Using Machine Learning Techniques for Audio Classification

Yalda Ghasemi, Jerry Chethalan, Mounika Pasavula

Abstract

The proper use of model selection and model evaluation techniques is essential in data analysis in machine learning applications of academic and industrial settings. This report reviews different classification models and their outputs by interpreting from different model evaluation and model selection techniques on the Audio forensics dataset. Here we tried to predict the response variable in the binary class setting. Simple classification techniques such as logistic regression, Naïve-bayes, K-Nearest Neighbors, support vector machine, Decision tree, Support vector machine, and model selection techniques such as ridge regression, principal component analysis (PCA)) applied to the dataset to observe the uncertainties in estimating the performance estimates. Optimal choices of hyper parameters for the data analysis are acquired from common cross-validation techniques such as K-fold cross-validation as a model section technique. Different statistical methods for data analysis comparisons are presented, and strategies for dealing with multiple comparisons are discussed. Finally, Linear discriminant analysis is feasible and the best method for data analysis with PCA as model selection and cross-validation as model evaluation is recommended by achieving a Training score of 0.99 and a testing score of 0.98.

1. Introduction

Within the past half-century, audio forensics has been a vital science in establishing the authenticity of audio recordings, especially in crime-related investigations. Audio forensics is the field of science related to the acquisition, analysis, and evaluation of sound recording. For crime investigations, audio forensic scientists often recreate sounds from a crime scene audio recording, record the sound on the same device that was used to record the evidence, and compare the two to determine whether the audio recording evidence had been tampered with. Audio forensics is also used to determine if a musical composition was illegally copyrighted. Musical compositions are either recorded with a mono or stereo sound format. A mono sound is when only one channel is used to convert a signal to a sound while a stereo utilizes two signals to create a depth effect. For example, when listening to audio with headphones, a mono sound system will project the same output to both earpieces. Headphones with a stereo sound system allow for more flexibility in controlling what attributes are outputted to each ear (i.e., in one earbud the artist's singing may be outputted while in the other ear only the instrumental parts of the song are outputted.

The following data set was obtained from Kaggle, a data set community with thousands of open-source datasets, and has been provided by Dr. Arunpriya, an assistant professor at PSGR Krishnammal College for women in Coimbatore, Tamil Nadu, India. The data set consists of over 26 predictor variables and one categorical response variable listed below. A total of 614 recordings, obtained from either a mono or stereo system, were analyzed and predicted to originate from a mono or stereo source. Audio Forensics | Kaggle

The variables included in this dataset which contributed to classifying the predicted response are Duration (ss): x1 - Duration of the recording, Fundamental Frequency (Hz): x2 - Lowest Frequency of a periodic wave, Mean Pitch (Hz): x3 - Average quality of a sound governed by the rate of vibrations producing it, Pitch Sigma (Hz):x4 - Standard deviation of an individual's frequency production in a given connected speech sample, Minimum Pitch (Hz):x5 - The lowest quality of a sound that is governed by the rate of vibrations producing it, Maximum Pitch (Hz):x6 - The highest quality of a sound that is governed by the rate of vibrations producing it, No. of Pulses: x7 - Number of rhythmical beatings, vibration, or sounds, No. of Periods:x8 - Number of completed cycles, Mean Periods (ss):x9 - Average time it takes for a cycle to complete, Standard Deviation of the period (ss):x10 - A measure of period dispersion compared to the mean period, Fraction of Locally unvoiced frames (%):x11 - An automated method of obtaining the percentage of a segment that is voiced, No. of voice breaks:x12 - The number of a sudden interruption of speech or a sudden decrease in vocal amplitude, Degree of voice breaks:x13 - Intensity of the voice breaks, Mean autocorrelation:x14 - Average degree of similarity between a given time series and a lagged version of itself over successive time intervals, Mean noise-to-harmonics ratio:x15 - Average measure that quantifies the amount of

additive noise in the voice signal, Mean harmonics-to-noise ratio (dB):x16 - Average measure that quantifies the amount of additive noise in the voice signal, Peak Amplitude Channel 1 (dB):x17 - Maximum positive or negative deviation of a waveform from its zero-reference level for channel 1, Peak Amplitude Channel 2 (dB):x18 - Maximum positive or negative deviation of a waveform from its zero-reference level for channel 2, Bit Rate (KBS):x19 - The number of bits that are conveyed or processed per unit of time, Sample Rate (Hz):x20 - The number of samples that are conveyed or processed per unit of time, Resolution (bits):x21 - Number of bits per sample, No of Samples:x22 - Total number of samples, Size on memory (kB):x23 - Size on memory in kilobytes and Size on disk (MB):x24 - Size on disk in megabytes. The response variable is Channels (Stereo or Mono).

