Audio Feature Based Music Genre Classification Using Machine Learning Models

Marcus Wong Yew Hon[1]

Abstract Music information retrieval (MIR) has been an active field of research for the past decade, especially in retrieving music genre information as genre classification for music files during distribution are still reliant on manual annotations. The purpose of this research is to determine the best set of hyper parameters for the best performing machine learning models in classifying music genres based on Mel Frequency Cepstral Coefficient (MFCC) audio features. The paper also explores the best feature selection techniques to be used in tandem with the best performing machine learning models. From the results, it was found that the Extra Trees ensemble model yielded the highest accuracy among the 8 models implemented in this paper with an accuracy score of 68.5%. The greedy feature selection method provided the best accuracy improvements for the K-Nearest neighbors model, but reduced accuracy rates for Random Forest and Extra Trees models.

1 Introduction

Research in MIR (music information retrieval) for genre classification has been an active field for the past few years, yet there still remains uncertainty in the best technique to automatically classify the genre of a song based on music features. Till now, retrieval of music genre information is still reliant on the manual annotation given within the metadata of music files such as MP3s and FLACs. The act of manual annotation of music information is an expensive operation and resulted in inconsistent representation of music genre information in music files, which is prevalent in more niche and unpopular music recordings.

A dataset based on the GTZAN dataset found on Kaggle was used for this paper. The GTZAN dataset comprises of 1000 audio files in .au format belonging to 10 different genres (blues, jazz, rock, pop, classical, reggae, metal, country,

1 Cork Institute of Technology, Cork, Ireland
marcus.wong@mycit.ie

disco and hip-hop), distributed evenly among the 1000 audio files, at 100 each. Each file was sampled at 22050Hz in mono at 16-bit. The dataset found on Kaggle have all 1000 of GTZAN's audio files converted to numerical audio features to be used for classification purposes. The 30 features in the dataset includes filename, tempo, beats, chroma feature, root mean square error, spectral centroid, spectral bandwidth, roll off, zero crossing rate, genre label and 20 Mel Frequency Cepstral Coefficient (MFCC) features. As there are 10 genres within the dataset, the genre labels values will be given a numerical value (1 to 10) for each belonging genre.

. This dataset was selected for the research of this paper as it provides ample MFCC features (20) to be used for music genre classification. As MFCC provides numerical representation for the musical timbre of an audio file, it is a suitable feature for predicting music genres. The GTZAN dataset was created for the seminal work of (Tzanetakis and Cook, 2002)[2] in music genre classification. Also, the GTZAN dataset, which this dataset was based on, have been widely used for various music information retrieval studies (Sturm, 2014)[3] due to its standardized sampling formats and its wide encompassing music genre types.

2 Research

As the dataset chosen for this research paper consists of 30 features, feature selection can be done to improve the general performance and accuracy of our machine learning models. Therefore, feature selection will be chosen as the area of research for this paper to determine the most suitable feature selection method to be used for audio feature-based music genre classification.

2.1 Feature Selection

Feature selection is a method of selecting a subset of features within a dataset that is deemed more significant in representing the overall dataset. This process removes less desired features in the dataset that might degrade the performance of machine learning models as well as reducing the complexity of the model. It might also help reduce the occurrence of overfitting in a model as the reduction in

[2] Tzanetakis, G. and Cook, P. (2002). Musical genre classification of audio signals. IEEE Transactions on Speech and Audio Processing, 10(5), pp.293-302.

[3] Sturm, B. (2014). The State of the Art Ten Years After a State of the Art: Future Research in Music Information Retrieval. Journal of New Music Research, 43(2), pp.147-172.

dimension in the dataset also reduces the variance of the model (Bermingham et al., 2015)[4].

There are three main categories in feature selection methods, namely filter, wrapper and embedded. Wrapper feature selection methods utilizes machine learning models to iteratively train subset of features until the optimal subset of features that provides the best representation of a dataset is found. However, the iterative training of subsets in wrapper methods is akin to a search algorithm, thus making it computationally expensive.

