Stylistic analysis of Hebrew song lyrics

Dean Amar & Meital Yeruham

The Goal:

This project aims to analyze the stylistic elements of Hebrew song lyrics. The **motivation** stems from a controversial statement made by singer Yehoram Gaon in 2021, who criticized "oriental" (Mizrahi) music for its alleged poor and ungrammatical language. This project seeks to explore stylistic differences across various groups of songs, focusing on aspects such as vocabulary richness, syntactic complexity, and thematic diversity.

The Dataset:

The project utilizes a corpus of nearly 15,000 Hebrew song lyrics, downloaded from Kaggle. For each song, additional features such as the performer's year of birth, music style, and song release year were manually associated.
Moreover, we created a module that extracts additional features that will help us analyze the songs. These features provide a comprehensive analysis of the song's lyrics, covering lexical, syntactic, semantic, and sentimental aspects.
We added an appendix that includes information about the features.

Code explanation:

We divided the project into 3 parts.

1. Features extraction – Mostly numerical features.
2. Features Analyzing & Visualization – Using the features we extracted, we tried to prove or disprove hypothesis that set down before the start of the project, activating ML & DL methods in addition to data visualization algorithms.
3. Conclusions.

- **Feature extraction:**

  We created 3 classes – Class Artist, Class Song & Class DataFetcher.

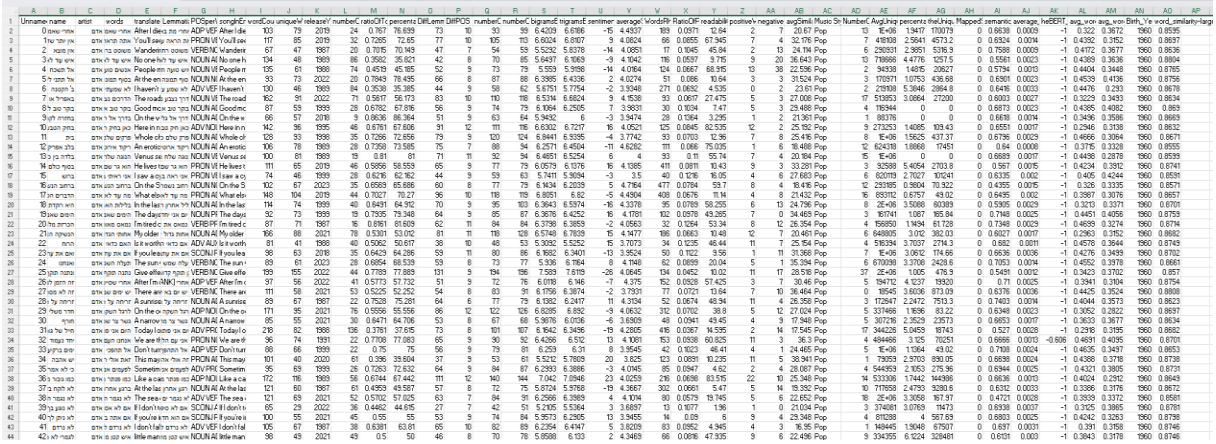
  Class Artist - Holds the artist properties including a dictionary of songs that corresponds to:
  Key – Song's name and its value is an instance of a class Song.

  Class Song - holds each song's properties and new features we extracted from the lyrics.

  Class DataFetcher – Is the class that uses both classes, Artist and Song for the new features extraction, it utilizes Modules we created such that each one focuses on the extraction of subset of features like sentiment features, lexical features and more. We used some pretrained models such as DictaBERT-il, Word2Vec using GloVe, WordFreq, Afinn. Each module produces an output that we process according to our needs. Moreover, we used some API's such spotify & Google translators to help us extract some data on the songs, for example, the release year and the lyrics in

English. We used the translated Lyrics for the extraction of features that aren't influenced by the lexical structure of the song – that changes when moving to a different language.

After the DataFetcher activation, we got a new dataset that includes some numerical features that explain the song's lyrics with different approaches.



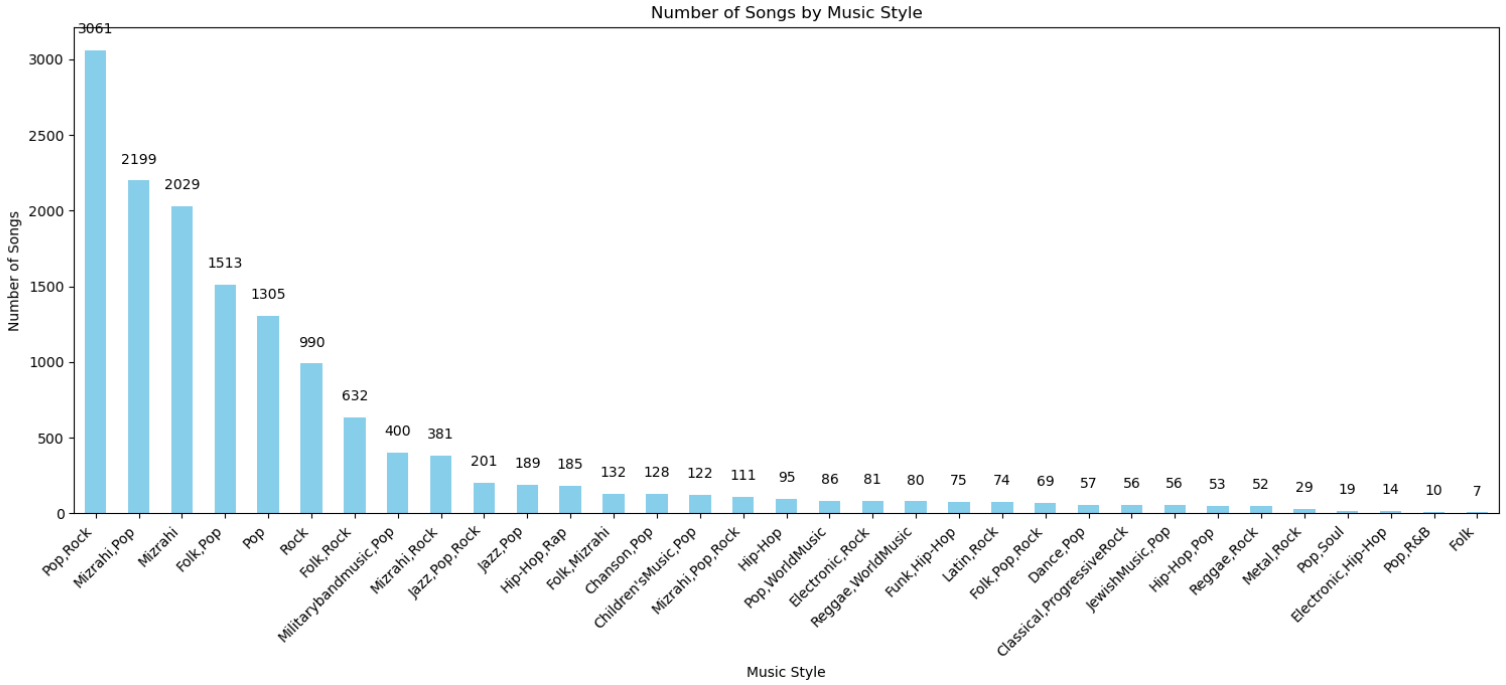
- **<u>Feature Analysis & Visualizations:</u>**

After extracting the new features and assembling the numerical dataset, we created a file called analyzer that includes the main class called FeatureAnalyzer.
It uses the dataset above and tags it according to our needs (By the song styles, year of release, born year of the artist, and artists)
This class uses methods for dimension reduction to provide us with the understanding of the best features that separates the songs by our requirements for each hypothesis and can visualize the data accordingly.

