Predicting Billboard Hot 100 Ranks with ARIMA, SARIMAX, WLS, and LGBM

Ryan Kelley
Fordham Graduate School of Arts and Sciences
NYC, New York, USA
rkelley8@fordham.edu

Abstract—Billboard and its charts have become the industry standard for measuring commercial music success in the United States. The Billboard Hot 100, in particular, serves as a weekly benchmark of a song's popularity based on streaming, radio play, and sales data. This project aims to predict a song's future position on the Hot 100 chart, specifically forecasting its rank one to three weeks ahead. To achieve this, I applied a combination of time series models (ARIMA and SARIMAX) and machine learning techniques (Gradient Boosted Decision Trees) to a culmination of weekly charts from 2000-2021. Preliminary results indicate that AR(3)IMA(3) performed the best of all the time series models, working well for short-term stable trajectories. These models were difficult to interpret and sensitive to the noisy non-normally distributed chart data. Therefore, gradient boosting was a better option for capturing the nonlinear patterns of rapidly changing ranks, explaining 97% of the variance in the dataset. These predictive models offer insights into chart dynamics and could support decision-making in music marketing, promotion, and talent scouting by anticipating how songs will perform in the near future.

Link to GitHub Respository:
https://github.com/ryankelley8/Predicting-Billboard-Hot-100-with-ARIMA-SARIMAX-WLS-and-LGBM-Machine-Learning-Final-Project-

I. Introduction

Established in 1958, the 'Billboard Hot 100' quickly became the industry standard for analyzing the success of a song. While its scoring metrics have evolved over the years, the website currently ranks songs based on a combination of streaming activity, radio airplay, and sales data. Currently, streaming activity plays the most significant role in the position of a song on the charts, as it is the most popular method to access music. Seeing that this ranking is universally accepted as a benchmark metric for a song's success, the ability to predict its future ranking would prove useful to independent artists and record labels looking to allocate promotional funds.

In this project, I set out to use current chart placement to predict future ranks, estimating a song path through the chart. To do so, I built several time series models using ARIMA, SARIMAX, and iterations using weights based on the inverse of squared residuals. The original data set was quite simple, since it was a culmination of weekly charts from 1958 to 2021. It had information on the date, the title of the song, the artist, the rank of the previous week, the highest rank obtained at that point, and how long it has been on the chart. I was able to engineer several features using this initial data set. I made three lag features for rank, momentum features like rank change (rank_lag2 − rank_lag1) and rank acceleration (rank_lag1 − 2 · rank_lag2 + rank_lag3), artist and song popularity features, as well as one hot encoded features for season to be used in the SARIMAX model. Beyond predictive modeling, I conducted an exploratory data analysis that revealed interesting trends and a bias for previous high ranking artists.

Gaining a better understanding of the dynamics that influence the charts is critical for industry professionals aiming to navigate an environment where commercial success is notoriously difficult to achieve. Our findings offer an opportunity to better inform decision-making at various stages of production, from budget allocation to marketing strategies. Moreover, by identifying biases that may be embedded in the charts algorithm this research contributes to broader efforts advocating for a more equitable and transparent landscape.

II. Related Works

This section outlines relevant research that informed my project on predicting Billboard Hot 100 Ranks. I examined studies that apply machine learning and social science methodologies to investigate the relationship of past ranks to a song's success success, while analyzing any bias present. We focus on literature exploring various models and how they can be applied to rank predictions, comparing the outcomes to inform my project.

A. Predicting a Song's Path through the Billboard Hot 100

This project titled "Predicting a Song's Path through the Billboard Hot 100" by Cristian Cibils, Zachary Meza, and Greg Ramel focuses on forecasting a song's future positions on the Billboard Hot 100 chart using machine learning techniques. The authors utilized historical chart data and musical attributes such as mood, genre, and artist gender to train a Ridge regression model. Their approach involves predicting a song's next chart position based on its previous positions and features, and then iteratively using these predictions to forecast subsequent positions, effectively mapping the song's

trajectory on the chart. Their findings indicate that historical chart trends are strong indicators of future performance, and their model demonstrates the potential to accurately predict a song's progression on the Billboard Hot 100. This study strongly influenced my project, leading me to prioritize past rankings as key predictors of future performance. [1]

B. Predicting the Top and Bottom Ranks of Billboard Songs Using Machine Learning

The study titled "Predicting the Top and Bottom Ranks of Billboard Songs Using Machine Learning" by Vivek Datla and Abhinav Vishnu investigates whether the language used in song lyrics can help predict a song's success on the Billboard charts. The researchers collected lyrics from songs that appeared on the Billboard Hot 100 between 2001 and 2010. They analyzed these lyrics using computational linguistic techniques and trained a Support Vector Machine (SVM) classifier to distinguish between songs that reached the top 30 and those that fell into the bottom 30 of the chart. The model achieved a precision of 76%, suggesting that the emotional and linguistic content of lyrics can be a significant indicator of a song's commercial performance. Although their modeling approach differed from mine, I found this study particularly informative for its pre-processing techniques. It guided my thinking on how to prepare my own dataset and inspired the creation of additional features to enhance models based solely on historical chart data. [2]

