Sri Sankeerth Koduru - G42612138

Assignment 1:

Question1:

Data Cleaning:

The following steps were taken to clean the data:

Rearranging the Dataframe:

After loading the data, I rearranged the columns to position the feature columns before the target column. This restructuring involved dropping the target column and subsequently appending the ‘streams’ column to the end of the dataframe.

Removing Columns:

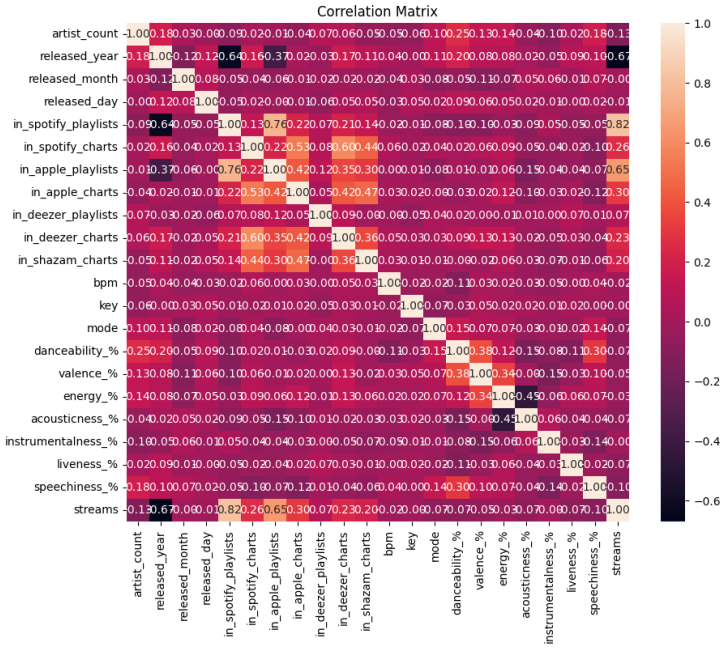
After loading and examining the dataset, I focused on the discrete variables. I observed that the 'track_name' and 'artist(s)_name' columns contained mostly unique values. This suggests that the 'track_name' and 'artist(s)_name' columns offer limited predictive power for understanding our target variable. Consequently, I removed these columns from the dataset.

Encoding Strings:

After the initial data cleaning, I looked at the string variables and converted them into numerical values using the LableEncoder. Furthermore, Though the target variable is discrete in nature it is classified as an object in the dataframe. To rectify this, I used the to_numeric function to convert the target variable from an object target variable into a continuous numeric target variable.

Feature Engineering:

After string encoding, I generated a correlation matrix and selected variables with correlations exceeding an absolute value of 0.1. The features 'in_spotify_playlists', 'in_apple_playlists', 'in_apple_charts', 'in_shazam_charts', 'in_spotify_charts', 'in_deezer_charts', 'speechiness_%', 'artist_count', and 'released_year' were used to build a new dataframe named df_feature_selection.



Dealing with Null Values:

After the feature engineering process, I conducted a data quality check and identified a single null value within the target variable. Given the low frequency of missing data and its presence in the target variable, I opted to remove the row containing the null value.

Dealing with Outliers:

After dealing with null values, I calculated Z-scores for each column to identify outliers exceeding 3 standard deviations from the mean. Using a threshold of 3 standard deviations, I identified and removed the outliers. Finally, I reviewed the cleaned dataset, confirming its size and inspecting the first few rows.

Normalized Data:

After dealing with outliers, I used the MinMaxScaler function to normalize all the features to prevent feature domination and ensure feature Interpretability.

Machine Learning Algorithm:

Data Split:

After normalizing the data, I slip the data into a training and testing dataset. The training set consists of 80% of the data (652 rows) and the testing set.

Machine Learning Algorithm:

The machine learning algorithm used for this problem was the Polynomial Regression Model.

Initialization:

The model's initialization allows for adjustable polynomial degrees, enabling us to tailor the complexity as needed. The default 'soft' value is set to 1; this ensures the model can execute standard logistic regression if no hyperparameter is explicitly provided during the class function call.