2. Data Cleaning

The data set from Kaggle needed data cleaning prior to the development of models. The following subsections provide a detailed report on how the dataset was improved for model applications.

2.1 Creating Dummies for Categorical Variables

The audio forensic data set used for this project includes a categorical response variable. To utilize classification models, the instances of the response variable were converted to a binary zero-one dummy variable. A zero response indicates a mono source and one response a stereo source. A label encoder function from the library sklearn was used to convert the categorical response variables to binary instances. Furthermore, upon further investigation of the dataset, it was determined that the Sample Rate entries may also be considered a categorical variable, despite entries being numerical values (48000 and 44100). The pandas function get dummies converts entries to indicator or dummy variables is used here.

2.2 Elimination of Variables

The first and last variables (File name and source) were removed from the data set because the file name was simply an index, and the source was a link to the file name. Neither contributed to the prediction of classifying the recordings. In addition, three of the file names (rows) were removed because the majority of their corresponding values were missing. Furthermore, from the correlation matrix displayed in figure 2, it was determined that the predictors', *sample rate and size of memory*, were highly correlated, indicated by a correlation of 1. Thus, the *size of the memory* column was removed to reduce variable correlation.

2.3 Replacing missing values

The data set also consisted of missing values and two methods were used to replace these missing values. The first method was replacing the missing value with the mean of the corresponding variable and the second was replacing the missing values with the median of the corresponding variable. We used mean to replace the missing values as this provided better results. In certain instances, the variable unit was included along with the value. These units were removed with the drop function to ensure only values remained throughout the dataset.

2.4 Standardization

Standardization was performed on the remaining dataset using the MinMaxScaler function because the ranges within variables were drastic and could potentially affect the future model's performance

3. Preliminary Analysis

3.1 Matrix Scatter Plot

From the scatter plots for the data set when the missing values were replaced with the corresponding variable mean, analyzing the OLS tables obtained, replacing the missing values with mean values appears to be a more efficient method to represent the missing values.

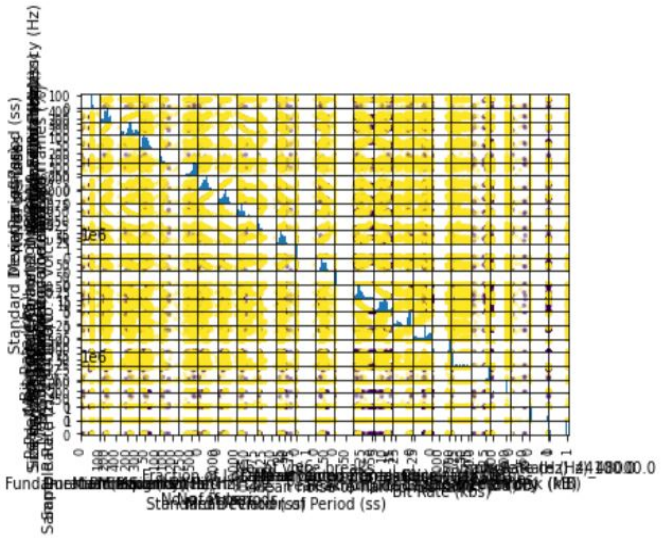


Figure 1: Matrix Scatter Plot

3.2 Correlation Matrix

From the correlation matrix displayed in Figure 2, we can observe relationships between correlating predictors.