C. ARIMA and Regression in Analytical Review: An Empirical Test

The article titled "ARIMA and Regression in Analytical Review: An Empirical Test" examines how statistical models can enhance the process of analytical review in accounting. Specifically, it compares four statistical approaches—including ARIMA (Auto-Regressive Integrated Moving Average) and regression models—with two simpler, naive methods to assess their effectiveness in evaluating monthly account data. The study aims to determine whether these advanced statistical techniques can provide more accurate and reliable insights during the analytical review process, ultimately aiding accountants in identifying anomalies or inconsistencies in financial data. Although this study did not focus on Billboard Hot 100 prediction specifically, it proved to be one of the most valuable references for my project and was revisited multiple times throughout its development. The authors demonstrated that ARIMA and regression models substantially outperformed simpler, naïve forecasting methods. These findings strongly influenced my decision to implement ARIMA time series models and compare their performance with a Gradient Boosted Decision Tree Regressor. [3]

D. Predicting and Composing a Top Ten Billboard Hot 100 Single with Descriptive Analytics and Classification

The article "Predicting the Billboard Hot 100: A Machine Learning Approach" discusses the application of machine learning techniques to forecast the performance of songs on the Billboard Hot 100 chart. The authors employed various models, including decision trees and support vector machines, to predict a song's peak position and longevity on the chart. They utilized features such as previous chart positions, release dates, and artist popularity to train their models. The study found that machine learning models could effectively predict chart performance, offering valuable insights for the music industry. This article was frequently referenced throughout the project for its valuable insights into modeling and practical applications. It utilized similar features and compared multiple models, while also examining how the predictions could be applied within the music industry to accurately inform artists about their songs' commercial potential. [4]

E. Predicting the Popularity of Songs - CS221

The study "Predicting the Popularity of Top 100 Billboard Songs" aimed to classify whether a song would make it onto the Billboard Hot 100 chart. The authors utilized two datasets: one from the Billboard API containing songs that made the Top 100 and another from Kaggle with songs that did not. They extracted features from song lyrics using the Genius API and audio features from the Spotify API. After extensive data cleaning and pre-processing, they employed machine learning models such as logistic regression and support vector machines to predict a song's success. The results indicated that using lyrics and audio features could effectively predict a song's likelihood of charting on the Billboard Hot 100. This project provided valuable ideas for enhancing my model through the use of APIs. The detailed preprocessing techniques were especially helpful in guiding how I prepared my own dataset, which also demanded thorough cleaning. Overall, this article was instrumental in both shaping my project plan and identifying ways to further develop it. [5]

III. Datasets

One primary dataset was used containing a culmination of all weekly chart entries from 1958 to 2021. This dataset was then cleaned and modified with engineered features for the purpose of modeling. These features included three lag ranks, rank acceleration, rank change, artist popularity metrics, song popularity metrics, and one hot encoded seasonal features. As previously stated several related works informed my preprocessing methods, which will be explored in the Methodology section. [2]

Public Dataset

- "Billboard 'The Hot 100' Songs": Collection of the "Billboard Hot 100" charts from 1958-2021. [6]
  - `Features: date, rank, song, artist, last-week, peak-rank, weeks-on-board.`
  - 330,000 entries and 24,620 unique songs

Engineered Datasets

- df_charts.csv

  - Features: date, rank, peak-rank, weeks-on-board, rank_lag1, rank_lag2, rank_lag3, rank_change, rank_accel, weeks_since_peak, is_peak_hit, artist_avg_rank, artist_best_rank, artist_chart_appearances, artist_unique_songs, song_avg_rank, song_best_rank, song_weeks_on_chart, season_Spring, season_Summer, season_Winter.
  - 111,000 entries and 8,189 unique songs
  - Years: 2000-2021
- df_charts_ARIMA.csv
  - Features: log_rank, weeks-on-board, weeks_since_peak, is_peak_hit, artist_avg_rank, song_avg_rank, song_best_rank, song_weeks_on_chart
  - It was important to cut down because too many exogenous features can poorly influence the model.
  - 111,000 entries and 8,189 unique songs
  - Years: 2000-2021
- df_charts_LGBM.csv
  - Possesses the same features as df_charts, however date, seasons, song popularity feature, and is_peak_hit (is a number one song) was removed for modeling.
  - 111,000 entries and 8,189 unique songs
  - Years: 2000-2021

IV. Methodology

Datasets: For this project, the primary publicly available dataset was instrumental in building the engineered datasets used for modeling and analysis. The initial dataset, "Billboard 'The Hot 100' Songs": Collection of the "Billboard Hot 100" charts from 1958-2021, provided typical chart information like date, song, artist, and rank. It also included peak-rank, which was the highest rank at that point, and last-week, which was the lag rank of the previous week. The column last-week initially had around 32,000 missing values, indicating missing information for the previous week rank. This made sense for debut songs, so to handle those I imputed the maximum rank of 100 where the weeks on board was one to signify entering the chart at the bottom. This left me with around 2,000 missing values, around 0.76% of my dataset, representing songs that either lacked information or re-entered the chart. These were removed to avoid using noisy data and because it was such a small portion of my dataset. After these features were cleaned they were analyzed in an exploratory data analysis to determine important feature interactions and the necessary feature engineering methods.