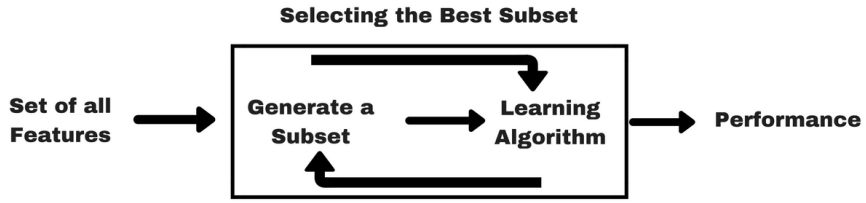

Figure 1 *Logic Flow of Wrapper Feature Selection Methods*

Filter methods compares and statistically calculates the correlation score of each feature against a target feature. This makes it less computationally expensive as compared to wrapper methods as no machine learning model training is required. However, as each feature is individually assessed against the target feature, the method will disregard the correlation between multiple features, making it a univariate approach. Thus, filter methods yield lower prediction performance than wrapper methods.

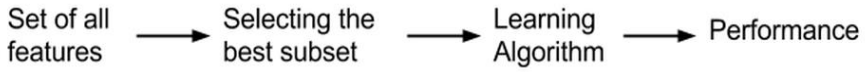

Figure 2 *Logic Flow of Filter Feature Selection Methods*

Embedded feature selection methods refer to machine learning models that have feature selection operations embedded within. This means that the feature selection process will occur as the machine learning model is being trained or constructed. Computation wise, the embedded methods are usually faster than wrapper methods but slower than filter methods as training of models are still required.

---

[4] Bermingham, M., Pong-Wong, R., Spiliopoulou, A. et al. Application of high-dimensional feature selection: evaluation for genomic prediction in man. Sci Rep 5, 10312 (2015) doi:10.1038/srep10312

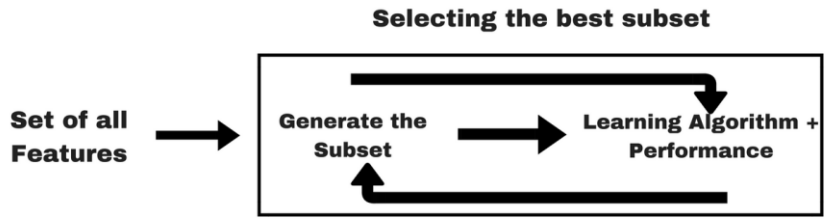

Figure 3 *Logic Flow of Embedded Feature Selection Methods*

In this paper, the Pearson's Correlation method, Backwards Elimination method, Greedy Selection method, LASSO method and Ridge method will be implemented for our music genre classification task.

2.1.1 Pearson's Correlation

Pearson's Correlation belongs to the filter category of feature selection methods as it is independent of any machine learning model training. Instead, it provides information regarding feature importance from the correlation score it calculates. Pearson's Correlation measures the linear correlation between the chosen feature and the target feature. It outputs the correlation result of values in between -1 and 1, -1 indicating maximum negative correlation, 1 indicating maximum positive correlation and 0 indicating non-existent correlation. Based on the formula (1), Pearson's correlation calculates the correlation score between two features by dividing covariance of the two features with the product of the standard deviation of the two features.

$$\rho_{X,Y} = \frac{\mathrm{cov}(X,Y)}{\sigma_X \sigma_Y} \qquad (1)$$

2.1.2 Backwards Elimination Method

Backwards Elimination method starts with providing all the features in the dataset in training a model. The feature that is deemed the least contributing (highest p-value) will then be removed. The model will then be trained again with the reduced features and have the least contributing feature removed. This process repeats until no improvements can be observed upon further feature removal. The method essentially removes features that are least contributing iteratively until a certain condition cannot be met.

As the backwards elimination method iteratively trains a model until an optimal subset of features are found, it can be classified as a wrapper feature selection method. A counter algorithm called forward selection behaves similarly to the backward elimination method, only that instead of starting with all features, it starts with no features. The best contributing feature will then be added iteratively until no significant improvements can be observed in the algorithm.