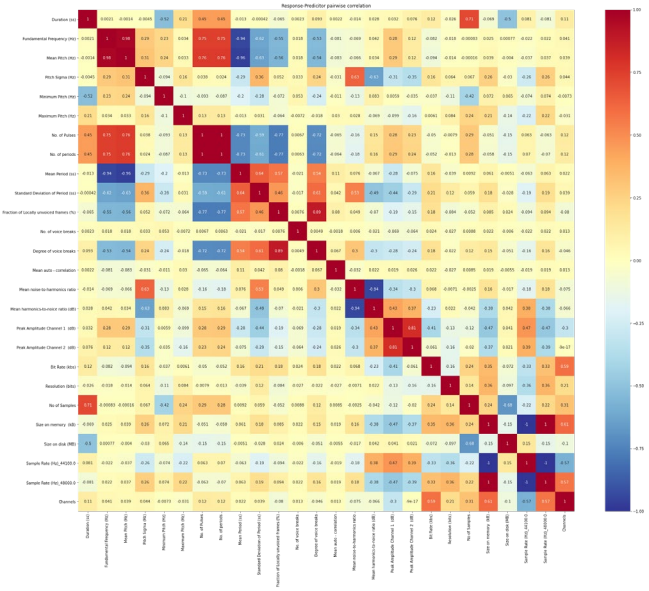


Figure 2: Correlation Matrix

There exists a high positive correlation between the following: Number of Samples – Duration, Peak Amplitude Ch. 2 – Peak Amplitude Ch. 1, Number of Voice Breaks – Mean of Locally Unvoiced Frames, Number of Periods – Mean Pitch, Number of Periods – Fundamental Frequency, Number of Pulses – Mean Pitch, Number of Pulses – Fundamental Frequency, Size on Memory – Sample Rate and Mean Pitch – Fundamental Frequency. At the same time, a high negative correlation was observed between Mean Harmonics to Noise Ratio – Mean Noise to Harmonics Ratio, Number of Voice Breaks – Number of Periods, Number of Voice Breaks – Number of Pulses, Mean of Locally Unvoiced Frames – Number of Periods, Mean of Locally Unvoiced Frames – Number of Pulses, Mean Period – Number of Periods, Mean Periods – Number of Pulses, Mean Periods – Mean Pitch and Mean Periods – Fundamental Frequency.

As a rule of thumb, a correlation factor with a value of 0.7 is considered a high correlation between two variables. Still, for the purpose of this study, we only dropped variables with a correlation of 1 representing the same information.

3.3 Data Split and transformation

The Data is divided into two subjects: training, and testing with testing set as a 0.30 ratio. Transformation on predictor observations is performed for interpreting model section techniques.

4. Modeling Techniques

The following modeling techniques were created based on the obtained mono stereo Kaggle data set: Logistic Regression, LDA/QDA, KNN, Decision Tree, Random Forest, and Support Vector Classifier. The following section provides an in-depth description of each model in addition to their respective parameters and hyper parameters.

4.1 Logistic Regression

Logistic regression is a mathematical model that is used to describe the relationship between the predictors (X) and the response variable (Y). This model is unstable when classes are well separated. It models the probability of a discrete outcome given an input variable. The logistic regression function is given as

$$f(Z) = \frac{1}{1+e^{-Z}}$$

Parameters:

- solver – the algorithm to be used in the optimization problem. Since our dataset is smaller, we used 'liblinear' as the solver.
- random_state – this parameter controls the random number generator used. We set this as 3.
- penalty – the norm of penalty is set as default ('l2').
- C - Inverse of regularization strength.
- dual – whether the formulation is desired or not. Set as default (false).
- tol – the tolerance for the stopping criteria. Set as default (1e-4).
- fit_intercept - specifies if an intercept should be added to the decision function. Set as default (true).
- max_iter – specifies the maximum number of iterations required. Set as default (100).

Hyper parameters:

There was no critical hyper parameter to be tuned in logistic regression.

Performance of the model:

- Training accuracy: 0.9274
- Testing accuracy: 0.8641

After performing cross-validation, the mean cross-validation score obtained for 10 folds was found to be 0.9320.