The outcome of the exploratory data analysis will be touched upon more in my Results section. I found a high correlation with rank between numerical features like peak-rank, last-week, and weeks-on-board (weeks-on-board having the lowest correlation). The correlation coefficients revealed these values would be very informative in feature engineering and modeling. A histogram depicting the distribution of ranks indicated my dataset was slightly left skewed, illustrating that high ranking songs tend to dominate the charts. This finding was reinforced by comparing the peak rank of songs that hit number one to all other songs on the chart, finding that number one songs completely dominate all other songs, converging slightly with each other around week forty.

Next, I used a function to extract season to be one hot encoded. When comparing average weeks-on-board across the years to season, some underlying patterns were revealed. Winter overall had the highest values, perhaps indicating the presence and long duration of Christmas music on the chart. Spring had the most volatile weeks-on-board possessing the overall highest and lowest yearly average. I figured this seasonal component would be useful to incorporated into my SARIMAX model. After this I created an Autocorrelation and Partial Autocorrelation plots to determine the best amount of rank lag to use in my modeling. The autocorrelation showed a steady decrease as lag increased, however, the partial autocorrelation was much stepper indicating that one to three weeks of lag was optimal for modeling. Finally, I conducted an augmented Dickey–Fuller test to check the stationarity of my data set. This was confirmed with an extremely low ADF statistic (lower than critical values of 1%, 5%, and 10%) and a p-value that was virtually zero. This would inform my modeling as high stationarity means the integrated parameter in ARIMA would be zero.

After cleaning and analyzing the initial dataset, I began engineering features for modeling.

- Lag Features: rank_lag1, rank_lag2, rank_lag3, weeks_since_peak (how many weeks since hitting it's highest known value), and is_peak_hit (is the peak rank number one)
- Momentum Features: rank_change (rank_lag2 − rank_lag1), rank_accel ($\text{rank\_lag1} - 2 \cdot \text{rank\_lag2} + \text{rank\_lag3}$)
- Artist Popularity Features: artist_avg_rank, artist_best_rank, artist_chart_appearances, artist_unique_songs
- Song Popularity Features: song_avg_rank, song_best_rank, song_weeks_on_chart
- Season Features: season_Winter, season_Spring, season_Summer

It is important to note that not all engineered features were used, and some initial features were removed for redundancy. I started with a correlation matrix to see if any correlations were too high and might lead to overfitting. I avoided using the song popularity features in my Gradient Boosted Model because they would likely cause data leakages, and this information is usually not available for a song debuting on

the chart. However, they were quite informative to ARIMA so I kept them for that model as it would help to inform a charts path that has already entered the chart. Additionally, I opted out of using is_peak_hit in the Gradient Boosted Decision Tree model for a similar reasoning. Features like last-week were replaced with rank_lag1 to avoid repetition. I went back and adjusted many formulas to ensure no column relied on the target variable rank. Overall, the feature selection process incorporated components to prevent any leakages or overfitting. Through careful feature engineering and selection, this initial dataset was modified to provide a rich foundation for modeling, incorporating various dimensions of rank history, artist popularity, and seasonal features. The synthesis of these features allowed me to explore trends in the Billboard Hot 100 ranks, while ensuring that the engineered features were robust and comprehensive.

Model Selection and Training: I will employ time-series machine learning models such as ARIMA, SARIMAX, and applying weights to these models based on the inverse of their square residuals. Additionally, I will utilize a Gradient Boosted Decision Tree Regressor, LightGBM, to compare against the time-series modeling. These models were selected for their ability to work with both time-series (ARIMA) and tabular (LightGBM) data, my data set having both. Model selection was also heavily informed by related works. The article ARIMA and Regression in Analytical Review: An Empirical Test by William Kinney Jr inspired me to use and compare time-series and regressor models because of their success when compared to naive counterparts [3]. Stationarity and the exogenous influential variables determined by correlation coefficients indicated that time-series models would be a good fit for this dataset. However, the time-series models lack interpretiability as summaries output many variables that change output drastically with small input changes. LightGBM makes up for this offering more interpretibility, while lacking a time-series component. By analyzing these two models I can compare the time-series and regressor methods to have a comprehensive analysis of Billboard rank prediction.