Another file named as classifier, including the Classifier class which activates different ML & DL methods for classification on our data based on the current needs. Meaning I can sample different subset of the data to validate or dis-validate our hypothesises.

Important detail: It was super hard to extract the song's style for each song due to lack of APIs capable for that. So, we attached to each song the artist's main music style. This way we got outliers of each style that doesn't fit good enough the style class. To overcome this issue, we batched each group into batches with different sizes and processed the data this way

Overlook on the Data:

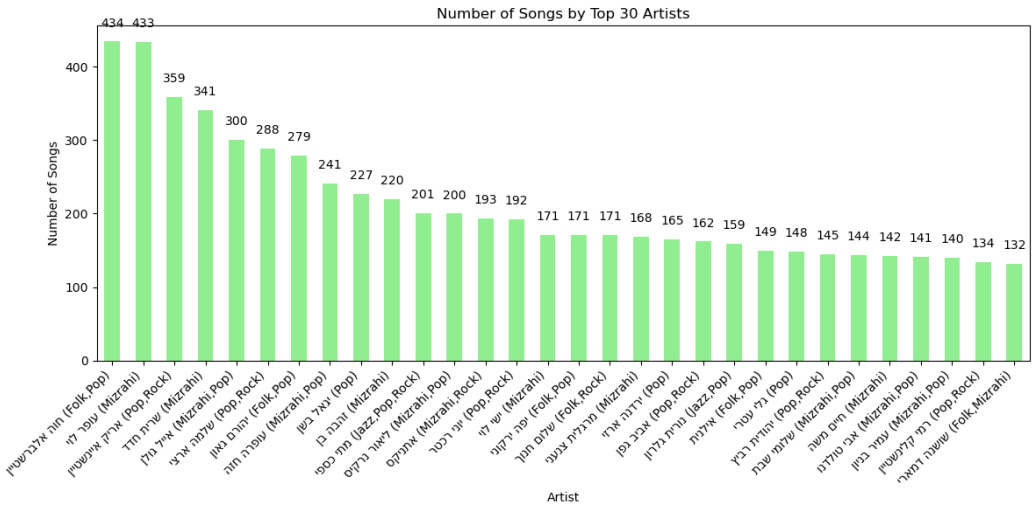


The five primary music styles represented in the data are:

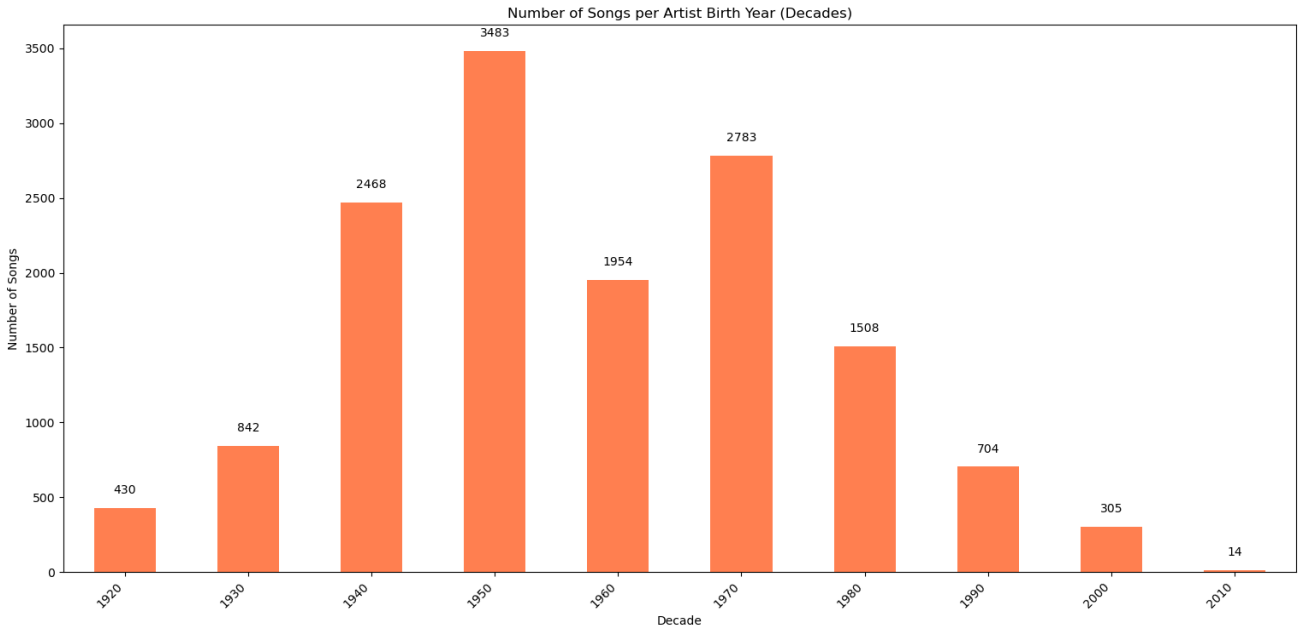
- Pop & Rock
- Mizrahi & Pop
- Mizrahi
- Folk & Pop
- Pop

Most of the data is related to Pop music, either by itself or in combination with other styles.

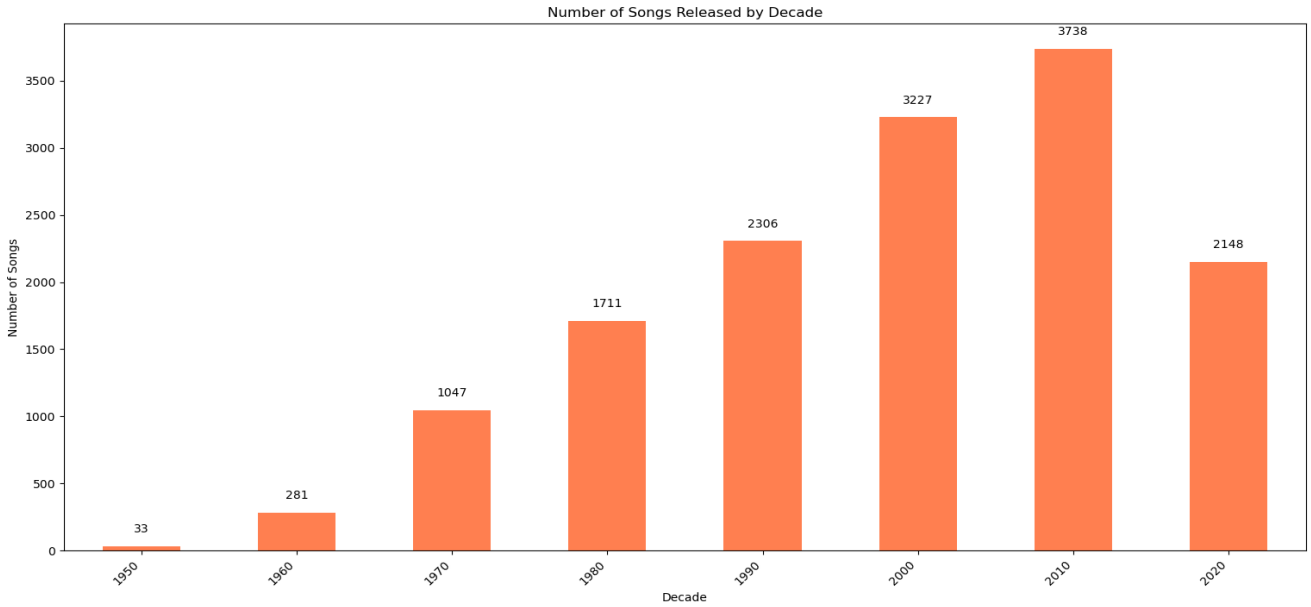
Top 30 artists by song count:



Most of the artists in the dataset were born between 1940 and 1990, with a significant concentration in the 1950s and 1960s:



Most songs in the dataset were released between 2000 and 2023:



Hypothesizes:

First, we will focus on the changes that have occurred in the music industry over the years. We had an observation that the release year of the songs can produce interesting results that are caused by the changes of each area during the years that can be interpreted in the lexical levels, sentiment levels and more.

Therefore, we hypothesize the following hypothesizes:

1. Creativity measurement has been reduced over the years due to commercialization of music, technological advancements, evolution of musical styles saturation, etc.
    - We can achieve that by looking at different features that change over the years and match the criteria of creativity levels.
2. Syntax complexity has reduced over the years.
3. Political and inter-country events significantly influence the sentiment. Periods of conflict and wars will correlate with more negative sentiments, while times of stability and peace will correlate with more positive sentiments in the lyrics.
4. Vocabulary richness increases over the years.
5. Songs style variety has increased over the years – song's style elaborated, and new styles created and merged due to society diversity.

Moreover, we thought about more unsupervised hypotheses that can be examined using our dataset that aren't dictated by specific area changes but only by the artists themselves.

6. Oriental songs are inferior to other genres by their complexity levels (Syntax and Lexical).
7. Songs by Yehoram Gaon are less creative (more banal) than other songs
8. Genres like 'Folk, Pop', and 'Pop, Soul' are close to each other by their syntax complexity.

To validate or dis-validate the hypotheses, we divided our features into subsets of features that represent each subject we want to check.

Before we made the analysis of the hypothesizes, we made different normalizations for each subject and action we wanted to perform on the data.

Before classifications, we normalized the data using Min-Max-Scaler.

- We did this normalization to prevent outliers affecting the results in a way that produces inaccurate results and ensures comparability across different features.

How we produced Creativity Measurement:

We chose a subset of features related to lyrical creativity - Some features are inverted to align with the creativity score, meaning higher values in these features represent higher creativity.

```
adjusted_creativity_features = [
    'uniqueWords', 'ratioOfTotalWordsToUnique', 'percentageOfTotalWordsToUnique',
    'DiffLemmas', 'DiffPOS', 'bigramsEntropy', 'trigramsEntropy',
    'averageSetWordLength', 'WordsRhymes', 'RatioOfPOStoWords','NumberOfUniqueWordsby1/freq',
    'inv_avgSimilarityMeasure', 'inv_average_word_frequency','inv_avg_word_similarity_hebrew','inv_avg_word_similarity_english'
]
```

- The normalized features are combined to create a single creativity score for each song.

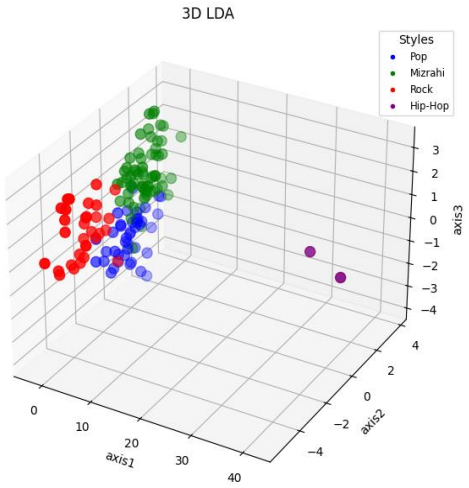
Before we tried to prove our hypotheses, we wanted to validate that Song styles can be differentiated by our numeric dataset.
We chose four main music styles for that mission: "Pop", "Oriental", "Rock" & "Hip-Hop".

To do so, we will activate LDA on the song styles creating a better separating feature space of the full features – Indicating that the data can be classified well by our numerical features.
Moreover, we will activate our classifications method to validate the LDA results.

We batched the songs into an average of 32 songs of the same style to make a good generalization and robust results.

Results:

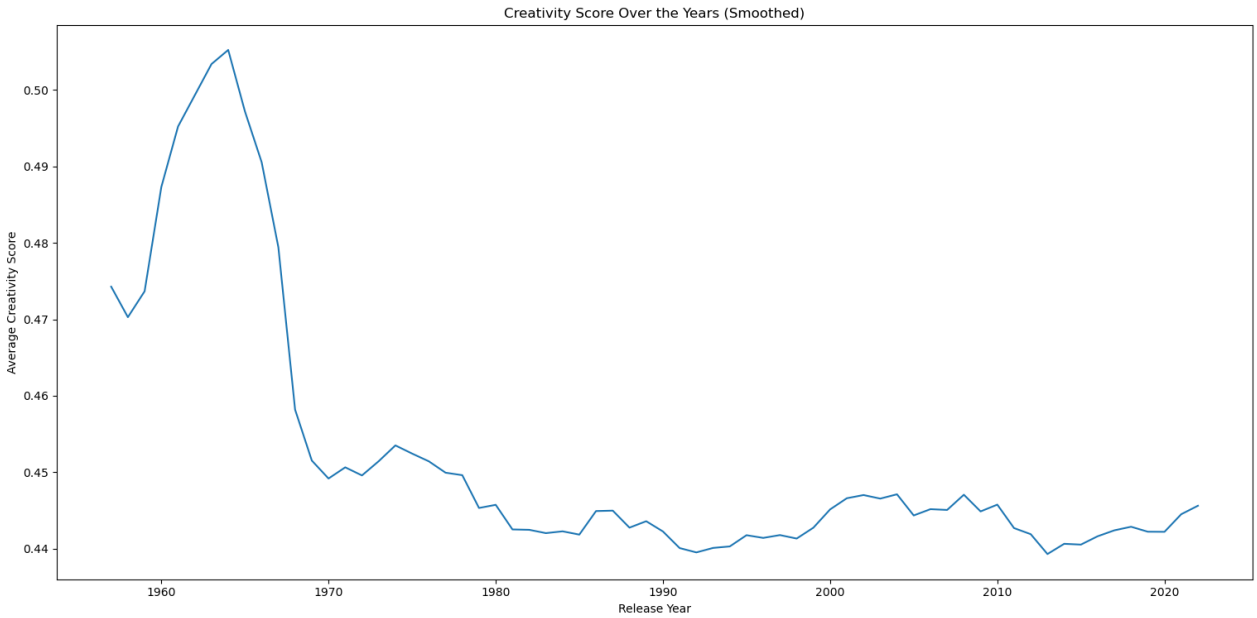
```
Model: RandomForest
Train Accuracy: 1.0000
Test Accuracy: 0.7500
current model: LogisticRegression

Model: LogisticRegression
Train Accuracy: 0.8308
Test Accuracy: 0.7857
current model: SVM (Linear)

Model: SVM (Linear)
Train Accuracy: 0.8154
Test Accuracy: 0.8214
current model: SVM (RBF)

Model: SVM (RBF)
Train Accuracy: 0.8462
Test Accuracy: 0.8214
current model: NeuralNetwork

Model: NeuralNetwork
Train Accuracy: 1.0000
Test Accuracy: 0.8214
```



We can see from the results that our numerical dataset achieves pretty good results on the test set (82%) predicting the style of a given song of four different songs styles.
It can be observed either by the 3D feature space of the LDA algorithm.