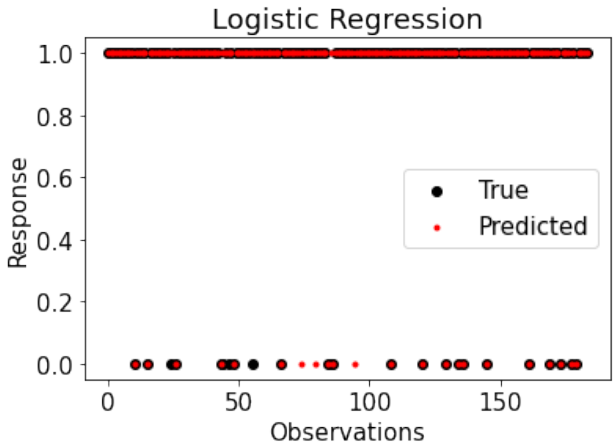


Figure 4: Logistic Regression, true values versus predicted values plot

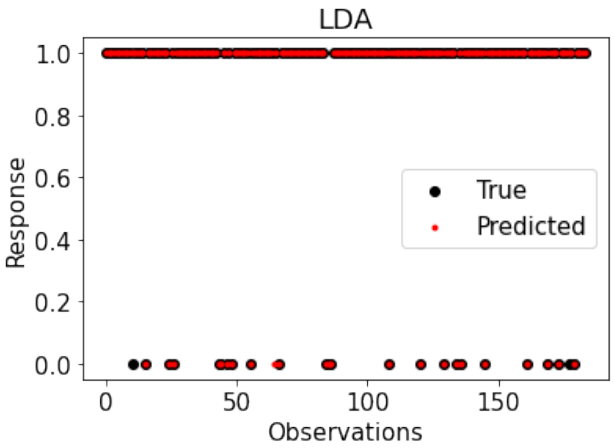


Figure 5: LDA, true values versus predicted values plot

4.2 Linear discriminant analysis (LDA)

Linear discriminant analysis is a pre-processing step in machine learning that is used to model the differences in the groups of two or more classes. It is commonly used in classification problems. It works by projecting high-dimensional features into low-dimensional features.

Parameters:

- solver – The type of mathematical method used to solve the fit. Set as default ('svd').
- shrinkage – the shrinking method used. Set as default (None).
- n_components – Number of components desired for dimensionality reduction. Set as default (None).
- tol – the tolerance desired for the stopping criteria. Set as default (1e-4).

Hyper parameters:

No critical hyper parameters were to be tuned in linear discriminant analysis.

Performance of the model:

- Training accuracy: 0.9929
- Testing accuracy: 0.9837

After performing cross-validation, the mean cross-validation score obtained for 10 folds was found to be 0.9906.

4.3 Quadratic discriminant analysis (QDA)

Quadratic discriminant analysis is similar to linear discriminant analysis. The difference is that QDA assumes that the covariance matrix can differ for each class. QDA allows more flexibility for the covariance matrix.

Parameters:

- tol - the tolerance desired for the stopping criteria. Set as default (1e-4).
- reg_param – regularization parameter. Set as default (0).

Hyper parameters:

There were no critical hyper parameters to be tuned in Quadratic discriminant analysis.

Performance of the model:

- Training accuracy: 0.0702
- Testing accuracy: 0.8863

After performing cross-validation, the mean cross-validation score obtained for 10 folds was found to be 0.0702.

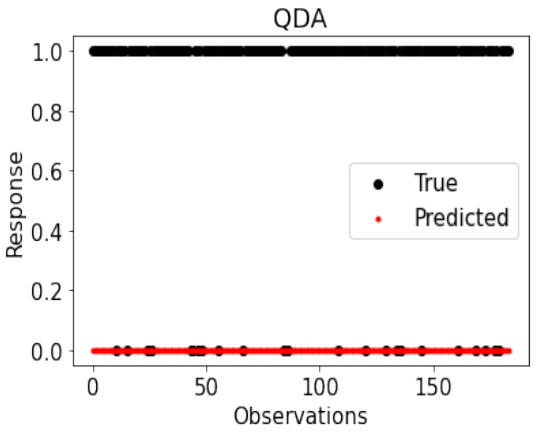


Figure 6: QDA, true values versus predicted values plot

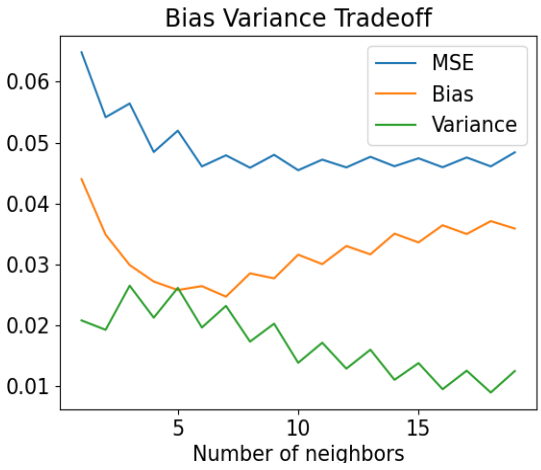


Figure 7: KNN, 'K' versus Bias, Variance, MSE plot

4.4 K-nearest neighbors (KNN)

K- Nearest neighbors is a supervised learning algorithm used to perform extremely complex classification problems. The methodology of KNN is one of the simplest of all machine learning methodologies. The prediction is made by computing the distance of a test data point with the 'n' number of neighboring data points. In this project, the Euclidean distance calculation method is used.