As previously stated, several features were engineered for these models. Features were selected based on two criterion: correlation with rank and potential for data leakages. I started by selecting features that had an absolute value correlation with rank of 0.3 or higher. Features like seasons, artist popularity metrics, and date had some of the lowest correlations. That being said, artist popularity metrics were much more obtainable than song popularity metrics (seeing that was a component I was trying to predict) so I opted to utilize artist features as opposed to song features in my LGBM model. Another feature, is_peak_hit, was also removed because knowing if a song will hit number one is not always possible, especially if the song debuts low and climbs to number one. It is also important to note that the lag features were removed from the exogenous features in the ARIMA models to avoid redundancy. After taking any leakages into a count and selecting features based on correlation I began applying parameter selection for my time-series models.

TABLE I
AUGMENTED DICKEY-FULLER (ADF) TEST RESULTS

Metric	Value
ADF Statistic	−57.187 189 174 432 916
p-value	0.000 000 000 000 000 000 000 0
Critical Values	
1%	−3.430 408 317 130 088
5%	−2.861 565 775 288 586 4
10%	−2.566 783 719 278 821 4

ARIMA takes three model order parameters (p,d,q) each corresponding to the AR, I, MA components of ARIMA. P refers to the Auto-Regressive portion informing the model on how many past values of the series are used to predict the current value. D refers to the Integrated portion informing the model on how many times the series is differenced to make it stationary. Q refers to the moving average component informing the model on how many past forecast errors the model uses to improve predictions. Since I found my dataset to be extremely stationary, as seen in Table 1 where the p-value was virtually zero (too small for Python to calculate) and the ADF is much smaller than all its critical values, I knew the d parameter could be set to zero since no differences were required to make it stationary. This discovery combined with the information from my Autocorrelation plots allowed me to Grid Search for the best parameters between AR(0)MA(0) and AR(3)MA(3) by comparing AIC values that balance model fit and complexity. I found the best model order was AR(3)I(0)MA(3). Additionally, this Grid Search provided me with the AIC value for the zero parameter ARIMA model AR(0)I(0)MA(0), which I used as a baseline to compare my time-series models.

After model order and feature selection for ARIMA, I ran two models a standard (3,0,3) ARIMA and a (3,0,3,4) SARIMAX to incorporate seasonality (ie. 4 corresponds to 4 seasons). After that I compared several different metrics including log likelihood, AIC, BIC, heteroskedacity, and P is greater than z value (indicates how much the variable contributes to random predictions). This allowed me to iteratively cut my exogenous variables down to tune the model. Ultimately I used these features: weeks-on-board, weeks_since_peak, is_peak_hit, artist_avg_rank,song_avg_rank, song_best_rank, song_weeks_on_chart. I decided to keep song popularity metrics and is_peak_hit for the time-series modeling because of their high autocorrelation coefficients. Additionally, it would inform the model on the decay of number one songs on the charts. The Heteroskedasticity value indicated that the models could benefit from the weighted inverse of squared residuals. Additionally, the non-normal distribution of my data set motivated me to use the log(rank) rather than rank to smooth out some of the noisy entries. Therefore, I adjusted the initial ARIMA and SARIMAX models to incorporate log(rank) and added two models that weighted based on the

inverse square residuals. This greatly improved the performance of my models, the best model obtaining an AIC that was around half of the baseline AIC.

The LightGBM model was much more straight forward and easy to interpret. I simply removed low correlated features and any that may cause any data leakages, like song popularity metrics. The model was then run using a 20-fold time-series split that was determined by diminishing returns. The R squared values, RMSE, and MAE were calculated for each fold. A scatter plot of Predictions v. True Values was created along side a residuals plot. Finally, I also utilized a confusion matrix to analyze my predictions.

V. Anticipated Challenges

Throughout the development of this project, we anticipated a range of technical and methodological challenges that required careful consideration. Comparing the time-series models proved to be the most difficult portion of the project. I had to balance convergence warnings and potential data leakages/overfitting, while maintaining inveritibility. Additionally, all the metrics being compared changed drastically with small input changes. This resulted in certain models outperforming others at different times. For example, when I had first implemented the WLS ARIMA and SARIMAX, it was the best performing model. However, after adjusting the models to predict log(rank) the standard ARIMA (3,0,3) became the best performing.

Overfitting was another main concern for my modeling. As previously stated, my feature engineering offered some potential for data leakages. Initially both the time-series and LGBM models exhibited overfitting. The primary cause were the momentum features utilizing the rank I was predicting. This motivated me to adjust rank_change to be based on two lag features. I then incorporated a third lag features so the equation rank_accel did not rely on rank. With these changes all models stopped exhibiting signs of overfitting.

The most significant change I saw from a challenge I encountered was when I applied the logistic transformation to rank to address heteroskedasticity, or non-constant variance in errors. The main cause for my heteroskedasticity value was the normal distribution assumption built in to ARIMA and SARIMAX on non-normally distributed data. Applying a logistic transformation improved the predictive power of models drastically. Furthermore, prior to this change, my WLS models were exhibiting some signs of overfitting, likely due to aggressive weighting. This logistic transformation solved this problem, drastically lowering R squared values and eliminating any indication of overfitting.

VI. Results and Discussion

A. Exploratory Data Analysis

An exploratory data analysis was conducted on the initial dataset for the years 2000-2021 to determine any important feature interactions that would inform feature engineering or modeling.