Hypothesis One:

The creativity over the song's years released



1950s-1960s:

- There is a slight increase in creativity scores in the late 1950s, followed by a significant rise and peak around the mid-1960s, indicating high lyrical creativity.

1970s-1980s:

- After the mid-1960s peak, creativity scores decline steeply in the late 1960s and continue to decrease, remaining low throughout the 1970s and 1980s, suggesting reduced lyrical creativity.

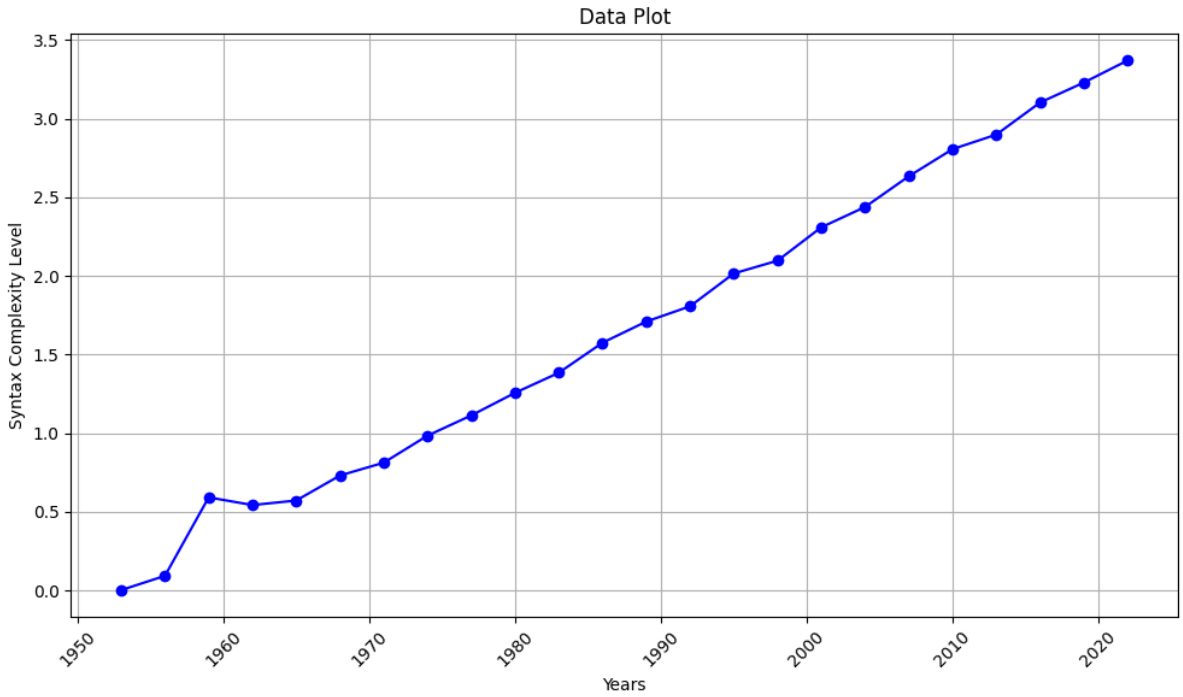
1990s-Present:

- The scores stabilize in the 1990s, with minor fluctuations, and show a slight upward trend from the late 2000s onwards, indicating a gradual increase in creativity in recent years.

Hypothesis Two:

Similarly, we chose the subset of features that define the syntactic complexity of the songs and made an average of them per song over the years.

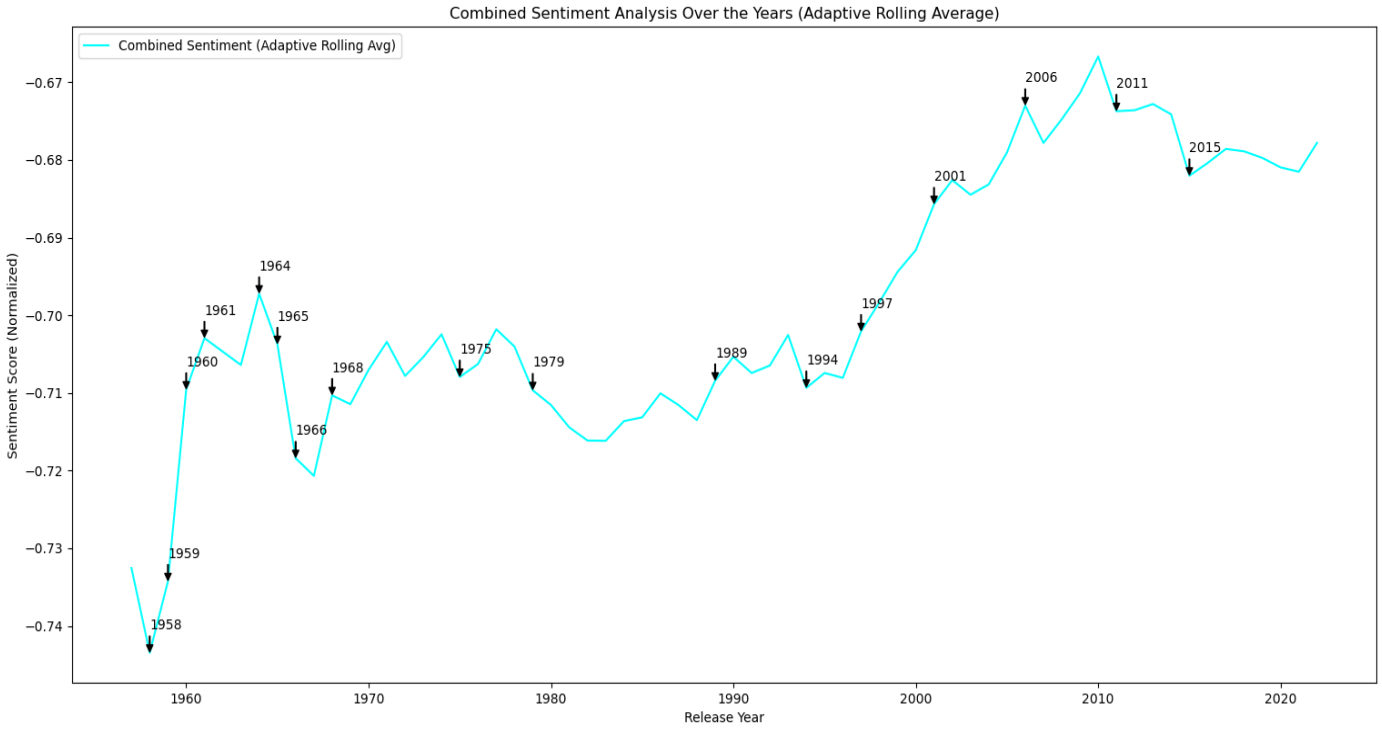
Features chosen: ‘**numberOfRepeatedWords’**, **'readabilityMeasure', ‘DiffLemmas’, ‘DiffPOS’, ‘trigramsEntropy’, ‘bigramsEntropy’.**



Surprisingly, we got a result that shows the complexity measurement over the years increasing linearly, indicating that the songs are becoming more and more syntactically complex in contrast to our hypothesis that claims a reduction in the measurement values.