Parameters:

- n_neighbors: The number of neighbors the algorithm should consider before deciding the class of the new data point. The number of neighbors we selected was 5 since n_neighbors=5 returned a high training and testing accuracy score.
- weights: Weight function used in prediction. Set as default ('uniform').
- algorithm: The algorithm used to estimate the nearest neighbors. Set as default ('auto').
- leaf_size: This parameter influences the speed of the construction and query. Set as default (30).
- p: Power parameter of the Minkowski metric. Set as default (2).
- metric: The metric is used for distance computation. Since p=2, it is set as 'Euclidian'.

Hyper parameters:

The model was tuned based on n_neighbors and the highest accuracy was achieved when using n_neighbors = 5.

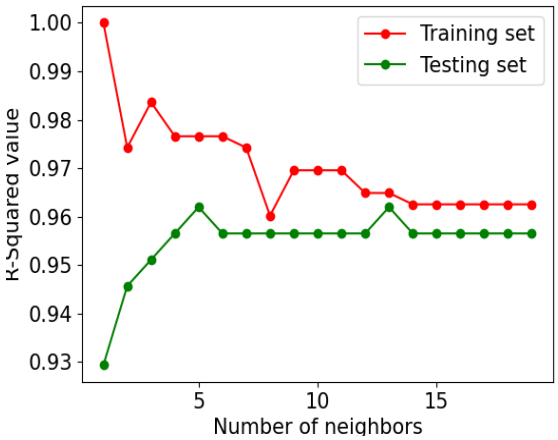


Figure 8: KNN, 'K' versus training and testing scores plot

Performance of the model:

- Training accuracy: 0.9648
- Testing accuracy: 0.9619

After performing cross-validation, the mean cross-validation score obtained for 10 K- folds were found to be 0.9601.

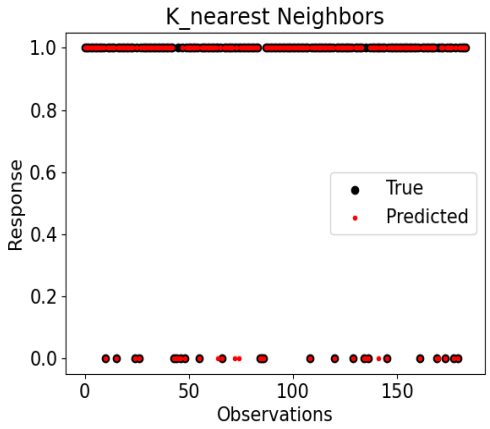


Figure 9: KNN, true values versus predicted values plot

Figure 9 shows the relation between true and predicted values. It shows from the plot that the dataset has a lot of instances whose value is 1 and even it is predicting as 1 in most cases.

4.5 Decision tree classifier

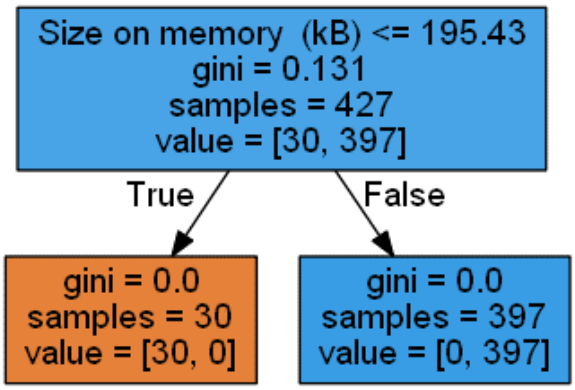
A decision tree classifier can be used for both classification and regression problems. It is commonly used for image classification, strategy analysis, and more. The classifier splits the data down into subsets and then again split into multiple branches or partitions.

Parameters:

- criterion - The function to measure the quality of a split. Set as default ('gini').
- splitter - The strategy used to choose the split at each node. Set as default ('best').
- max_depth - The maximum depth of the tree. The value is provided based on the value of maximum depth that provides the best accuracy. In our project, it is set as 1 as the simplest case.
- min_samples_split - The minimum number of samples required to split an internal node. Set as default (2)
- max_features - The number of features to consider when looking for the best split. Set as default (None).
- random_state - This parameter controls the random number generator used. We set this as 3.

Hyper parameters:

The model was tuned based on criterion, max_depth, and min_samples_split. The best performance was achieved when criterion = gini, max_depth = 7, and min_samples_split = 3.



- Training accuracy: 1.0
- Testing accuracy: 0.8967
- After performing cross-validation, the mean cross-validation score obtained for 10 folds was found to be 0.9983.