2.1.3 Greedy Feature Selection

Greedy feature selection, also known as Recursive Feature Elimination (RFE), chooses the best performing subset of features by recursively training and obtaining the scores of each model. The features are scored and rank based on the linear regression fit of the model and the lowest ranking features will be removed from the current feature set. The process repeats with the reduced current feature set until the desired number of features have been achieved in the current feature set.

This makes the greedy feature selection algorithm a type of backwards selection algorithm. The difference of greedy feature selection from regular backwards elimination method is that greedy feature algorithms removes features based on feature ranking obtained from the linear regression fit, whereas the backwards elimination method removes features on the basis of feature contribution denoted by its p-value.

2.1.4 LASSO Based Feature Selection

LASSO (*Least Absolute Shrinkage and Selection Operator*) feature selection method is classified as an embedded feature selection as the construction of the model will have feature selection performed by having regularization techniques applied to it. Regularization is a technique in which slight alteration to the fitting of a model was done in order for the model to perform generalizations better, in order words, it reduces overfitting. For LASSO, L1 regularization is performed, which penalizes the regression coefficients (feature values) with the absolute value of the coefficient's magnitude as illustrated in the formula (2). The penalty incurred on the coefficients will lead some of them to shrink to 0. LASSO will then select all features than contains non-zero coefficients, which acts as a form of feature selection.

$$Cost\ (W) = RSS(W) + \lambda * (sum\ of\ absolute\ value\ of\ weights)$$
$$= \sum_{i=1}^{N} \left\{ y_i - \sum_{j=0}^{M} w_j\, x_{ij} \right\}^2 + \lambda \sum_{j=0}^{M} |w_j| \qquad (2)$$

2.1.5 Ridge Based Feature Selection

The ridge-based feature selection method is very similar to the LASSO feature selection method, whereby they are both embedded feature selection methods and they both perform regularization techniques as a form of feature selection. Where the Ridge based approach differs is the way it penalizes regression coefficients. Instead of L1 regularization, Ridge performs L2 regularization, which penalizes regression coefficients with the square of the coefficient's magnitude as illustrated in (3). Unlike the L1 regularization in LASSO, the L2 regularization penalty does reduce coefficients to 0.

$$Cost\ (W) = RSS(W) + \lambda * (sum\ of\ squares\ of\ weights)$$
$$= \sum_{i=1}^{N} \left\{ y_i - \sum_{j=0}^{M} w_j\, x_{ij} \right\}^2 + \lambda \sum_{j=0}^{M} w_j^2 \qquad (3)$$

3 Methodology

In this section, the implementation steps done for this research paper will be detailed in its respective subsection. First, preprocessing of the dataset was performed, followed by test run of the initial 8 machine learning models using the preprocessed data. Once all 8 models have been run, the best 3 performing models will be subjected to hyper-parameter optimization. Lastly, a multitude of feature selection techniques discussed in the previous section will be implemented and tested with the 3 best performing models along with their optimal hyper-parameters obtained from the optimization process.

3.1 Preprocessing and Initial Models

Several preprocessing steps were omitted from the implementation. As there are no missing values in the dataset, missing value handling will not be necessary. As mentioned earlier, the dataset consists of 1000 rows with 10 genre labels, each split evenly among the dataset, amounting to 100 rows for each genre. This means that data imbalance handling will not be necessary as well as the dataset is already balanced. Lastly, as the dataset only consists of 30 features, dimensionality reduction will not be necessary as the process will only be useful for datasets with an extremely large number of features.

3.1.1 Initial Models

A total of 8 initial models were built for this research paper, namely, K-Nearest Neighbors (KNN), Support Vector Machines (SVM), Random Forest Ensemble, Decision Tree Classifier, Gaussian Naïve Bayes, Nearest Centroid Classifier, Extra Tree Classifier and the Extra Trees Ensemble. All initial models are created the use of the SciKit Learn library and the models are all initialized with default settings, with the exception of the tree-based classifiers (random forest, decision tree, extra tree, extra trees), where a random state of 0 was given as a parameter. All models will be subjected to 10-fold cross validation. The models are initialized in the *models.py* file and imported to the *main.py* file for use.