Hypothesis Three:

Israeli political event that we explore according to it. timeline:
https://he.wikipedia.org/wiki/%D7%A6%D7%99%D7%A8_%D7%94%D7%96%D7%9E%D7%9F_%D7%A9%D7%9C_%D7%94%D7%94%D7%99%D7%A1%D7%98%D7%95%D7%A8%D7%99%D7%94_%D7%A9%D7%9C_%D7%99%D7%A9%D7%A8%D7%90%D7%9C



conclusions:

Based on the sentiment analysis chart of Hebrew songs over the years, where higher values on the y-axis indicate more positive sentiment, a brief of the significant years and periods of sentiments:

1. 1958-1966:
   - 1958: The lowest sentiment score observed, potentially influenced by the aftermath of the Suez Crisis in 1956.
   - 1964: Marked improvement in sentiment scores. This period corresponds with the establishment of the Increase in morale Victory - we managed to conquer Sinai

2. 1966-1984:
   - 1967: increase in sentiment after the Six-Day War, military success.
   - 1973: The Yom Kippur War's impact is evident, with declining sentiments in the early 1970s.
   - 1984: Recovery in sentiment scores post-Yom Kippur War, possibly reflecting societal resilience and adaptation.

3. 1985-2000:

- 1987: The First Intifada leads to a sharp decline in sentiment, indicating the impact of ongoing conflict.
- 1990-1991: Fluctuations in sentiment during the Gulf War period.

4. 2001-2011:
- 2001: The Second Intifada brings another sharp decline in sentiment.
- 2005-2008: Disengagement from Gaza and subsequent military operations like the Second Lebanon War and Operation Cast Lead result in fluctuating sentiments.
- 2011: Social protests may have contributed to fluctuations but overall reflect a period of political and social change.

5. 2015-2022:
- 2015: Sentiments remain relatively stable post-2014 Operation Protective Edge.
- 2022: Slight improvement, possibly reflecting current political and social dynamics, including responses to recent events like Operation Breaking Dawn.

Throughout different periods, significant political events have clearly influenced the sentiment in songs. During times of conflict, such as wars and intifadas, the sentiment in songs tends to be more negative, reflecting the societal mood and the challenges faced by the nation. Conversely, periods marked by peace processes, social change, and economic growth show an increase in positive sentiment, mirroring the hopeful and optimistic outlook of society.

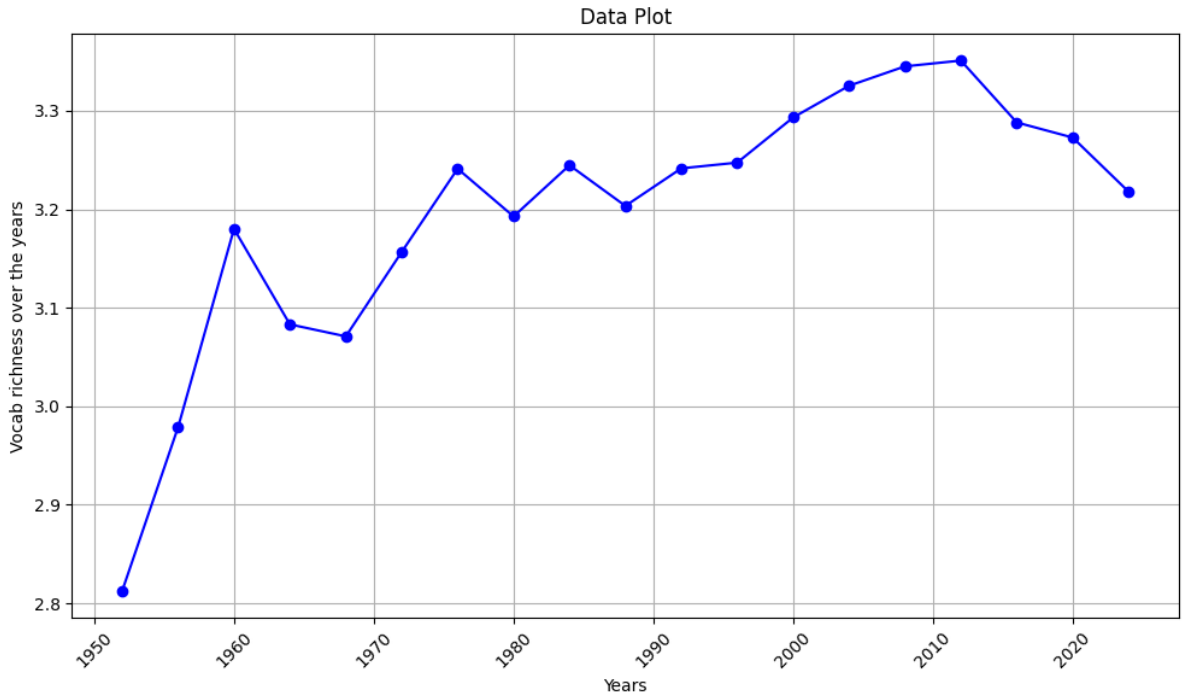
Our analysis supports the hypothesis that political events significantly influence the sentiment expressed in Hebrew song lyrics. The findings highlight a correlation between the societal mood during different political periods and the sentiment reflected in the music. The study underscores the role of music as a powerful medium for expressing and influencing public sentiment, providing valuable insights for understanding the cultural and historical context of musical trends.

It should be noted that there are different factors such as different styles of music that can affect semantics.

Hypothesis Four:

For that hypothesis, we chose to use the subset of the following features that describes best vocabulary richness.

'avgSimilarityMeasure', 'RatioOfPOStoWords', 'numberOfRepeatedWords', 'averageSetWordLength', 'word_similarity'



The hypothesis has been approved by the data showing that the vocabulary richness has getting higher and higher values over the year.

Hypothesis Five:

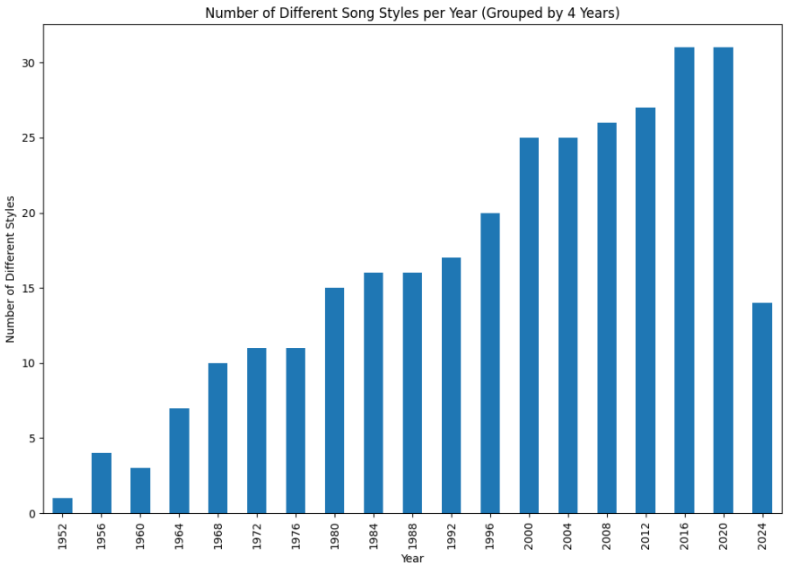
For this hypothesis, we wanted to group or dataset in a way that will allow us to check whether this hypothesis is correct or not.