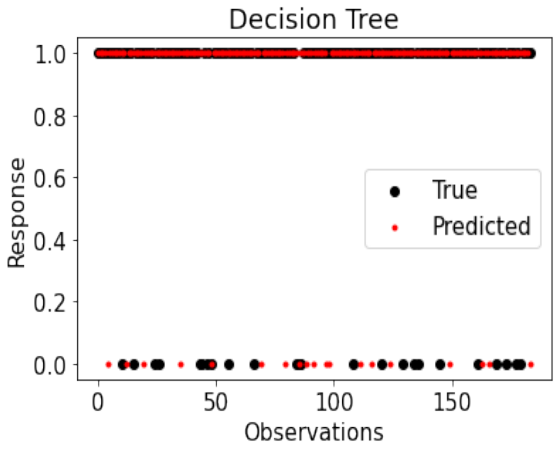


Figure10: Decision tree, true values versus predicted values plot

4.6 Random forest classifier

Random forest classifier is used for both classification and regression problems and is the most popular algorithm as it offers more flexibility and ease of implementation of the model. Random forest classifier is basically a collection of multiple decision tree. It prevents overfitting. Random forest classifier creates various decision trees based on the random selection of data samples and procures predictions from each decision tree.

Parameters:

- N_estimators - The number of trees in the forest. Set as default (100).
- criterion - The function to measure the quality of a split. Set as default ('gini').
- max-depth - The maximum depth of the tree. The value is provided based on the value of maximum depth that provides the best accuracy. In our project, it is set as 1 as it provides the best accuracy.
- min_samples_split - The minimum number of samples required to split an internal node. Set as default (2).
- random_state - This parameter controls the random number generator used. We set this as 3.
- max_features - The number of features to consider when looking for the best split. Set as default (None).

Hyper parameters:

The model was tuned based on n_estimators, criterion, max_depth, and min_samples_split. The best performance was achieved when n_estimators = 150, criterion = entropy, max_depth = 1, and min_samples_split = 5.

Performance of the model:

- Training accuracy: 0.9859
- Testing accuracy: 0.9402

After performing cross-validation, the mean cross-validation score obtained for 10 folds was found to be 0.9813.

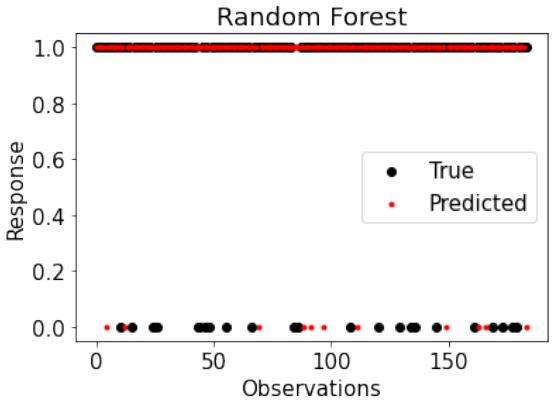


Figure 11: Random forest, true values versus predicted values plot

4.7 Support Vector Classifier

Support vector machine (SVM) analyzes the data for classification and regression analysis. It is a supervised learning method that looks at the data and sorts it into one of two categories.

Parameters:

- C: Regularization parameter. The strength of the regularization is inversely proportional to C. It is set equal to 1.
- kernel: Specifies the kernel type to be used in the algorithm. The kernel is specified as 'rbf'
- gamma: Kernel coefficient. Set to 'scale' for the kernel.

Hyper parameters:

Due to the substantially long computation time for determining the optimum parameters, the default values were used.

Performance of the model:

- Training accuracy: 0.9297
- Testing accuracy: 0.8695

After performing cross-validation, the mean cross-validation score obtained for 10 folds was found to be 0.9204.

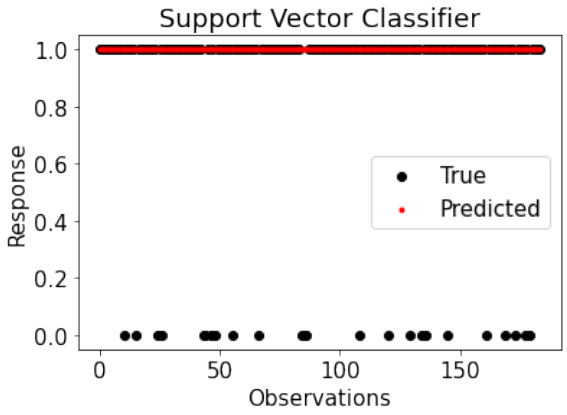


Figure 12: Support vector machines, true values versus predicted values plot

5. Model selection

5.1 Ridge Classification

It is a shrinkage methodology that converts the label data to -1, and 1, and then performs regression on it. This type of regression is suitable for models having high

multicollinearity. Ridge methodology prevents overfitting. Ridge simplifies the model, thus reducing the variance with the data.