3.1.2 Categorical Mapping

As the chosen dataset have numerical data for all features except for the genre labels and filename columns, not much was needed to be done for categorical encoding. As the filename column is not relevant for genre classification, the column will be dropped first. This leaves the genre label column to be the only categorical column left.

As mentioned earlier, the dataset contains 10 genres, meaning the genre label column would only consist of 10 unique label values. Therefore, each genre will have its string value converted to an integer value of 1 through 10 for each respective genre. To achieve this, a dictionary containing all 10 genres will be created with the keys being the genre label and the values being its assigned integer value (1 to 10). The dictionary will then be mapped to the genre label column of the dataset, thus converting all genre labels into their numerical representation.

Table 1. Genre Labels and its New Numerical Representation

Genres	After Mapping
Rock	3
Metal	4
Disco	9
Hip-hop	8
Country	10
Jazz	5
Reggae	7
Blues	6
Classical	2
Pop	1

3.1.2 Scaling

Scaling of data must be done as the feature values have high variance and are not of scale among one another as illustrated in figure 4. A normalization approach to scaling the dataset was chosen as there are negative values present in the dataset, particularly in the MFCC columns. Since normalization will not be affect positive and negative values, the minMaxScaler function was used. Figure 5 illustrates the feature values after scaling.

tempo	beats	chroma_stft	rmse	spectral_centroid
103.359375	50	0.38026021	0.248262286	2116.942959
95.703125	44	0.306450871	0.113475412	1156.070496
151.9990809	75	0.253487102	0.151570767	1331.07397
184.5703125	91	0.269320022	0.119071715	1361.045467
161.4990234	74	0.391058605	0.137728348	1811.076084
107.6660156	51	0.356588176	0.162027627	2068.371125
161.4990234	80	0.37470972	0.11049626	2340.432873
151.9990809	74	0.430894125	0.19622159	1946.565652

Figure 4 *Feature Values Showing High Variance, Not in Scale*

tempo	beats	chroma_stft	rmse	spectral_centroid
0.268888889	0.323232323	0.423916188	0.618701922	0.400312782
0.226337449	0.262626263	0.273833256	0.275502449	0.15167252
0.539215686	0.575757576	0.16613743	0.372502293	0.196957312
0.720238095	0.737373737	0.198331878	0.289751967	0.204712889
0.592013889	0.565656566	0.445873501	0.337256199	0.321165108
0.292824074	0.333333333	0.375781787	0.399127948	0.387744086
0.592013889	0.626262626	0.412629894	0.267916819	0.458144173
0.539215686	0.565656566	0.52687452	0.486193919	0.35622508
0.20734127	0.272727273	0.242310807	0.213748148	0.139639717

Figure 5 *Scaled Feature Values*

3.1.3 Outlier Detection

For outlier detection, the 1.5x inter-quartile range rule was applied for univariate outlier detection. However, upon performing the mentioned outlier detection method, it was found that 112 rows in the dataset were deemed as outliers, which represents 11.2% of the entire dataset. As the current dataset is too small, removal of the detected outliers will negatively impact the performance of the models. Therefore, it was decided that no outliers will be removed from the dataset.

