Machine Learning Project RepoRt

Spotify Streams Prediction

Maganti Gopala Krishna

**G23745216, gopalakrishna.maganti@gwu.edu**

**1.Introduction**

In this report, I outline my journey through a challenging machine learning project using a dataset from Spotify. The assignment involved several key stages: data preprocessing, algorithm implementation, model evaluation, and a detailed discussion of the results. My aim was to not only apply machine learning techniques but also to deepen my understanding of the underlying principles.

**2. Data Preprocessing**

Working with the Spotify dataset, my first task was to prepare the data for analysis and modeling. This dataset, comprising various features related to music tracks, required careful preprocessing.

**Handling Missing Values**: I noticed some missing values in columns like 'key' and 'in\_shazam\_charts'. For the 'key' column, I replaced missing values with the most common value (mode). For 'in\_shazam\_charts', I decided to fill in missing values with a unique strategy: the maximum value in the column plus one. This approach seemed appropriate given the nature of the data.

**Categorical Data Encoding**: Some features in the dataset were categorical, such as 'mode' and 'key'. I used Label Encoding to transform these into numerical format, a necessary step since the algorithms I planned to use work with numerical inputs. I also explored One-Hot Encoding but eventually decided not to include it in my final preprocessing pipeline.

**3. Machine Learning Algorithm Implementation**

For this project, I chose to implement Elastic Net Regression, a sophisticated algorithm that combines the properties of both Lasso and Ridge regression.

**Understanding Elastic Net**: Elastic Net has two main parameters that control its behavior: `l1\_ratio` for the balance between Lasso and Ridge regularization, and `alpha` for the overall strength of regularization. I fine-tuned these parameters to find a balance between underfitting and overfitting.

**Building the Model**: I initiated the model with specific values for `l1\_ratio`, `alpha`, `n\_iter` (number of iterations), and `learning\_rate`. During the `fit` method, I added an intercept term to the input features and used Xavier/Glorot initialization for setting up the initial weights. The weights were updated using gradient descent during each iteration.

**Prediction and Evaluation**: After training, I used the model to predict values on both the training and test sets. To evaluate the model's performance, I calculated the Root Mean Squared Error (RMSE) and R² score. These metrics provided me with a clear picture of how well the model was performing.

**4. Model Evaluation**

**Model Training and Prediction**

The Elastic Net Regression model was trained on the training set and then used to predict values on both the training set (`X\_train`) and the test set (`X\_test`).

**Evaluation Metrics Used**

- Root Mean Squared Error (RMSE): This metric was used to quantify the difference between the predicted values and the actual values. A lower RMSE indicates better model performance.

- R² Score: This metric reflects the proportion of variance in the dependent variable that is predictable from the independent variables. A higher R² score (closer to 1) suggests a better model fit.

**Evaluation Results**

The model's performance was evaluated on both training and testing datasets. The RMSE and R² scores were calculated, providing insights into the accuracy and predictability of the model.

**Evaluation Output:**

* RMSE: 0.11440353065922743
* Train RMSE 0.1099444459274312
* R^2: 0.727334747571351
* Train R^2: 0.7468976579379404

These results suggest that the model has a good fit, with relatively low errors and high predictability.

**5. Results and Discussion**

**Summary of Results**

The Elastic Net Regression model demonstrated good predictive performance, with reasonable RMSE and R² scores on both the training and test datasets. These results indicate a balanced model that neither overfits nor underfits significantly.

**- Results Overview:**

- Test RMSE: 0.114, indicating the model's predictions are, on average, about 0.129 units away from the actual values.

- Training RMSE: 0.109, showing slightly better performance on the training data.

- Test R²: 0.73, meaning about 73% of the variance in the dependent variable is predictable from the independent variables.

- Training R²: 0.75, a bit higher than the test R², suggesting good model fit without overfitting.

**A graph with red and green lines

Description automatically generated**

**Discussion**

- The RMSE and R² scores on the test set are close to those on the training set, which is a good sign of the model’s generalizability.

- The balance between bias and variance seems well-maintained, indicating effective regularization by the Elastic Net approach.

- One challenge I might face is selecting the right hyperparameters for the Elastic Net model; it requires careful tuning to balance the trade-off between bias and variance. Additionally, interpreting the learning curve correctly to diagnose and fix issues with model underfitting or overfitting can be complex and requires a deep understanding of the model's behavior with varying data sizes.

**Visualizations**

In addition to the numerical analysis, plots were generated to visually interpret the data and results. These include scatter plots for feature analysis and a heatmap for correlation analysis.

A graph of a graph

Description automatically generated

A graph of red and blue dots

Description automatically generated

PART-2

**A. Optimal Balance Between Bias and Variance**:

For the **GaussianNB** model, the optimal balance seems to be achieved around 500 to 750 training instances. This is where the training score starts to plateau and the cross-validation score begins to converge with the training score, indicating that adding more data doesn't significantly improve the model's ability to generalize.

For the **SVC** model, the optimal balance appears to be reached just after 250 training instances. Beyond this point, both scores remain stable and close to each other, suggesting that both bias and variance are relatively low.

**B. Regime of Operation:**

1. **Small dataset size (e.g., 250 data points):**

- **GaussianNB**: At 250 data points, the model is likely in a high variance regime, as there's a noticeable gap between the training score (high) and the cross-validation score (lower), indicating that the model is overfitting to the training data.

- **SVC**: At 250 data points, the model appears to be in the optimal regime since the training and cross-validation scores are both high and close together, indicating good generalization.

**b. Large dataset size (e.g., 1000+ data points):**

- **GaussianNB**: In the high data regime, the model still has a gap between the training and cross-validation scores but less so than at 250 data points, which may suggest a move towards the optimal regime, but with a persistent bias.

- **SVC**: The SVC model remains in the optimal regime, as both scores are high and converge, indicating that the model generalizes well even with more data.

**C. Modifying Model Complexity:**

- **High Bias Regime**: To address high bias, we can increase the model's complexity by adding more features, using a more sophisticated model, or tuning hyperparameters to allow the model to learn more complex patterns in the data.

**- High Variance Regime**: To address high variance, we can simplify the model, reduce the number of features, collect more training data, or use techniques like regularization to prevent the model from fitting the training data too closely, would not increase the complexity.

**D. Effect of Adding More Data:**

- **GaussianNB**: Adding more data might help the GaussianNB model to some extent, as the cross-validation score is increasing with more data. However, as the curve starts to plateau, the improvement will likely be marginal.

- **SVC**: Since the SVC model is already performing well with high scores and little gap between training and cross-validation, adding more data may not significantly improve performance. It's already in an optimal regime with low bias and low variance.

**E. Underfitting Model Plot:**

The curves would illustrate a clear need to increase the model's complexity to capture the underlying patterns in the data. A screen shot of a computer

Description automatically generated

This code sets up a Decision Tree Classifier with **max\_depth=1**, which is a simplistic model that doesn't capture much complexity of the data and is expected to underfit.