We grouped the dataset into years and made a bucket for each year of the different song's styles per four-year gap.

We were assuming that because music genres constantly evolve and diversify over time. New genres and subgenres emerge as artists experiment with new sounds and influences.

In addition, increased global connectivity allows for greater cross-cultural exchange, leading to the introduction and blending of diverse musical styles.

Lastly, Advances in music production technology have made it easier for artists to experiment and create new sounds, contributing to the proliferation of styles.



We can see that as we expected we can see constant increasing number of different music styles over the years.
Hypothesis approved by the data.

For the validation of the next four hypothesis, we needed first to determine if the data can even be classified solely based on numerical features extracted from song lyrics and other metadata. We sampled the data by specific song styles and activated ML and DL classifiers. We were expecting to get good classification results due to the differences between songs styles.

The process involved the following steps:

1. **Train-Test Split**: The dataset was divided into two sets, with 70% used for training the model and 30% for testing it. This split was randomly sampled to ensure equal representation of each genre in both sets.
2. **Train a Classifier**: We used a DL simple network, SVM with 'RBF' kernel, a linear one, Logistic regression and random forest that was trained on the training set. This model learned to classify songs into different genres based on various extracted vocabulary richness features.
3. **Predict and Evaluate**: The trained model was used to predict the genres of the songs in the test set. The accuracy of these predictions was calculated for each genre, providing a measure of the model's performance using the features we extracted.
4. **Feature Importance**: The importance of each feature in making predictions was calculated and ranked. This analysis helped identify which features were most influential in distinguishing between genres, providing insights into the characteristics that define each genre.

If we had a good percentage of classifying, we understood that there's a difference between them on the syntactic plain.

After that part, we used dimension reduction ML algorithms on unsupervised data to show the results of each hypothesis.
What we did was to activate PCA and LDA and use its 2d and 3d feature space per hypothesis to help us visualize the classifiers results.
As mentioned above, we used a ML model to help us identify the best 3d feature space where the data is separated with the best accuracy.

Hypothesis 6:

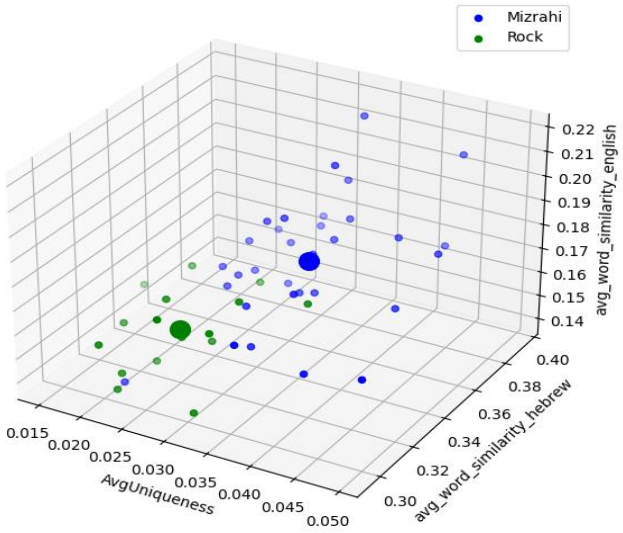
For this hypothesis, we isolated the Rock & Oriental songs and as described above, activated the process on this sub-dataset while choosing only a subset of features that describes well the syntactic and lexical complexity of a song.
It is important to notice that we are batching the data to 32 sized batches of each genre and processing the mean value for each batch.
We can see that this way, our classification models get better generalization and better accuracy on the test-set.

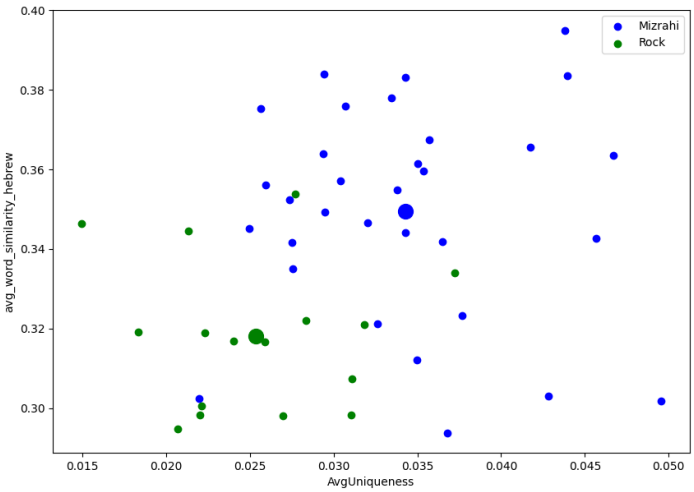
These features are:

Wordcount, uniqueWords, numberOfRepeatedWords, ratioOfTotalWordsToUnique, DiffLemmas, bigramsEntropy, trigramsEntropy, averageSetWordLength, WordsRhymes, RatioOfPOStoWords, readability Measure, NumberOfUniqueWordsby1/freq, AvgUniqueness, percentageOfRepeatedWords, theUniquenessLvlOfTheRepeatedSongs, semantic_similarity, average_word_frequency, avg_word_similarity_hebrew, avg_word_similarity_english, word_similarity_large,

There are the classification and visualization results:

```
Model: RandomForest
Train Accuracy: 1.0000
Test Accuracy: 0.7931
current model: LogisticRegression

Model: LogisticRegression
Train Accuracy: 0.9091
Test Accuracy: 0.8621
current model: SVM (Linear)

Model: SVM (Linear)
Train Accuracy: 0.9545
Test Accuracy: 0.8621
current model: SVM (RBF)

Model: SVM (RBF)
Train Accuracy: 0.9848
Test Accuracy: 0.8276
current model: NeuralNetwork

Model: NeuralNetwork
Train Accuracy: 1.0000
Test Accuracy: 0.8966
```