TABLE II
Correlation Coefficients with Rank

Variable	Correlation with Rank
Last-week	0.8943
Peak-rank	0.8349
Weeks-on-board	-0.3814

Table II shows the relationship between rank and the numerical features in the initial dataset. These high values suggest they are very informative, which heavily motivated my feature engineering.

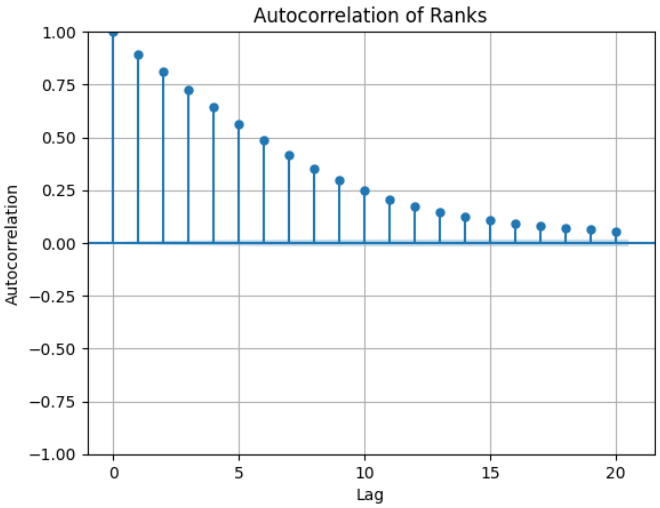

Fig. 1. Autocorrelation of Ranks for Various Lag Values

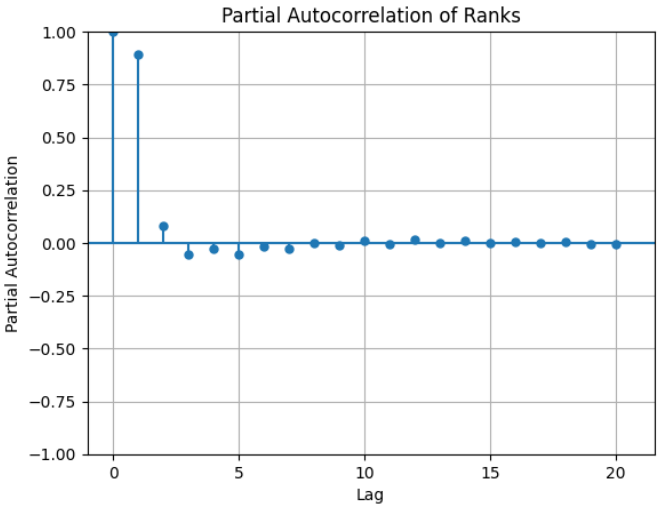

Fig. 2. Partial Autocorrelation of Ranks for Various Lag Values

Next, I plotted various lag values against autocorrelation and partial autocorrelation as seen in Figure 1 and Figure 2. It is clear that the autocorrelation decreases steadily with lag, having a slightly exponential curve. The partial autocorrelation, however, is much steeper, which provided me with a better understanding for the desired amount of lag. Anywhere from

lag1 to lag3 would be sufficient for this dataset, informing me on my model order selection and the number of lag ranks to manually add to my dataset for the LightGBM model.

