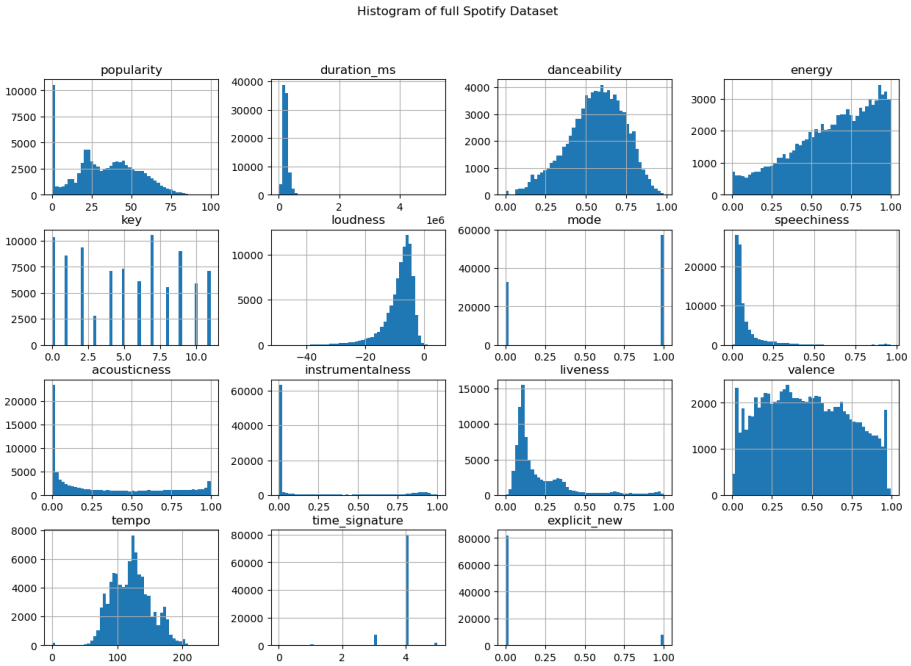


Figure 9.2.1: Numerical Predictors Distribution

9.3 Table & Figures from Results & Discussions

9.3.1 Nested Five Fold Cross Validations Numerical using

K-Fold	α	MSE	RMSE	R^2
1	43.29	402.006032	20.050088	0.034652
2	57.22	413.959490	20.345994	0.033216
3	75.65	414.522874	20.359835	0.035608
4	1000.0	407.146742	20.177878	0.034130
5	43.28.0	414.469144	20.123245	0.032387

Table 9.2: Results of five fold nested cross validation

9.3.2 Nested Five Fold Cross Validation on Numerical & Categorical Predictors using Leave-One-Out Encoding

K-Fold	α	MSE	RMSE	R^2
1	1.450829e-10	154.405636	12.426006	0.629222
2	1.450829e-10	157.574855	12.552882	0.631991
3	2.104904e-10	162.674350	12.754386	0.621536
4	2.104904e-10	154.295621	12.421579	0.633966
5	2.308417e-10	155.386378	12.465407	0.637237

Table 9.3: Results of five fold nested cross validation

9.3.3 Validation Curve For Ridge Regression on Numerical & Categorical Predictors using Leave-One-Out Encoding

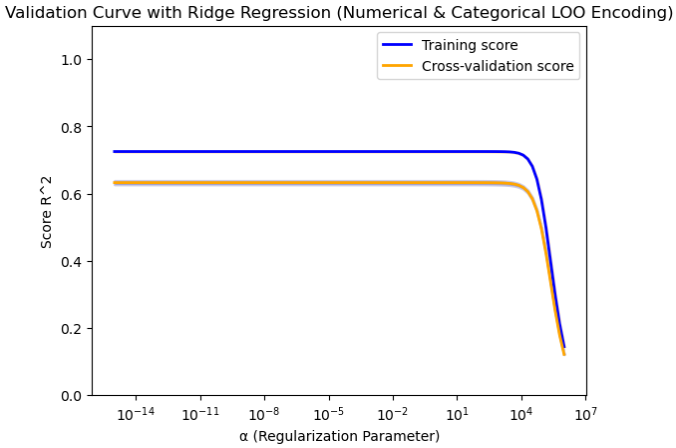


Figure 9.3.1: Validation Curve for Numerical & Categorical Predictors treated using Leave-One-Out Encoding

The validation curve for the ridge regression ran on numerical and categorical predictors using the leave one out encoding indicates that it is steady for very small values of regularization parameters. The constant R^2 from 10^{-14} until 10^4. This indicates that the ridge regression for the numerical and categorical predictors treated using leave-one-out encoding performs better on very minute values of alpha.

9.3.4 Learning Curve for Ridge Regression on Numerical & Categorical Predictors using Leave-One-Out Encoding, using $\alpha = 2.308417e-10$

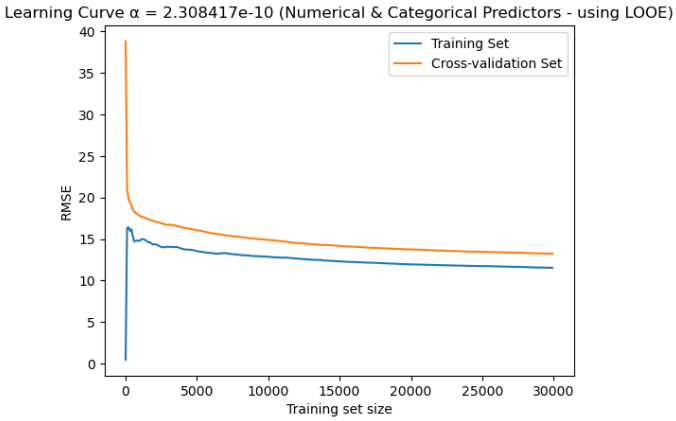


Figure 9.3.2: Validation Curve for Numerical & Categorical Predictors treated using Leave-One-Out Encoding

9.3.5 Nested Five Fold Cross Validation on Numerical & Categorical Predictors using One-Hot Encoding

K-Fold	α	**MSE**	**RMSE**	$\mathbf{R^2}$
1	0.1	85.707107	9.257813	0.794189
2	1.450829e-10	81.958364	9.053086	0.808590
3	0.1	88.385661	9.401365	0.794370
4	0.1	83.262446	9.124826	0.802477
5	0.1	83.946935	9.162256	0.804019

Table 9.4: Results of five fold nested cross validation on numerical and categorical predictors using One-Hot Encoding