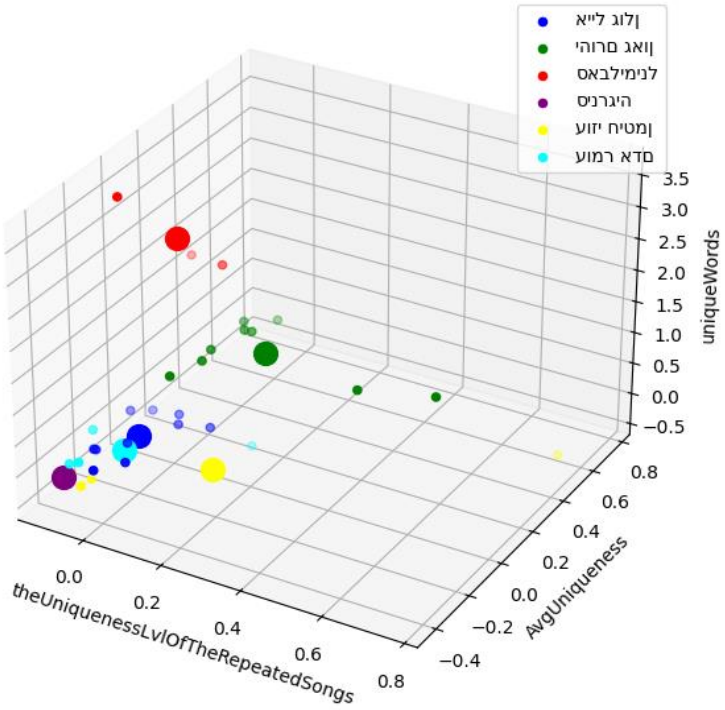


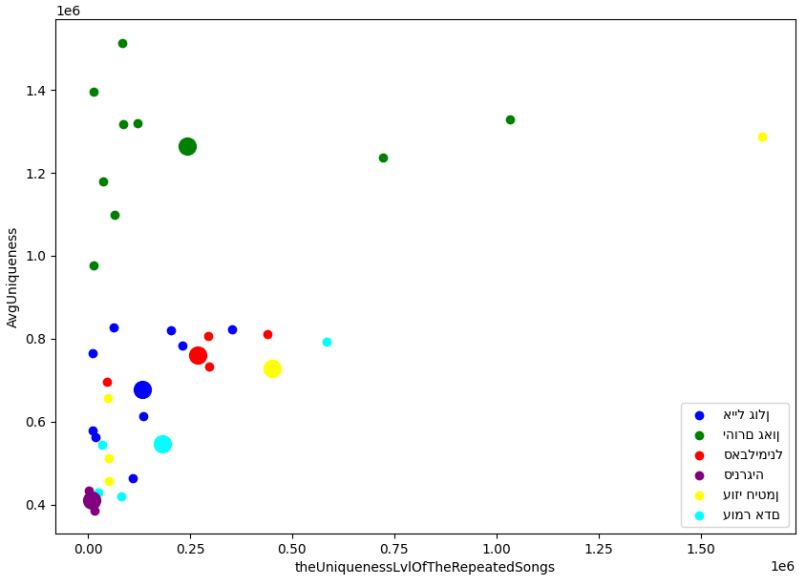
We can see that the classification result supports our hypothesis in the way that these two genres have differences on the syntactical feature space.
After we are reducing our features into the best 3D and 2D feature spaces, we can see the differences between the two genres. We can see on the 3D & 2D feature space that the value of the average word uniqueness in the oriental songs are much higher then rock. In addition, the value of the word similarity using word2vec in both languages is higher. This result implies a higher level of language complexity due to the usage of different words with high uniqueness level with the same meaning indicating the oriental songs have more diverse words. We can deduce that the complexity level of oriental songs is higher, contradicting our hypothesis.

Hypothesis 7:

For this hypothesis, we prepared the data in such a way that the label now is the artist – We took a group of artists including Yehoram Gaon.
We're using dimensional reduction over a set of features that describes creativity as explained above.

The data indicates on interesting results:





We can see in the 2D space that songs by Yehoram Gaon are more creative than other songs.
The observation concluded by the fact that the Average Uniqueness of the words in Yehoram's songs are much higher than the rest. We can see that even the uniqueness of the words that repeat more than 4 times in his songs are usually more unique than others. Looking at the 3D space, showing even slightly higher values of the unique words that Yehoram uses in his songs.
Overall, by the observation on the data we can conclude that songs by Yehoram Gaon are surprisingly more creative than other songs, contradicting our hypothesis.

Hypothesis 8:

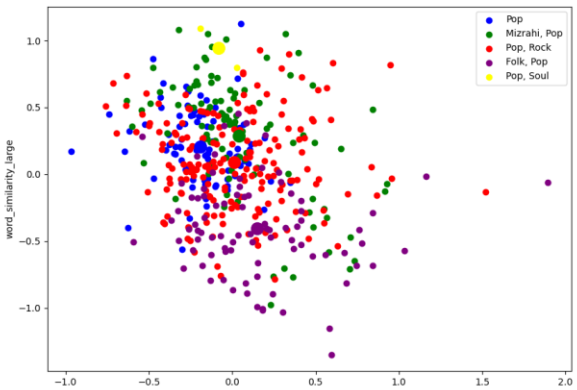
To check that hypothesis, we will create a sub-dataset of the songs that are partly Pop styled. In addition, we will choose a subset of features that align with syntax complexity analysis.
We will try to classify them, and then sample the best separating features for visualization. Our expectations are low accuracy in the classification methods and avg values for the feature spaces that are close to each other.

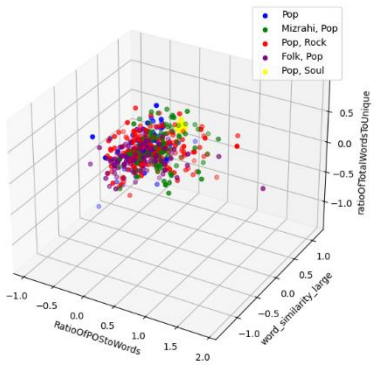
The set of features:

- **DiffLemmas**
- **DiffPOS**
- **bigramsEntropy**
- **trigramsEntropy**
- **readabilityMeasure**
- **NumberOfUniqueWordsby1/freq**
- **AvgUniqueness**
- **percentageOfRepeatedWords**
- **theUniquenessLvlOfTheRepeatedSongs**
- **avg_word_similarity_hebrew**
- **word_similarity_large**

The results:

```
Model: RandomForest
Train Accuracy: 1.0000
Test Accuracy: 0.5221
current model: LogisticRegression

Model: LogisticRegression
Train Accuracy: 0.5315
Test Accuracy: 0.5037
current model: SVM (Linear)

Model: SVM (Linear)
Train Accuracy: 0.5315
Test Accuracy: 0.4890
current model: SVM (RBF)

Model: SVM (RBF)
Train Accuracy: 0.5757
Test Accuracy: 0.5368
current model: NeuralNetwork

Model: NeuralNetwork
Train Accuracy: 0.5568
Test Accuracy: 0.5257
```





As expected, we can see that the classification results indicate that the styles are kind of close to each other by their syntax complexity features. We can see that by the low accuracy level of predicting the song's style of this similar styles.
Moreover, when we are sampling the 2D and 3D space by the best separating features, we can see that we get a big point cloud that cannot be classified.
We can conclude that Genres like 'Folk, Pop', and 'Pop, Soul' are close to each other by their syntax complexity as we assumed. Hypothesis proved as correct.

Appendix:

Explanation of Features in the Song Class

4. wordCount

- **Description**: The total number of words in the song's lyrics.
- **Calculation**: Directly from the dataset.

5. uniqueWords

- **Description**: The number of unique words in the song's lyrics.
- **Calculation**: Directly from the dataset.

6. releaseYear

- **Description**: The year the song was released.
- **Calculation**: Retrieves the release year of a song using Spotify's API, Wikipedia, and Shironet, selecting the first available valid year from these sources.

7. songInEnglish

- **Description**: The English translation of the song's lyrics.
- **Calculation**: Translates song lyrics from Hebrew to English using Google Translate and a machine learning translation model, combining chunks of translated text for complete lyrics representation.