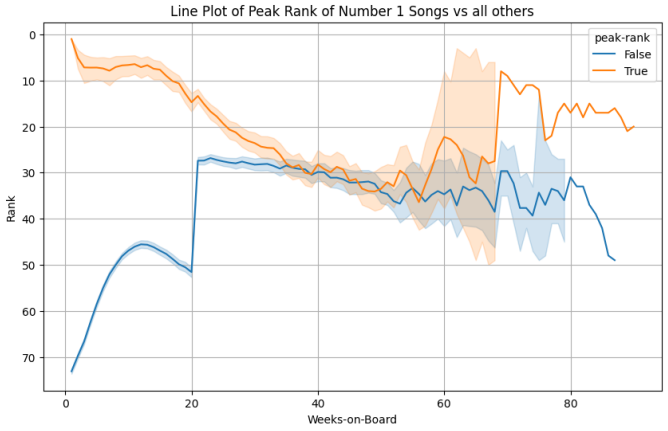

Fig. 3. Comparing Peak Rank of Songs that Hit Number 1 to all other Songs

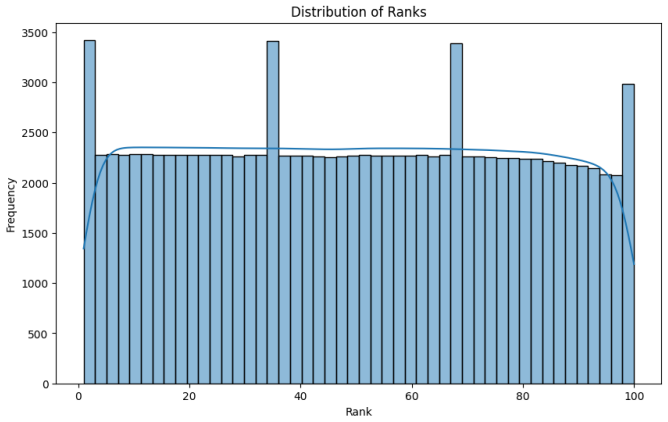

Fig. 4. Histogram of Rank Frequency

Figure 3 and Figure 4 reinforce the notion that higher ranking songs dominate the Billboard Hot 100. The line chart compares the peak rank of number one songs (orange) to all others (blue) as weeks on board increases. Overall, number one songs tend to last at higher ranks longer on the chart. However, there is a period of convergence around week 40 that is broken by week 60 with number one songs increasing in rank and all others decreasing. The histogram reinforces this notion that higher rank songs dominate the charts as it illustrates a slightly left-skew. I finished the exploratory data analysis by analyzing seasonal patterns. The patterns were minuscule and hard to observe, but could still be informative to the SARIMAX model.

B. ARIMA, SARIMAX, and WLS

As previously stated, four time series models were compared in this project. Each model was supplemented with a summary. I used four primary metrics to compare these models, since certain metrics were more informative about the data set rather than the models performance. These were log likelihood, AIC, BIC, and R squared.

TABLE III
MODEL COMPARISON: LOG-LIKELIHOOD, AIC, BIC, AND R-SQUARED

Model	Log Likelihood	AIC	BIC	R-squared
ARIMA(3,0,3)	-28366.7	56763.4	56907.8	-
SARIMAX(3,0,3)	-32261.1	64550.2	64685.0	-
WLS ARIMA	-435530.0	871100.0	871200.0	0.617
WLS SARIMAX	-480950.0	961900.0	962000.0	0.757

Table III illustrates a comprehensive overview of these models performances. As you can see, ARIMA and SARIMAX without weights of inverse squared residuals out performed the WLS models. This only became the case after the logistic transformation of rank. This transformation overall lowered AIC and BIC values and raise log likelihood for all models. However, the R squared value dropped from about 0.99 to 0.62 (ARIMA) and 0.75 (SARIMAX). The previous high R squared values and poor performance in Log Likelihood, AIC, and BIC indicated overfitting. I handled this by adjusting some of my engineered features, selecting only the most important exogenous features, and applying the logsitic transformation. OVerall, the lower R squared values were a positive change that adjusted for overfitting due to aggressive weighting.

The results reveal that among the unweighted models, ARIMA(3,0,3) offers the best balance between model fit and complexity, as indicated by its superior (less negative) log likelihood and lower AIC and BIC values compared to SARIMAX(3,0,3). While R-squared values are not reported for these likelihood-based models, the weighted least squares (WLS) variants do provide them. WLS SARIMAX achieves the highest R-squared (0.757), suggesting stronger explanatory power than WLS ARIMA (0.617), but this comes at the cost of substantially worse AIC and BIC scores, implying potential overfitting or added complexity. Overall, ARIMA(3,0,3) is preferable for model simplicity and interpretability, while WLS SARIMAX may be favored when maximizing explained variance is the primary objective.

C. Gradient Boosted Decision Tree

After time-series modeling, I analyzed my Gradient Boosted Decision Tree Model. I tuned the model several times to get the right combination of features. Additionally, I was mindful of any data leakages during this process opting to remove date, seasons, song popularity features, and is_peak_hit. Removing these features would improve the reliability of my model, decrease potential for overfitting, and offer better insights based solely on obtainable historical chart data. I analyzed RMSE, MAE, and R Squared value across a 20 fold time-series split to assess this model. Originally. I had observed signs of overfitting (R squared of 1 and MAE of 0), however adjustments to my momentum features and cutting down on features solved this problem. In tandem with these performance metrics, I visualized predicted versus actual values on a scatter plot, residuals across ranks, feature importance, and

a confusion matrix for the top 10 ranks, ranks 11-50, and all other ranks.

TABLE IV
AVERAGE PERFORMANCE METRICS ACROSS 20-FOLD CROSS-VALIDATION USING LIGHTGBM.

Metric	Value
RMSE	5.15
MAE	2.89
R^2 Score	0.967

LGBM is designed for tabular data so it works well with my engineered features. With an ensemble of decision trees that sequentially correct errors from previous trees the model is able to improve prediction accuracy. This was evident across the twenty folds as earlier iterations had R squared values of around 0.95-0.96, while later folds had values of 0.97-0.98. Analyzing these metrics in each fold allowed me to rule out any overfitting. As you can see from Table IV across the twenty folds the model estimated ranks correctly within 5 units, explaining about 97% of the variance in the model. This results were very promising and the interpretibility of this model was a huge improvement from the time-series models.

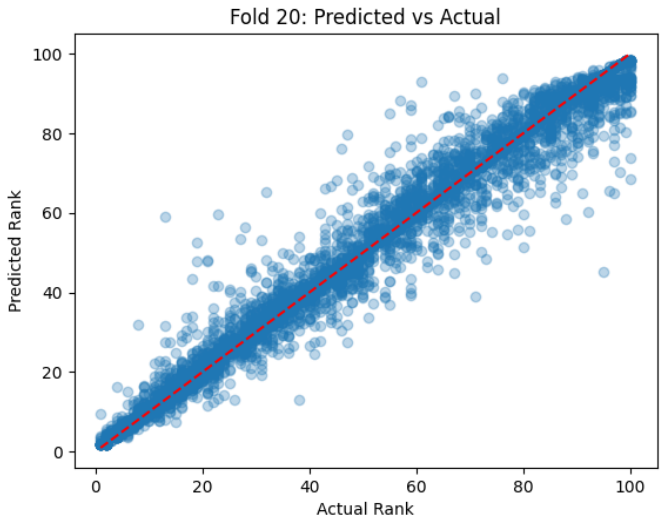

Fig. 5. Predicted v. Actual Values for LGBM

Figure 5 compares the LightGBM model's predictions against the true Billboard Hot 100 ranks. The points cluster somewhat tightly around the 45-degree line, indicating accurate predictions. However, there appears to be a slight underestimation occuring, especially at lower ranks. This is beneficial for the industry, and would avoid predicted ranks that are too high leading to misallocation of resources. There is still some indication that the model may be struggling with the least competitive chart positions because of this underestimation.

Figure 6) further analyzes these errors by plotting residuals (actual - predicted) against predicted ranks. This model shows residuals randomly scattered around zero with relatively no

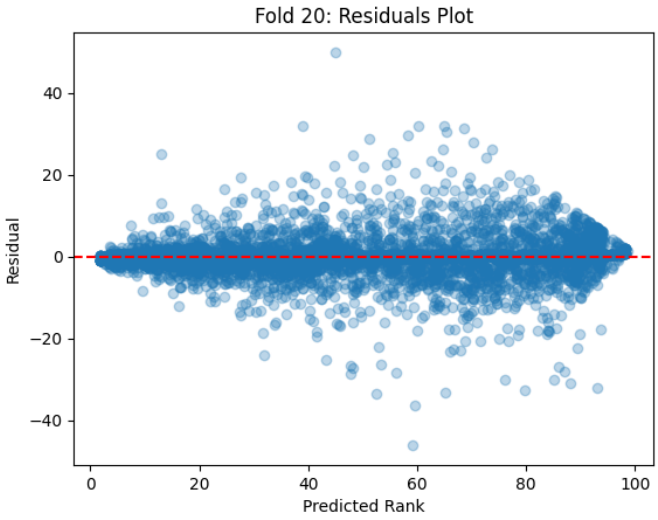

Fig. 6. Residuals Across Ranks for LGBM

patterns. However, residuals do fan out, having larger errors for higher ranks, suggesting heteroskedasticity or a bias for high ranking songs. While this may seem like an issue, having lower residuals for higher ranks is desired here because it is more important for higher ranks to be accurate. The patterns in these charts indicate that the model performs differently for songs at different popularity tiers, performing the best on the most popular songs. Together, these visuals highlight the model's strengths and areas for refinement, such as applying weights to lower ranks to better capture those trends if necessary.

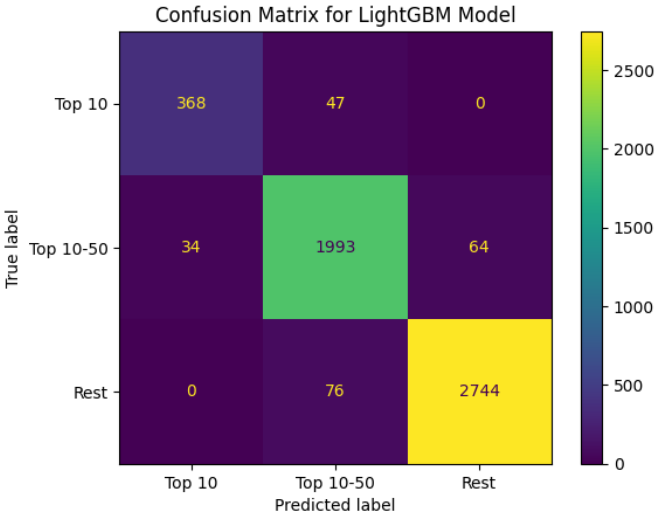

Fig. 7. Confusion Matrix for LGBM

The results in Figure 6) can also be observed in the Confusion Matrix depicted in Figure 7). As you can see high ranking songs only ever got mislabeled as songs in the top 50. When comparing to Figure 6) it appears that songs ranking 1-10 likely get mislabeled as the higher ranks that are

lower than fifty. The middle ranks are the most misclassified, indicating the model has a difficult time determining how mid-ranking songs will perform. The lower ranks resemble the Top 10, indicating that the model does well for extreme values but struggles with mid-level values. Figure 6) illustrates this showing residuals increasing around and after rank 50, and decreasing after 80. This is desired for this modeling project as the extremes of the chart can be very informative within the music industry.

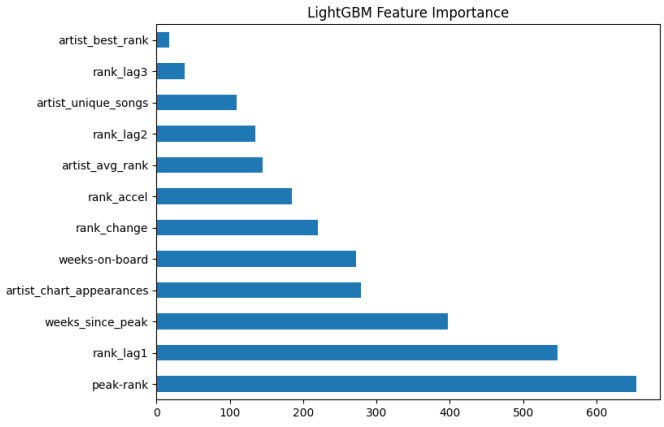

Fig. 8. Feature Importance for LGBM

Figure 8 reveals which factors most influence Billboard Hot 100 rank predictions. The top three most important features are peak_rank (the song's highest known chart position), rank_lag1 (the song's rank one week prior), and weeks_since_peak (the time since hitting the highest known rank value). This suggests that the rank a song enters at is highly correlated to where it will rank in the following week. Momentum features also had high importance values, indicating short-term trends and movement dynamics are critical drivers of chart performance. Other notable features include artist popularity features indicating that there is some bias to artists having entered the chart multiple times, as seen by the high importance value for artist_chart_appearances (how many songs they've had chart). Surprisingly, rank_lag2 and rank_lag3 have lower importance than rank_lag1, further emphasizing the importance of short term trends. This notion that longevity and all-time peak matter less than recent dynamics for short-term predictions aligns with Billboard's reputation for favoring recent popularity shifts over sustained performance. Overall, the model's reliance on lagged ranks and artist-centric metrics underscores the importance of incorporating temporal and artist-specific trends when forecasting chart behavior.

VII. Conclusion

This project demonstrated the effectiveness of time-series and machine learning models in predicting Billboard Hot 100 ranks, with the ARIMA(3,0,3) model providing the best balance of interpretability and predictive power among time-series approaches, while LightGBM achieved remarkable accuracy (R squared = 0.967) by leveraging engineered features like lagged ranks, momentum metrics, and artist popularity. The analysis revealed that short-term trends dominate chart movements, as evidenced by the critical importance of momentum features such as rank_accel and rank_lag1, which outweighed the influence of long-term historical data. Artist-related features like artist_best_rank and artist_chart_appearances highlighted systemic advantages for established artists, even as the model prioritized recent performance trends. While ARIMA/SARIMAX models effectively captured temporal dependencies, they faced challenges with noise and interpretability, whereas LightGBM delivered robust predictions but required careful feature engineering to prevent overfitting. These findings have significant implications for the music industry, suggesting that labels could optimize promotional strategies by focusing on songs with upward momentum and that real-time adjustments based on weekly streaming and radio data may be more impactful than long-term planning. The project also uncovered inherent biases in the chart system that favor established artists, pointing to opportunities for more equitable evaluation metrics. Future research could enhance predictive power by incorporating external data like social media trends or streaming patterns, experimenting with hybrid models that combine time-series and machine learning strengths, and addressing systemic biases in feature selection. Ultimately, this work bridges statistical rigor with practical applications, offering valuable tools for navigating the volatile music industry while acknowledging that no model can fully capture the cultural unpredictability of hit songs.

References

[1] C. Cibils, Z. Meza, and G. Ramel, "Predicting a song's path through the billboard hot 100," https://cs229.stanford.edu/proj2015/012_report.pdf, 2015, cS229: Machine Learning Final Project, Stanford University.

[2] V. Datla and A. Vishnu, "Predicting the top and bottom ranks of billboard songs using machine learning," *arXiv preprint arXiv:1512.01283*, 2015. [Online]. Available: https://arxiv.org/abs/1512.01283

[3] J. William R. Kinney, "Arima and regression in analytical review: An empirical test," *The Accounting Review*, vol. 61, no. 1, pp. 28–41, 1986. [Online]. Available: https://www.jstor.org/stable/245725

[4] A. Datla and S. Ghosh, "Predicting the billboard hot 100: A machine learning approach," 2015. [Online]. Available: https://cs229.stanford.edu/proj2015/012_report.pdf

[5] M. Isogawa, S. Masling, and M. Pan, "Predicting the popularity of top 100 billboard songs," 2019. [Online]. Available: https://www.maikaisogawa.com/wp-content/uploads/2019/07/CS_221_Predicting_the_Popularity_of_Top_100_Billboard_Songs.pdf

[6] D. Dave, "Billboard the hot 100 songs," https://www.kaggle.com/datasets/dhruvildave/billboard-the-hot-100-songs, 2021, accessed: May 15, 2025.