Parameters:

- alphas - List of alphas where to compute the models. In our project, we provided a set of values from 0 to 2000.
- fit_intercept – Decides whether to calculate the intercept. Set as default (True).
- max_iter - The maximum number of iterations. Set as default (1000).
- tol - The tolerance for the optimization. Set as default (1e-4).
- random_state - This parameter controls the random number generator used. We set this as 3.

Hyper parameters:

The optimal value for hyper parameter 'alpha' is obtained as 0.0250. This value is obtained by the ridge algorithm by performing cross-validation. The value obtained was for a cross-validation value of 3.

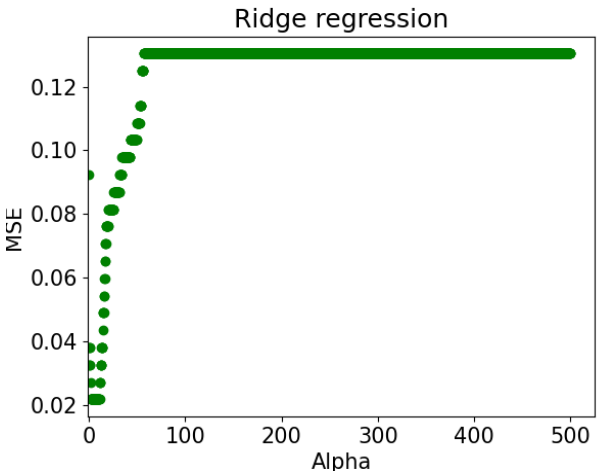


Figure 13: Ridge, alpha versus mean square error(MSE) plot for alpha tuning

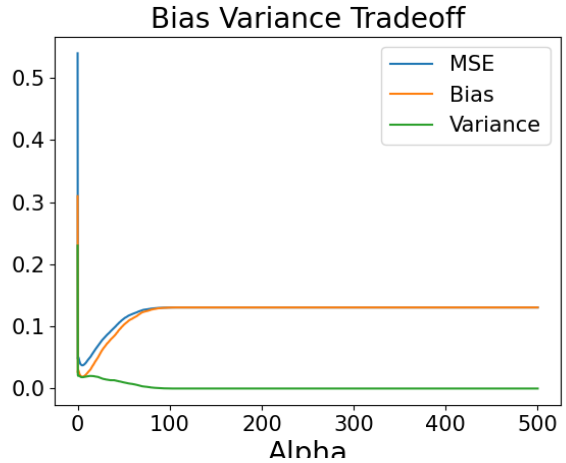


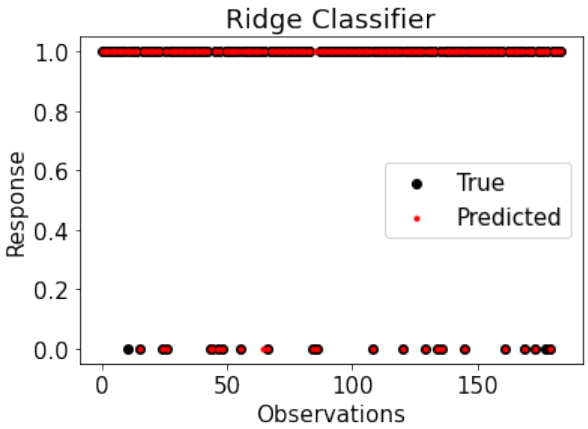
Figure 14: Ridge, alpha versus MSE, bias, variance plot for alpha tuning

Performance of the model:

- Training accuracy: 0.9953
- Testing accuracy: 0.9673
- The best alpha value: 0.2501

After performing cross-validation, the mean cross-validation score obtained for 10 folds was found to be 0.9929.

Figure:15



.Ridge, true values versus predicted values plot

5.2 Principal Component Analysis (PCA) and LDA

PCA analysis is a dimension reduction technique used when the factors are large as well as highly collinear. It can also be used to model multiple outcome variables. PCA transformation is supervised. PCA allows for the reduction of the dimensionality of the correlated variables by reducing the dimensionality of the larger data set. In our

project, we used PCA along with the LDA model as the base model to determine the optimal number of principal components.