8. translatedWords

- **Description**: A list of words from the English translation of the song's lyrics.
- **Calculation**: Splits the English translation of the song's lyrics into individual words using regular expression-based delimiters like periods, commas, and spaces

9. bigrams

- **Description**: Count the unique word pairs (bigrams) in the song's lyrics.
- **Calculation**: Generates counts of unique word pairs(bigrams) in the song's lyrics using the same custom N-gram parser.

10. trigrams

- **Description**: Count the unique word triplets (trigrams) in the song's lyrics.
- **Calculation**: Generates counts of unique word triplets (trigrams) in the song's lyrics using the same custom N-gram parser

11. numberOfRepeatedWords

- **Description**: The count of words that are repeated in the song's lyrics.
- **Calculation**: Calculates the count of repeated words in the song's lyrics by subtracting the count of unique words from the total word count.

12. ratioOfTotalWordsToUnique

- **Description**: The ratio of unique words to total words in the song's lyrics.
- **Calculation**: Computes the ratio of unique words to total words in the song's lyrics.

13. percentageOfTotalWordsToUnique

- **Description**: The percentage of unique words out of the total words in the song's lyrics.
- **Calculation**: Calculates the percentage of unique words relative to the total words in the song's lyrics by multiplying the ratio of total words to unique words by 100.

14. LemmatizedWords

- **Description**: A list of lemmatized (base form) words from the song's lyrics.
- **Calculation**: Produces a list of lemmatized (base form) words from the song's lyrics using a text parsing module.

15. POSperWord

- **Description**: A list of parts of speech for each word in the song's lyrics.
- **Calculation**: Generates a list of parts of speech tags for each word in the song's lyrics using a POS tagging module.

16. sentimentScore

- **Description**: A numerical score representing the overall sentiment of the song's translated lyrics.
- **Calculation**: Provides a numerical score representing the overall sentiment of the song's translated lyrics using the Afinn sentiment analysis tool.

17. positiveWords

- **Description**: The count of positive words in the song's translated lyrics.
- **Calculation**: Counts the number of positive words in the song's translated lyrics using a sentiment analysis method that evaluates each word individually.

18. negativeWords

- **Description**: The count of negative words in the song's translated lyrics.
- **Calculation**: Counts the number of negative words in the song's translated lyrics using the same method as for positive words.

19. numberOfDiffLemmas

- **Description**: The number of different lemmas (base forms) in the song's lyrics.
- **Calculation**: Determines the number of different lemmatized forms in the song's lyrics.

20. numberOfDiffPOS

- **Description**: The number of different parts of speech in the song's lyrics.
- **Calculation**: Determines the number of different parts of speech tags in the song's lyrics

21. avgSetWordLength

- **Description**: The average length of unique words in the song's lyrics.
- **Calculation**: Calculates the average length of unique words in the song's lyrics.

22. avgAllWordLength

- **Description**: The average length of all words in the song's lyrics.
- **Calculation**: Calculates the average length of all words in the song's lyrics.

23. readabilityMeasure

- **Description**: A measure of the readability of the song's translated lyrics, calculated using readability formulas.
- **Calculation**: Computes a readability score for the song's translated lyrics using the Flesch and Fog readability formulas.

24. amountOfWordsRhymes

- **Description**: The number of rhyming words in the song's translated lyrics.
- **Calculation**: Counts the number of rhyming word pairs in the song's lyrics.(suffix 2)

25. ratioOfWordsToPOS

- **Description**: The ratio of different parts of speech to the total number of words in the song's lyrics.
- **Calculation**: Calculates the ratio of different parts of speech to the total number of words in the song's lyrics.

26. amountOfBiGrams

- **Description**: The number of unique bigrams (word pairs) in the song's lyrics.
- **Calculation**: Counts the number of unique bigrams in the song's lyrics.

27. amountOfTriGrams

- **Description**: The number of unique trigrams (word triplets) in the song's lyrics.
- **Calculation**: Counts the number of unique trigrams in the song's lyrics.

28. bigramsEntropy

- **Description**: The entropy (measure of randomness) of the distribution of bigrams in the song's lyrics.
- **Calculation**: Computes the entropy (a measure of randomness or diversity) for the distribution of bigrams in the song's lyrics.

29. trigramsEntropy

- **Description**: The entropy (measure of randomness) of the distribution of trigrams in the song's lyrics.
- **Calculation**: Computes the entropy for the distribution of trigrams in the song's lyrics.

30. avgSimilarityMeasure

- **Description**: The average semantic similarity between words in the song's lyrics.
- **Calculation**: Calculates the average semantic similarity between all pairs of words in the song's lyrics.

31. numberOfUniqueRankedWords

- **Description**: The number of unique words in the song's lyrics that are considered unique based on their frequency rank.
- **Calculation**: Counts the number of unique words in the song's lyrics that are rare based on their frequency rank.

32. avgUniquenessOfSong

- **Description**: The average uniqueness score of the words in the song's lyrics.
- **Calculation**: Calculates the average uniqueness score of the words in the song's lyrics based on their frequency.

33. repetitionWordsPercentage

- **Description**: The percentage of words that are repeated more than four times in the song's lyrics.
- **Calculation**: Computes the percentage of words that are repeated more than four times in the song's lyrics.

34. repetitionWordsUniqueness

- **Description**: The average uniqueness score of the words that are repeated more than four times in the song's lyrics.
- **Calculation**: Calculates the average uniqueness score of the words that are repeated more than four times in the song's lyrics.

35. semantic_similarity

- **Description**: Measures the cosine similarity between the embeddings of the original and translated lyrics.
- **Calculation Method**: Embeddings from BERT are calculated for both the original and translated lyrics. The cosine similarity between these embeddings is then computed.

36. average_word_frequency

- **Description**: Calculates the average frequency of all words in the song lyrics, based on a large word frequency list.
- **Calculation Method**: Each word in the lyrics is looked up in a frequency list to find its frequency of use in each language. The average of these frequencies provides the average word frequency.

37. heBERT_sentiment

- **Description**: The sentiment score derived from the Hebrew BERT (heBERT) model.
- **Calculation Method**: The song lyrics are input into a heBERT model fine-tuned for sentiment analysis. The output score reflects the overall sentiment conveyed by the lyrics.

38. avg_word_similarity_hebrew

- **Description**: The average cosine similarity between all pairs of Hebrew word embeddings in the lyrics.
- **Calculation Method**: Hebrew words are converted into embeddings using a pre-trained model. Cosine similarity is calculated for every pair of embeddings, and the average is taken.

39. avg_word_similarity_english

- **Description**: The average cosine similarity between all pairs of English word embeddings in the translated lyrics.
- **Calculation Method**: Like Hebrew but applies to the English translation of the lyrics.

40. Birth Year

- **Description**: The birth year of the artist or band.
- **Calculation Method**: Calculated by fetching the artist's Wikipedia page, extracting text surrounding the word "נולד," and then using a regular expression to locate and return the four-digit year found within that text segment.

41. word_similarity_large

- **Description**: A measure of the overall similarity between words in a song's lyrics, intended for longer texts.
- **Calculation Method**: Large text embeddings are generated for the entire lyrics, and similarity scores are calculated and averaged over the text span.