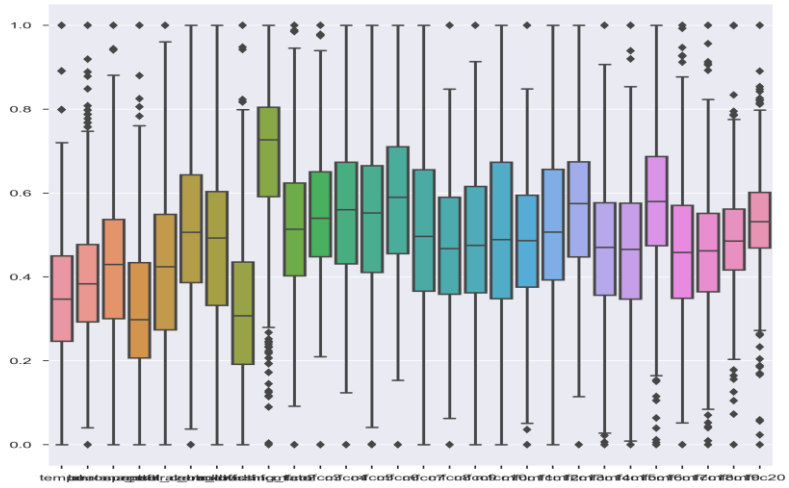

Figure 6 *Boxplot showing Outliers for All Features (112 outliers)*

3.1.4 Feature Selection

As feature selection was selected as the area of interest for the research portion of this paper, a simple univariate feature selection implementation was selected for the preprocessing phase. A rank-based feature selection approach was implemented using SciKit's SelectPercentile function, which selects features based on the highest scoring percentile. Once the ranks have been identified, the features are sorted from weakest to best. 11 of the worst features are then selected to be removed from the dataset.

A test run of the 8 initial models will be performed with a copy of the dataset that have not been reduced by feature selection. After the initial test run, the second test run will then be performed using the reduced dataset. The results from the two test runs will then be compared to have the better performing dataset be selected for hyper-parameter optimization.

3.2 Hyper-parameter Optimization

Once the best performing dataset (feature selected or original) and the three best performing models from the test runs have been selected, hyperparameter optimization can be performed.

From the test runs, the three best performing models were the KNN model, the random forest ensemble and the extra trees ensemble. As such, a parameter grid

for each of the models were initialized to be used for grid search. Grid search is a model selection technique where it performs exhaustive search of the best set of parameters for a given model. The grid search will be given a cross-fold value of 10 and the parameters grid values for each of the model are illustrated in table 2 below.

Table 2. Parameter grid values for best performing models

K-Nearest Neighbors	
Parameters	Values
n_neighbors	1 to 50
weights	Uniform, distance
p	1,2,3
algorithm	ball_tree, kd_tree, brute
Random Forest	
Parameters	Values
n_estimators	10,100,200,300,400,500
max_features	sqrt, log2
max_depth	1 to 20
criterion	gini, entropy
random_state	0
Extra Trees	
Parameters	Values
n_estimators	10,100,200,300,400,500
max_features	sqrt, log2
max_depth	1 to 20
criterion	gini, entropy
random_state	0

3.3 Feature Selection Methods

After hyper-parameter optimization is complete, the best parameters of each of the 3 selected models will be used in this section. As discussed in the research section of the paper, the Pearson's Correlation method, Backwards Elimination method, Greedy Selection method, LASSO method and Ridge method will be implemented and tested with the 3 selected models.

For Pearson's Correlation, the built-in correlation function in Panda's Dataframe data structure was used, as the function returns the pair-wise correlation of all features against a given target feature. All features that exceed a certain correlation score threshold value will then be selected as features for further processing.

In backwards elimination, an OLS model (Ordinary Least Squares) was used as the iterative training model as backwards elimination method utilizes the p-value of features to determine the feature importance ranking, which is provided by OLS. The OLS model will then be trained iteratively while slowly removing the least contributing feature denoted by the highest p-value of the current iteration. The feature removal is constrained by a highest p-value threshold, and failure to meet the threshold condition will stop the loop. This provides with the final subset of selected features.

The Greedy feature selection method was implemented using the RFECV (*Recursive Feature Elimination Cross Validation*) library in Scikit. A logistic regression model was used as the estimator for the RFECV and a cross-fold value of 10 was chosen.

A SelectFromModel feature selection model from the Scikit Library was used in tandem with a linear SVC model to perform LASSO and Ridge feature selection, as the SelectFromModel accepts any model that outputs a feature importance array. For LASSO, the penalty for the linear SVC model will be set to L1 penalty whereas the Ridge implementation will have the penalty of the SVC model be set to L2 penalty.

4 Evaluation

4.1 Initial Model Performance

Table 3. Accuracy of Initial Models Before and After Feature Selection (Univariate)

	Accuracy Score (%)	
Models	**Before Feature Selection**	**After Feature Selection**
KNN	61.6	61.3
SVM	40.4	43.59
Decision Tree	48.69	47.1
Gaussian NB	43.2	44.6
Random Forest	65.19	64.9
Nearest Centroid	41.4	42.1
Extra Tree	40.9	38.4
Extra Trees Ensemble	67.1	65.6

As observed in the table above, as mentioned in the previous section, the 3 best performing models among the 8 initial models are the KNN, Random Forest and the Extra Trees Ensemble for both instances (before and after feature selection). It can be seen that all tree-based models (decision tree, random forest, extra tree, extra tree ensemble) had better accuracy rates prior to feature selection. This might be due to the reduced complexity of the models after feature selection had also reduced the complexity of tree-based model structure, hence simplifying the decision rules within the tree. The more simplified the tree, the less fitter the model, the lower the accuracy rates.

However, the opposite can be said for non-tree models as the reduced dimension and complexity of the model after feature selection provided small improvements to the accuracy rates of predicting music genres. This might be due to the removal of less contributing features allowed the more contributing features to improve the fit of the models. This shows that tree-based models are less sensitive to less contributing features as compared to non-tree models. This also illustrates tree-based models to be more capable in classification in the presence of more features.

With the 3 best models in the non-feature selected dataset outperforming the models in the feature selected dataset, the non-feature selected dataset will be used for hyper-parameter optimization.

4.2 Hyper-parameter Optimization

Table 4. Returned Parameter Values After Hyper-Parameter Optimization

K-Nearest Neighbors	
Parameters	Values
n_neighbors	7
weights	distance
p	2
algorithm	ball_tree

Random Forest	
Parameters	Values
n_estimators	200
max_features	log2
max_depth	10
criterion	entropy
random_state	0

Extra Trees	
Parameters	Values
n_estimators	300
max_features	sqrt
max_depth	17
criterion	gini
random_state	0

Table 5. Accuracy of Best Performing Models Before and After Optimization

	Accuracy Score (%)	
Models	**Before Optimization**	**After Optimization**
KNN	61.6	64.9
Random Forest	65.19	67.6
Extra Trees Ensemble	67.1	69.0

It can be observed in table 5 that there are improvements in the accuracy of all 3 models after performing hyper-parameter optimization. It was also observed from the beginning that the Extra Trees Ensemble have been consistently ranked highest in accuracy compared to all the models implemented so far, with an accuracy reading of 69% after hyper parameter optimization.

4.3 Feature Selection Methods

4.3.1 Pearson's Correlation

The implementation of Pearson's Correlation on the chosen dataset produced the correlation heatmap as shown in figure 7.

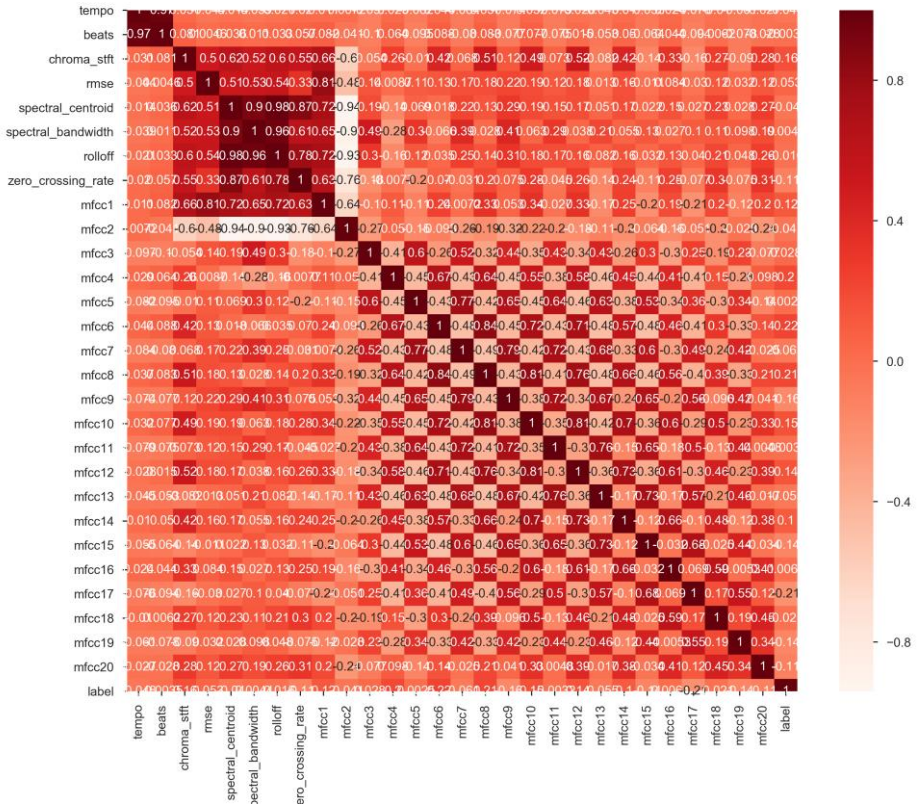

Figure 7 *Pearson's Correlation Heatmap*

Only the bottom-most row of the heatmap will be considered as it shows the correlation of all features in regards to the target feature (genre label). From the heatmap, it can be observed that the more correlated features with the label are displaying scores of more than 0.1. The selected features with scores more than 0.1 are then selected. Figure 8 shows the final subset of features.

```
Number of selected features:  13
Features:  ['chroma_stft', 'zero_crossing_rate', 'mfcc1', 'mfcc4', 'mfcc6',
'mfcc8', 'mfcc9', 'mfcc10', 'mfcc12', 'mfcc15', 'mfcc17', 'mfcc19', 'mfcc20']
```

Figure 8 *Number of features selected and feature names for Pearson's Correlation Method*

4.3.2 Backwards Elimination Method

The p-value threshold for feature removal was set to 0.05. The iterations will repeat until no features exceed the threshold as a high p-value correlates to less important features. Upon executing backwards elimination feature selection, the final subset of selected features is shown in Figure 9 below.

```
Number of selected features:  14
Features:  ['tempo', 'beats', 'chroma_stft', 'rmse', 'spectral_bandwidth',
'zero_crossing_rate', 'mfcc1', 'mfcc2', 'mfcc3', 'mfcc4', 'mfcc9', 'mfcc11',
'mfcc17', 'mfcc20']
```

Figure 9 *Number of features selected and feature names for Backwards Elimination Method*

4.3.3 Greedy Feature Selection

Upon executing the greedy feature selection algorithm, the returned selected features were obtained and is illustrated in figure 10 below.

```
Number of selected features:  28
Features:  ['tempo', 'beats', 'chroma_stft', 'rmse', 'spectral_centroid',
'spectral_bandwidth', 'rolloff', 'zero_crossing_rate', 'mfcc1', 'mfcc2',
'mfcc3', 'mfcc4', 'mfcc5', 'mfcc6', 'mfcc7', 'mfcc8', 'mfcc9', 'mfcc10',
'mfcc11', 'mfcc12', 'mfcc13', 'mfcc14', 'mfcc15', 'mfcc16', 'mfcc17', 'mfcc18',
'mfcc19', 'mfcc20']
```

Figure 10 *Number of features selected and feature names for Greedy Selection*

Interestingly, there appears to be no removed features when executing greedy features selection and seen in figure 10, where all features were retained.

4.3.4 LASSO Feature Selection

The C-value (regularization parameter) was set to 0.05. The higher the value of C, the lesser features the model will select. After executing LASSO feature selection, the selected features are obtained. The selected features are illustrated in Figure 10 below.

```
Number of selected features: 23
Features: ['tempo', 'chroma_stft', 'rmse', 'spectral_centroid',
'spectral_bandwidth', 'rolloff', 'zero_crossing_rate', 'mfcc1', 'mfcc2',
'mfcc3', 'mfcc4', 'mfcc5', 'mfcc6', 'mfcc7', 'mfcc9', 'mfcc11', 'mfcc12',
'mfcc13', 'mfcc14', 'mfcc15', 'mfcc17', 'mfcc18', 'mfcc20']
features with coefficients shrank to zero: 219
```

Figure 11 *Selected Features information for LASSO method*

It can also be observed in figure 10 that the 219 feature values have been shrunk to zero. The algorithm then chooses features that have no zero values as the selected feature.

4.3.5 Ridge Feature Selection

The C-value (regularization parameter) for Ridge Method was also set to 0.05. Similar to LASSO, the higher the value of C, the lesser features the model will select. After executing Ridge feature selection method, the selected features are obtained. The selected features will be illustrated in Figure 11 below.

```
Number of selected features: 11
Features: ['chroma_stft', 'rmse', 'zero_crossing_rate', 'mfcc1', 'mfcc2',
'mfcc3', 'mfcc4', 'mfcc6', 'mfcc7', 'mfcc9', 'mfcc17']
features with coefficients shrank to zero: 0
```

Figure 12 *Selected Features information for Ridge method*

As mentioned earlier in the research section, as Ridge method utilizes the L2 regression penalize method, no regression coefficients will be shrunk to zero. This is shown in figure 11 where it states that zero features were shrunk.

4.3.6 Final Results

The final accuracy scores for the three best performing models are tabulated and illustrated in Table 6 below:

Table 6. Accuracy of Best Performing Models on Feature Selection Methods After Optimization

	Accuracy Score (%)		
Methods	**KNN**	**Random Forest**	**Extra Trees Ensemble**
Pearson's Correlation	60.39	60.1	62.3
Backwards Elim.	59.9	63.1	64.2
Greedy	64.9	67.6	69.0
LASSO	64.8	66.39	68.5
Ridge	63.89	63.29	65.19

From Table 6, it was observed that the Pearson's correlation method performed the worst out of all feature selection methods. This was to be expected as discussed earlier in the research section of this paper, where the lack of multi-correlation relationships among other related features would lead to less accurate predictions. However, where Pearson's Correlation lacks in accuracy it makes up for computational efficiency as it does not rely on training models for feature ranking.

The greedy feature selection method presented with the most curious case out of the bunch as shown previously in figure 10, where no features were removed after execution of the algorithm. Therefore, it comes to no surprise that the accuracy results remained the same for all three best performing models compared to runs prior to the feature selection process.

With the exception of greedy method, it appears that all of the tested feature selection methods showed no improvements in accuracy results as compared to the results obtained after the hyper-parameter optimization phase. However, it can be seen that the LASSO and Ridge method performed the most consistently with the least overall reduction in accuracy when compared to the results after hyper-parameter optimization.

5. Conclusion and Future Work

In this paper, a decent accuracy rate of almost 70% was achieved in classifying genres for 1000 songs based on musical features such as tempo, beat, MFCCs etc. in a dataset consisting of 10 music genres. This was done through the use of various preprocessing techniques such as scaling, categorical mapping, feature selection and outlier detection, as well as hyper-parameter optimizations. However, the various feature selection methods implemented were not able to improve upon the accuracy scores after hyper-parameter optimization. LASSO and Ridge based feature selection methods yield the most consistent and gave the least accuracy reduction scores from the hyper-parameter optimization phase.

A deeper exploration of feature selection techniques should definitely be done in the future, particularly combining the use of filter feature selection techniques in wrapper-based search of optimal feature subsets. Besides that, future exploration of more audio features such as chord progressions and voice timbres could be incorporated along with MFCC for possible improvements in music genre recognition and classification.

References

1. Sturm, B. (2014). The State of the Art Ten Years After a State of the Art: Future Research in Music Information Retrieval. *Journal of New Music Research*, [online] 43(2), pp.147-172. Available at: https://arxiv.org/pdf/1306.1461.pdf [Accessed 5 Dec. 2019].
2. Tzanetakis, G. and Cook, P. (2002). Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing*, [online] 10(5), pp.293-302. Available at: https://pdfs.semanticscholar.org/4ccb/0d37c69200dc63d1f757eafb36ef4853c178.pdf [Accessed 5 Dec. 2019].
3. Bermingham, M., Pong-Wong, R., Spiliopoulou, A., *et.al.* Application of high-dimensional feature selection: evaluation for genomic prediction in man. Scientific Reports, [online] 5(1). Available at: https://www.nature.com/articles/srep10312#citeas [Accessed 5 Dec. 2019].