Optimization Algorithm:

This algorithm employs a customized form of normal equations for its optimization process. The key modification involves dynamically transforming the X_test based on a specified degree. For instance, setting the degree to 2 results in the addition of new features that are the squares of the original features. If the degree is set to 3, the algorithm adds both squared and cubed versions of the existing features.

After creating the new dataset, we used a modified normal equations algorithm to find the ideal weights (theta) for prediction. We adjusted the normal equation (i.e. (((X T X)^-1)*(X T y))) because polynomial regression can make the matrix (X T X) difficult to invert. To address this, we used the np.linalg.pinv function to calculate a pseudo-inverse. This allowed us to use the normal equation for efficient weight calculation.

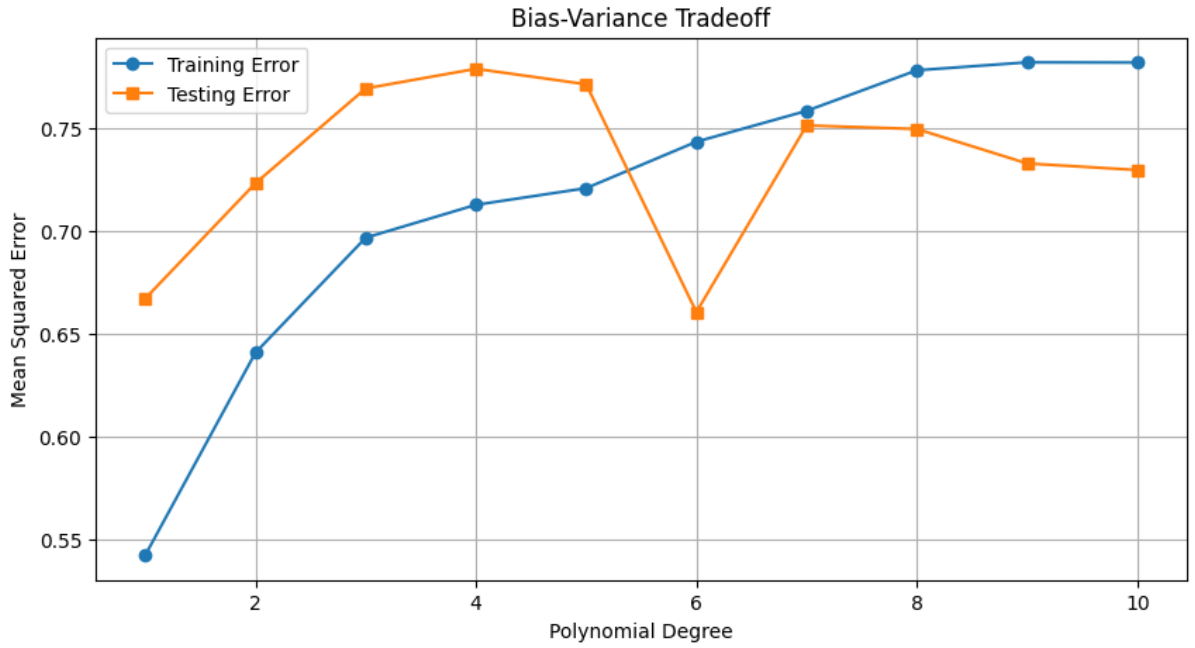
Prediction:

We can predict the y values by taking our test data (X_test) and multiplying it by our calculated theta values. This follows the formula f(x) = theta T X_test

Cost Function:

The cost function used to evaluate this algorithm is the R-squared. R-squared is a statistical measure that indicates how much better the model performs compared to simply using the average value for predictions. An R-squared of 0 indicates that the model is no better than the average, while an R-squared of 1 implies the model performs well above average and is a perfect fit for the data. Values between 0 and 1 show varying degrees of improvement over the average, with higher R-squared values demonstrating stronger model performance.

