Due to her immense popularity, there have been several exploratory data analyses of her songs using either machine learning or other Python data analysis techniques. However, no study so far has focused specifically on using supervised learning techniques such as K-Nearest Neighbour to verify the relationship of the audio features of the tracks to the album titles or ‘eras’ of her career.

One exploration (Panwar, 2022) focused on using certain musical attributes to predict the *popularity* of the song using linear regression and random forest machine learning algorithms, and concluding that using the random forest model had the highest accuracy score.

Another research project (Lord Adg, 2020) used NLP techniques such as word vectorization and TF-IDF to recommend songs based on their lyrics. One very thorough (Bernal, 2023) analysis visualized the mean popularity of each album, then plotted the audio attributes per album to capture the ‘mood’ or ‘tendency’ of each album and then looked at the correlations between an album’s audio features and its popularity, but did not utilize any machine learning techniques. Finally this Kaggle study (Tofany, 2023) used visualizations and aggregate grouping to suggest that there was considerable variability across the albums for features such as ‘acousticness’ and valence, thus concluding that these characteristics might have some relationship to the album style. However, it stopped short of using machine learning classification models to verify this further.

As such, this study hopes to contribute to what appears to be an active and lively community of Taylor Swift fans conducting various analyses of her work by comparing the performance of several supervised classification algorithms, as well as unsupervised K-Means clustering, to attempt to discern patterns and tendencies in the mood and audio features of Swift’s different albums.

**Overview of Analyzing Music using Machine Learning Techniques**

In terms of analyzing music using machine learning, there have been numerous studies and articles published about making use of the Spotify API to predict a song or album’s *popularity* using the audio characteristics of the music (Nair, 2023) (Leshem, 2022). As popularity on Spotify is a continuous, numerical variable, these studies have mainly used supervised regression models for their predictions. While there are fewer studies focused on predicting which *album* a song belongs to with machine learning, there exists a large body of research conducted on the classification of songs as certain *genres* based on their audio characteristics (Bahuleyan, 2018). This task constitutes a similar type of analysis to categorizing songs into albums, as a wide range of techniques are used and evaluated in terms of their ability to capture the musical ‘mood’ or feeling of a track based on a set of audio attributes including tempo, key and frequency. Bahuleyan explores the use of machine learning algorithms such as Random Forest, Logistic Regression and Gradient Boosting to evaluate a song’s genre, concluding that both accuracy and F-score (‘the harmonic mean between precision and recall’) were highest for the Gradient Boosting algorithm. Gradient Boosting is defined as a classifier which combines a number of weaker learning algorithms, such as decision trees, called ‘learners’, and improves over time by focusing ‘on the instances where the previous learners made errors’ (Bahuleyan, 2018).

Another article (GeeksForGeeks, 2022) demonstrates how supervised learning algorithms and neural networks can be used to class songs by their audio features into genres, comparing the performance between K-Neighbours and Decision Tree based classifiers. They concluded that the ‘CatBoost’ decision-tree based algorithm resulted in the highest accuracy score. While both this article and another research study (Zhang, 2021) suggest that deep neural networks have the greatest predictor power when inferring a track’s genre, the scope of this study will be to consider only supervised and unsupervised machine learning algorithms to attempt this classification task.

This study details the usage of the *librosa* music library to extract the waveforms and frequencies of various songs before predicting their genre using K-NN and Random Forest classifiers, which resulted in relatively high accuracy scores of 0.86 and 0.91 respectively (Kurganov, 2020), thus demonstrating the potential of applying supervised machine learning techniques to determine musical styles. Another project (Agrawal, 2022) showcases the manual implementation of the K-Nearest Neighbour algorithm to predict music genre with an accuracy score of 0.71. As these scores seem to be somewhat promising, this study will assess whether these techniques are equally proficient for determining which album a song belongs to (in the case of Taylor Swift’s work).

There is much potential for extending this project to evaluate the predictive power of more advanced techniques such as convolutional neural networks, or eXtreme Gradient Boosting training multiple decision trees in a ‘fast and parallelized manner’ (Bahuleyan, 2018). However, in context of the time limitations on this study, a weighted K-Nearest Neighbour classification algorithm will be implemented in Python and assessed in terms of its ability to predict the album a Taylor Swift song belongs to by calculating the Euclidian distance between the audio parameters which characterize an unseen song (sample) and those of the songs used as training data. Next, the results of the K-NN classifier will be compared to the performance using a Naïve-Bayes classifier which will also be explained and implemented using the NumPy library. Then, a ready-made Random Forest classifier will be applied from the scikit-learn library, and then tuned using the scikit-learn grid search feature in order to determine the optimal hyperparameters for prediction, as shown in one article (Tirendaz AI, 2022). Random Forest has been singled out before as having ‘a high level of performance accuracy’ (Chaudhury, Karami, & Ghazanfar, 2022) for predicting musical genres for song recommendation. Finally, K-Means clustering, an unsupervised learning algorithm, will be used to ascertain whether the clusters detected correspond to the actual album labels of the samples.

**K-Nearest Neighbour**

**Cross-Validation**

A serious challenge in supervised machine learning algorithms is overfitting: when the model is overly complex, and fits the training data too well (technically termed as having a ‘low bias’ and ‘high variance’), but does not *generalize* or make accurate predictions when extended to new, unseen data. The aim is to make the classifier model as generalizable as possible, maximizing its ability to predict labels for new data items as accurately as possible.

In the specific case of k-NN, overfitting can happen when the value of *k* is too small. (Sachinsoni, 2023). For instance, when *k*=1, the decision boundary will follow and ‘stick’ to each individual training sample too closely, leading to the model merely ‘memorizing the training data’ rather than trying to generalize by learning more nuanced patterns (Sachinsoni, 2023). In contrast, setting *k* too high can lead to underfitting as the decision boundary becomes overly smoothed and thus fails to reflect the local variations and tendencies in the samples.

Cross-validation can help to maximize the use of the data, allowing us to use all of it but while eliminating the redundancy of using the same data for training and test samples at any time. In a simple train-test split, if we set the proportion of the test set to 0.2 or a fifth of the data, this means that we are not making the maximum use of all the data in the dataset. As the dataset used in this project is not particularly large, *n*-fold cross validation can ensure that all of the data is being used to improve the model. After testing on each ‘fold’, we can calculate the mean error over all the folds to get an improved idea of how well the algorithm performs on this data.

In the case of k-NN, it is important to evaluate which value of *k* leads to the best model performance. Simple cross-validation using train and test sets does not allow the evaluation of which *k* hyperparameter value is optimal for this kind of data, because the test sets will be used up already to decide which value is best. Therefore, this does not retain any data for evaluating the performance of the selected hyperparameter. As a result, N-fold Nested Cross-Validation will be used here to decide which value of *k* is optimal for this task. This enables the selection of the best hyperparameter (value of *k*) for each ‘fold’ using train and test sets, before calculating the error using this hyperparameter on the remaining ‘validation’ set for that fold.