Parameters:

- n_components - Number of components to keep.
- svd_solver - The solver is selected by a default policy based on X.shape and n_components. Set as default ('auto').
- tol - The tolerance for the optimization. Set as default (0.0).
- copy - Whether to copy X and Y in the fit before applying centering, and potentially scaling. Set as default (True).
- random_state - This parameter controls the random number generator used. We set this as 3.

Hyper parameters:

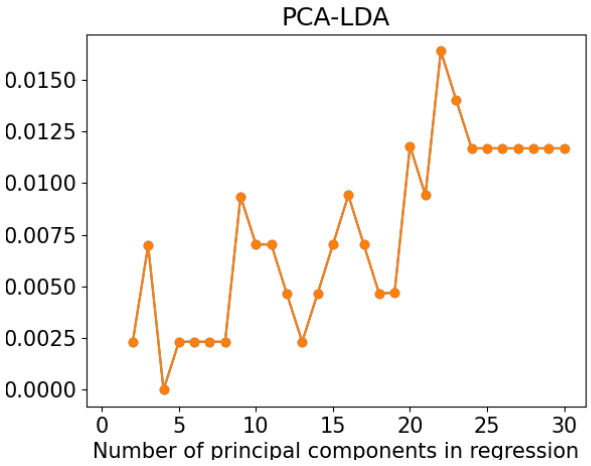


Figure 16: PCA-LDA, principal components in regression versus test MSE

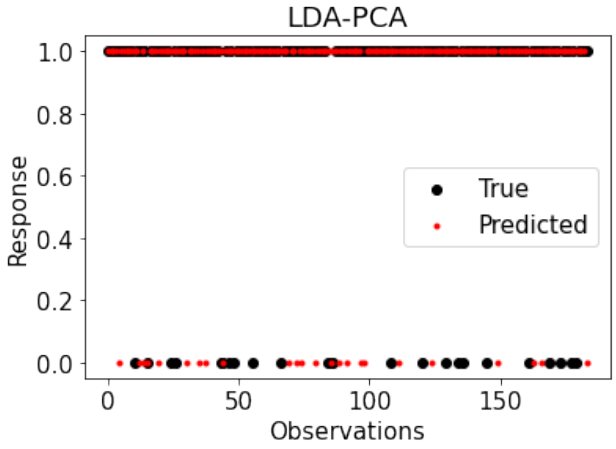


Figure 17: True values versus predicted values plot for LDA-PCA combination

The optimal value for hyper parameter 'n_components' is obtained as 4. This value is obtained by performing an algorithm by performing k-fold cross-validation.

Performance of the model:

- Training accuracy: 0.9976
- Testing accuracy: 0.9837

After performing cross-validation, the mean cross-validation score obtained for 10 folds was found to be 0.9953.

5.3 Best overall model

The best overall model is PCA-LDA. This conclusion is reached based on the fact that the PCA-LDA model has the lowest test error as well as the highest testing set accuracy score among all the models.

- Accuracy score of training set: 0.9976
- Accuracy score of testing set: 0.9837

A dimension reduction technique (PCA) provides better prediction by reducing the dimensionality of larger datasets into small dimensions. This reduction has resulted in higher testing and training accuracy.

Conclusion

In this study, different machine learning techniques were used to predict whether a given audio sample is Mono or Stereo. The dataset was found to be linear and highly correlated.

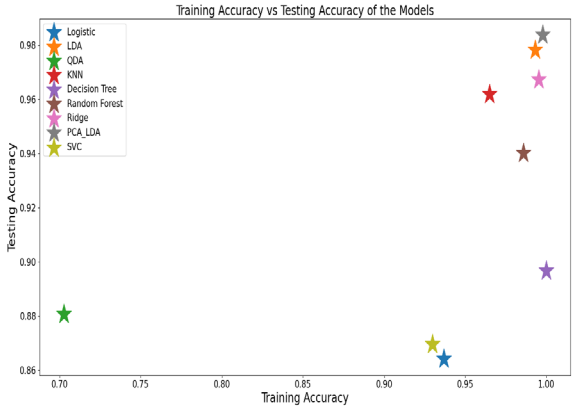


Figure:18. Training versus testing score for all the model techniques plot

From the figure shown below, it was observed that among all the models tested PCA-LDA model gives the best prediction with a testing accuracy of 0.9837 while Linear Discriminant Analysis (LDA) performs prediction with a testing accuracy of the same value and Partial Least Squares performed the worst with a testing accuracy of 0.5757. KNN and Random Forest Classifiers performed well as well with a testing accuracy of 0.96195 and 0.94021. The testing accuracy of Random Forest can be made better by providing the appropriate value for the number of estimators.