Variance Bias Tradeoff:



Underfitting:

Underfitting takes place when there is high bias in the model. High bias indicates that the model fails to capture the underlying patterns in the data. This indicates that the model is unable to predict the y values for both training and testing data. In the above image we can see that the model has underfit when using polynomial of 1 (Linear Regression) as it is performing poorly on both training and testing data. This indicates that a more complex model is needed to improve the predictive power.

Overfitting:

Overfitting takes place when there is high variance in the model. High variance indicates that the model not only captures the underlying pattern but also the noise in the training set. This indicates that the model predicts the y values well on the training data but poorly on the testing data. In the above case can see that the model has overfit when using polynomials of 6 or greater as it is performing well on both training data and poorly on the testing data. This indicates that a less complex model is needed to improve the predictive power.

Best Fit:

The training error is low, and the testing error is at its minimum value at a polynomial degree of 4. This indicates that a polynomial of the fourth degree captures the underlying patterns in the data well without overfitting the noise. The R-squared value of the test set is 0.7748 which suggests that the model explains 77.48% of the variance in the data.

Challenges Faced:

Data Cleaning:

The primary challenge I encountered was determining the optimal sequence for data cleaning procedures. My initial approach involved removing null values and outliers prior to feature engineering. However, this resulted in a significant loss of data. Since many of the null values and outliers resided within features ultimately omitted from the final model, I revised my strategy. The new process prioritized data encoding and feature engineering, followed by the removal of outliers and null values. Additionally, it was crucial to address null values in the target variable to improve the dataset's quality.

Machine Learning Model:

The primary challenge I encountered involved determining the optimal polynomial degree for achieving the best model fit.

Question 2

Part A:

GaussianNB: The optimal point for this model appears to be approximately 1500 training instances. At this point, the cross-validation score closely aligns with the training score. While 1500 instances yield the best results, the 750-instance mark is also viable since its cross-validation score remains relatively close to the training score. Despite slightly lower accuracy, the model trained with about 750 instances will still generalize well and demonstrate only marginal performance decline compared to higher instance counts. For large datasets, about the 750-instance mark may be preferable due to faster processing times.

SVC: Like the GaussianNB model the optimal point for this model appears to be approximately 1500 training instances. At this point, the cross-validation score closely aligns with the training score. While 1500 instances yield the best results, the 750-instance mark is also viable since its cross-validation score remains relatively close to the training score. Despite slightly lower accuracy, the model trained with about 750 instances will still generalize well and demonstrate only marginal performance decline compared to higher instance counts. For large datasets, about the 750-instance mark may be preferable due to faster processing times.

Part B:

Small Dataset:

GaussianNB: In the small dataset with 250 data points the model seems to be overfitting as the training score is high and the cross-validation score is low. This indicates that the model has high variance.

SVC: In the small dataset with 250 data points the model seems to be overfitting as the training score is high and the cross-validation score is low. This indicates that the model has high variance.

Large Dataset:

GaussianNB: In the large with 1000+ data points, the model seems to be optimal. This is indicated by the close similarity between training and cross-validation scores, suggesting strong generalization capabilities on unseen data.

SVC: In the large with 1000+ data points, the model seems to be optimal. This is indicated by the close similarity between training and cross-validation scores, suggesting strong generalization capabilities on unseen data.

Part C:

Underfitting (High Bias): A model suffering from high bias can benefit from increased complexity. This allows the model to better fit the underlying relationships in the data, enhancing its ability to generalize to unseen examples.

Overfitting (High Variance): A model suffering from high Variance can benefit from decreased complexity. This allows the model to better fit the underlying relationships in the data and not fit the noise in the data, enhancing its ability to generalize to unseen examples.

Part D:

The graphs suggest that increasing the size of the dataset enhances model generalization. When looking at the small dataset (250 data points), both models demonstrated overfitting. However, expanding the dataset significantly improved generalization. While gains became less pronounced after 750 data points, cross-validation scores continued to rise and converge with training scores. This

trend implies that predictive power and generalization improve with larger datasets, at least up to the tested threshold of 1500 data points.

Part E: