MISTY GROUP
Mohamed Abdelaziz k12137202
Lydia Mayer k11904969
Ivan Drinovac k12104744

Birds Audio Classification

Machine Learning & Pattern Classification [UE]

SYSTEMATIC CLASSIFICATION EXPERIMENTS

Table of Contents

1. Data Splits

Stratified Cross Validation:

- Stratified cross-validation is a technique commonly employed in machine learning to assess the performance of a model on a dataset with class imbalance or skewness. It involves partitioning the dataset into several folds while preserving the original class distribution in each fold. The aim is to ensure that each fold represents the same proportion of classes as the original dataset.
- The reason stratified cross-validation is particularly important in skewed datasets is that it helps mitigate the potential bias that can arise when evaluating model performance. In a skewed dataset, where one class significantly outweighs the others, a standard random partitioning of data into folds can result in some folds containing an insufficient representation of the minority class. As a result, the model may not be exposed to enough examples of the minority class during training, leading to poor performance and inaccurate evaluation metrics.
- In our experiments we have tried out different number of folds to identify the best results based on the log loss and the balanced accuracy, and utilized 16 folds in general since it gave us the best balance of "scores vs training time"

2. Features

2.1 Preprocessing

Procedure:

- **General:**
  - Transformed both inputs and outputs "Separate files" into a single data-frame for easier analysis/training.
- **Features:**
  - Normalized the Features (independently) in all of the dataset from [0,1] to better stabilize the model numerically. [Fig. 1]
  - We then eliminated the Outliers by:
    - Calculating the Z-Score for each feature independently.
    - Used a threshold of 0.99 to identify how many outliers exists outside this boundary.
    - [For Some Features] we carefully selected those outliers and replaced their values with the maximum value of the rest of the samples.
- **Labels:**
  - We first Calculated the Absolute Frequencies for each row with length 7 [5,5,0] -> [1,0,0,0,2,0,0]
  - We then Derived the Relative Frequency for each Absolute Frequency, [1,0,0,0,2,0,0] -> [0.33, 0, 0, 0, 0.67, 0, 0] for a standard representation.
  - Furthermore, Separately, we selected the highest probability of each row to be regarded as the class of the sample, since some of the model classes utilizes the decision boundary instead of the class probabilities

2. Features

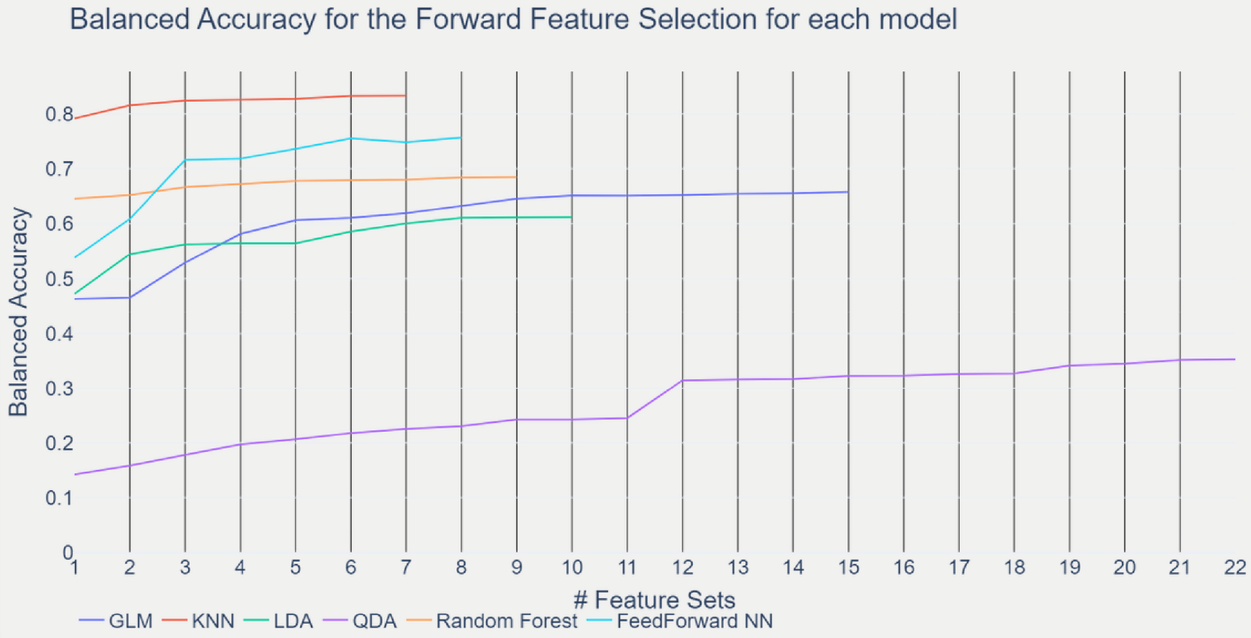
2.2 Feature Selection

Procedure:

- Since the model classes have different behaviors based on the features' characteristics:
  - We implemented Automatic Forward Selection Mechanism based on the feature sets [for each model class separately], governed by the balanced accuracy metric to determine which of the feature sets increases the accuracy.
  - we then used the previous knowledge about the redundant features and experimented with removing some of the redundant features in each selection, which significantly reduced the feature space.
- Using PCA, for each Feature Space independently, we experimented with reducing the dimensionality of some of the feature sets selected.

Conclusion:

- We ended up with 2 different selections of feature-sets for each model-class:
  - The first dataset utilizes the whole feature-space of each of the selected feature sets.
  - The second dataset has approximately 25% of the number of features of the selected feature sets.
- Furthermore, we experimented with balancing the dataset by randomly selecting 10000 samples from the "other" class and dropping the rest of the samples in that class to study the effect on various algorithms.
- In Total we ended up with approx. 25 different datasets varying in size.



- Feature sets vary among different model classes, [will be mentioned in the respective slides].

3. Evaluation Metrics

3.1 Metrics Used

- Negative Log Likelihood Loss [0, inf]
  - The negative log likelihood loss is implemented by taking the negative logarithm of the predicted probability of the true class and summing it over all the classes
  - It is commonly used in multiclass classification tasks because it encourages the model to assign higher probabilities to the correct class and penalizes incorrect predictions more severely.
- Balanced Accuracy [0,1]
  - Balanced accuracy is a performance metric that calculates the average of the sensitivity and specificity of a classifier across all classes.
  - By using the balanced accuracy, the performance of the classifier is evaluated more fairly across all classes, as it takes into account the relative importance of each class, which is more suitable for the skewed datasets.
- Confusion Matrix
  - The confusion matrix score is a performance metric that summarizes the classification results in a tabular form by counting the number of true positives, true negatives, false positives, and false negatives for each class in a multiclass classification problem.
  - In a skewed dataset, where the class distribution is imbalanced, the confusion matrix score provides valuable insights into the model's performance on each class, allowing us to identify which classes are being misclassified more frequently and to adjust the model or the evaluation strategy accordingly to address the imbalance.
- Precision/Recall [0, 1]
  - Precision: measures the proportion of true positives (correctly predicted positive samples) among all positive predictions made by a classifier, and it works by dividing the number of true positives by the sum of true positives and false positives.
  - Recall: measures the proportion of true positives (correctly predicted positive samples) among all actual positive samples in a dataset, and it works by dividing the number of true positives by the sum of true positives and false negatives.
- Weighted F1 Score [0, 1]
  - The weighted F1 score is a performance metric that calculates the harmonic mean of precision and recall, taking into account the imbalance of the classes in a multiclass classification problem by weighting the score of each class by the number of samples in that class, and it works by computing the F1 score for each class and averaging them, weighted by the number of samples in each class.

3. Evaluation Metrics

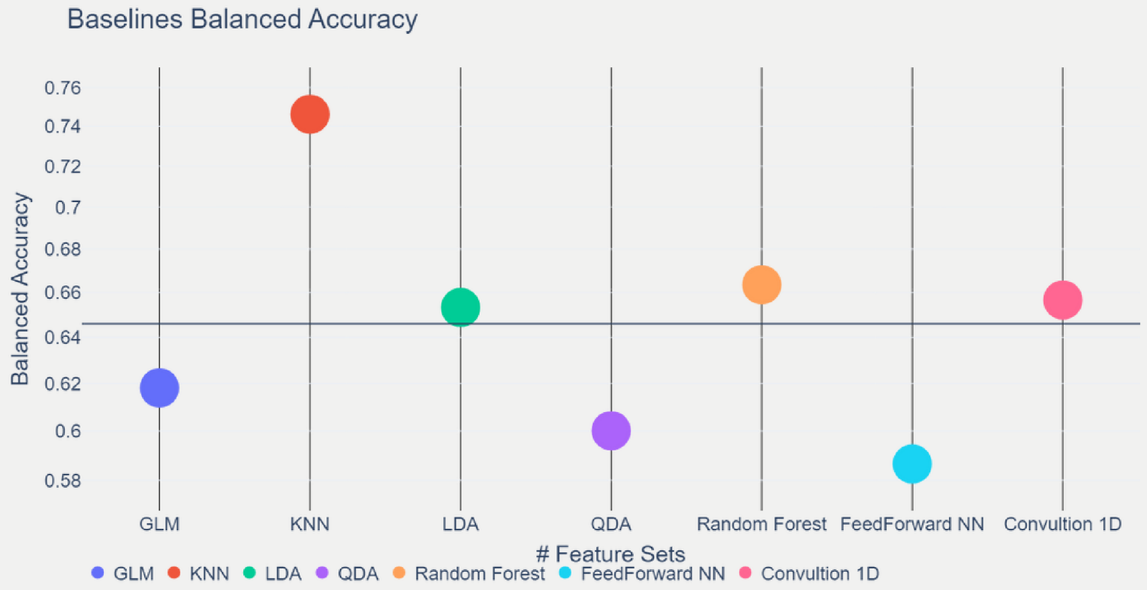
3.2 Baseline and Best Performance

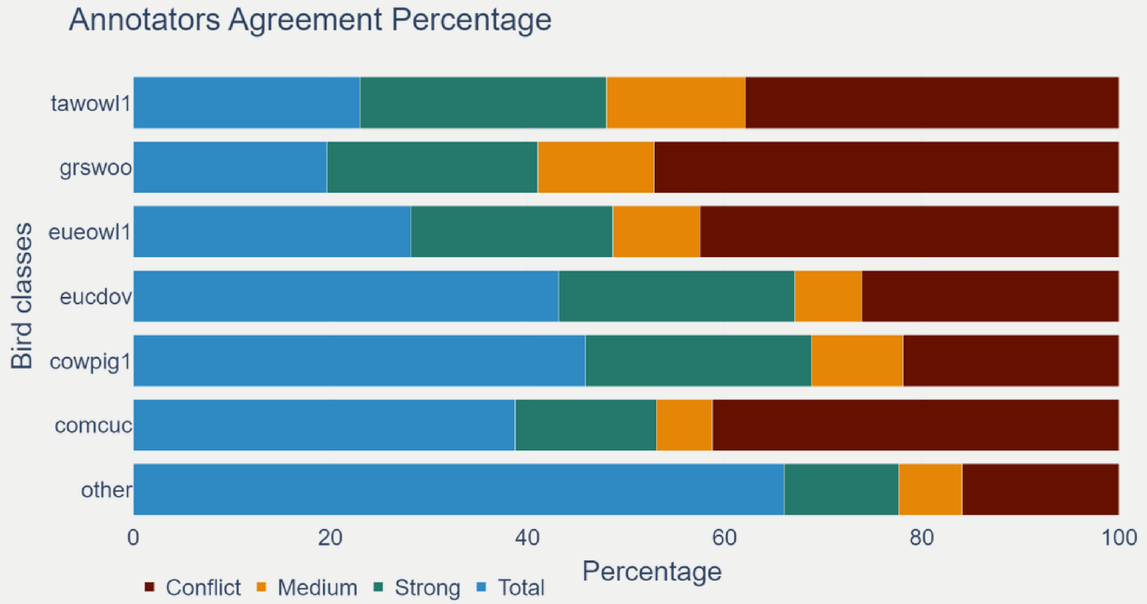
Baselines:

- We trained each model class on the full dataset with limited preprocessing, and generic hyperparameters, in order to find the baseline for each model-class separately.
- Then we calculated the average of each metric across all model classes to come up with a more reasonable baseline:
  - 68% Log Likelihood Loss [The lower the better]
  - 64% Balanced Accuracy. [The higher the better]
  - 63% Mean Average Precision. [The higher the better]
  - 80% Average Weighted F1 Score [The higher the better]

Best Performance:

- Since it is a fairly non-complex classification task, we guess that the best performance can reach the best limit of each metric.
- From our analysis in our previous report, we have discovered that the average annotation agreement was around 75% which can impact the final decision of any model class to some extent, in real life applications.





4. Experiments

General Procedure:

- In general we tried to unify the procedure of most of the experiments and the evaluation metrics used, in order to fairly compare the different model-classes.
- We tried in the first steps of each experiment to focus on studying the effects of certain hyperparameters independently, by fixing all the other hyperparameters and only observe the effects of changing a single hyperparameter at a time.
- Automatic forward feature selection mechanism was then applied accompanied by our previous knowledge of redundancy and correlation of the feature sets to select some feature sets and study their effect on the models being trained, either to mitigate the bias-variance trade-off, boost the performance, or reduce the training time.
- In some of the hyperparameters, that we know it can cause the model to overfit or underfit we implemented bisection method to find the best values for these hyperparameters.
- Further more we performed a simple confusion matrix analysis for the best model for each model-class over unseen test sets and generally it was found that it was acceptable

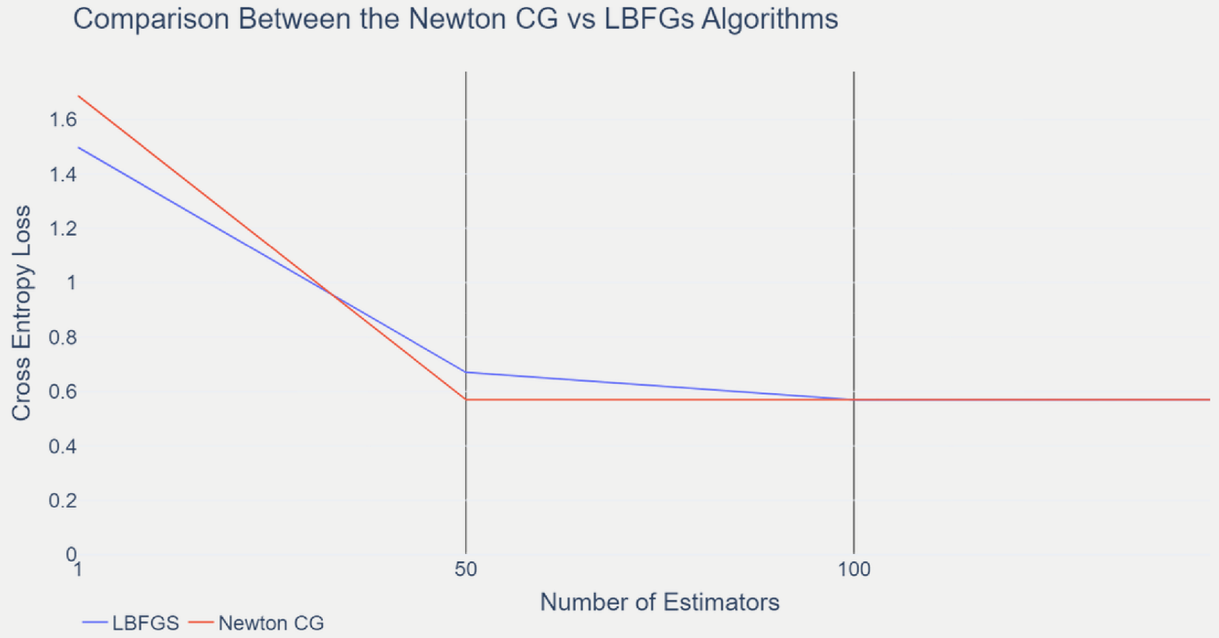
[removed from the report to stick with the 20-page length]

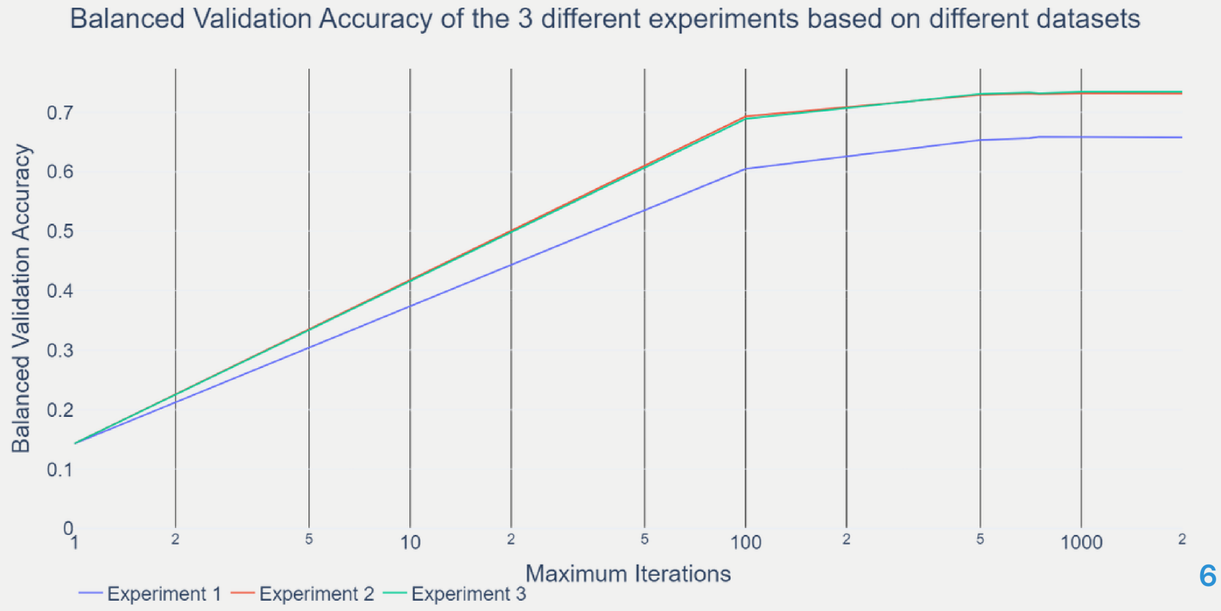
4. Experiments

4.1 General Linear Model

Procedure:

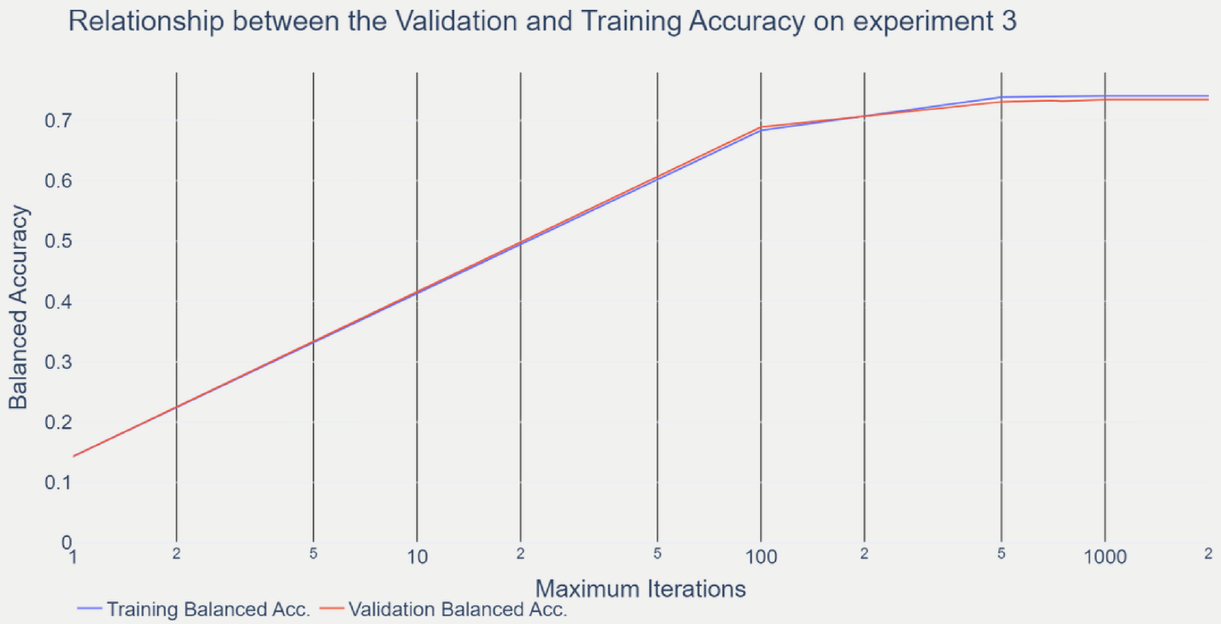
- **Step 1:** We first trained 8 different models using the whole dataset with different hyper parameters, 4 models utilizing the Limited-memory Broyden–Fletcher–Goldfarb–Shanno solver "LBFGS" and for the other 4 models we used the Newton Coordinate Descent Algorithm "Newton CG" to study the impact on performance for these 2 solvers.
- **Step 2:** Then we used the LBFGS Algorithm with 1000 Maximum Iterations in a forward search algorithm, accompanied by our previous heuristics, mentioned in the feature selection section, to find the best feature sets to use:
  - 'raw_melspect_mean', 'raw_mfcc_mean', 'raw_melspect_std', 'cln_melspect_mean', 'cln_melspect_std', 'cln_mfcc_mean', 'cln_contrast_mean', 'raw_contrast_mean', 'raw_mfcc_std', 'centroid_mean', 'energy_std', 'flatness_mean', 'bandwidth_std', 'flatness_std', 'centroid_std', 'flux_std', 'flux_mean', 'energy_mean'
- **Step 3:** Using the Bisection method to estimate the best hyperparameter for the "Maximum Iterations" on three different datasets:
  - Selected Features dataset [12000 x 362]
  - Selected Features with balanced classes dataset [69796 x 362]
  - PCA-reduced Selected Features with balanced classes dataset [69796 x 138]

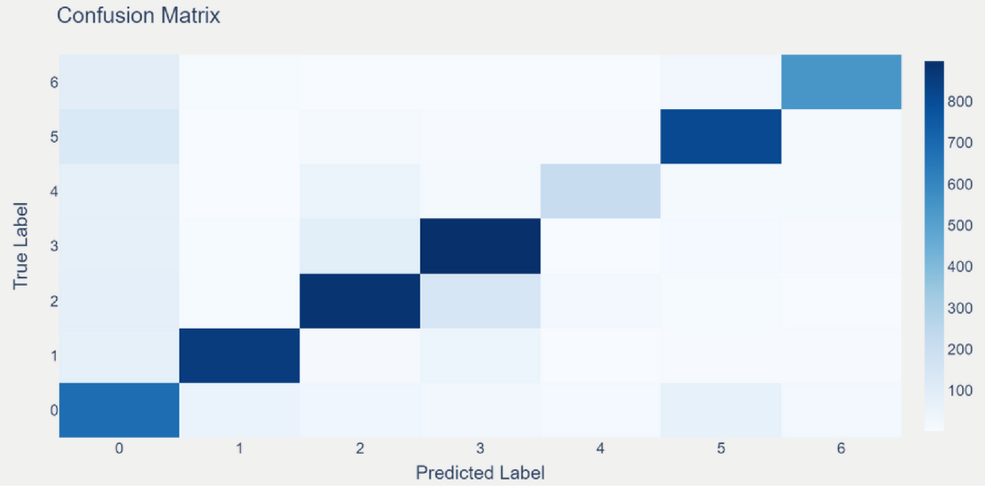




4. Experiments

4.1 General Linear Model





Conclusion:

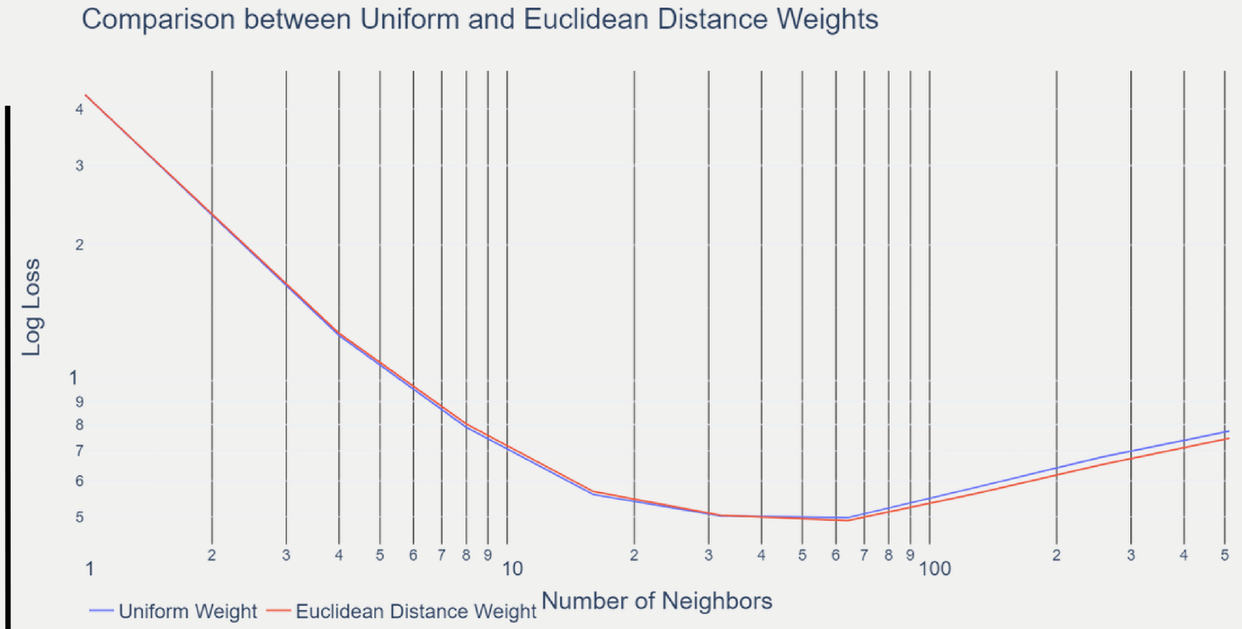
- It was noticed that the performance of the lbfgs and newton cg algorithms as optimizers are very similar according to different evaluation metrics especially after a threshold of 100 iterations.
- As for the experiments, The imbalance of the dataset clearly had an impact on the overall performance of the models, as it was shown that in the plots, the full dataset appeared to have the lowest performance at around 72% Balanced Accuracy while the balanced datasets showed almost 74% Balanced Accuracy, and 0.68 Cross Entropy Loss and 0.8 F1 Score, which has surpassed our baseline.
- It was also noticed that the performance of the balanced dataset with PCA Reduced selected features is almost identical to the performance of the balanced dataset with selected features, which further supports the hypothesis in our previous submission about the high inter-features correlation.
- Overall all the models suffered from some sort of underfitting, since they surpassed our baseline yet did not achieve the best performance, and this is perhaps due to the simplicity of the general linear models.
- By observing the Confusion matrix it was found that the model is giving more importance to the other class even on the balanced dataset as it consistently made more false positives, also there was some sort of confusion between classes 2 and 3 which can reflect similar features' values among these 2 classes.

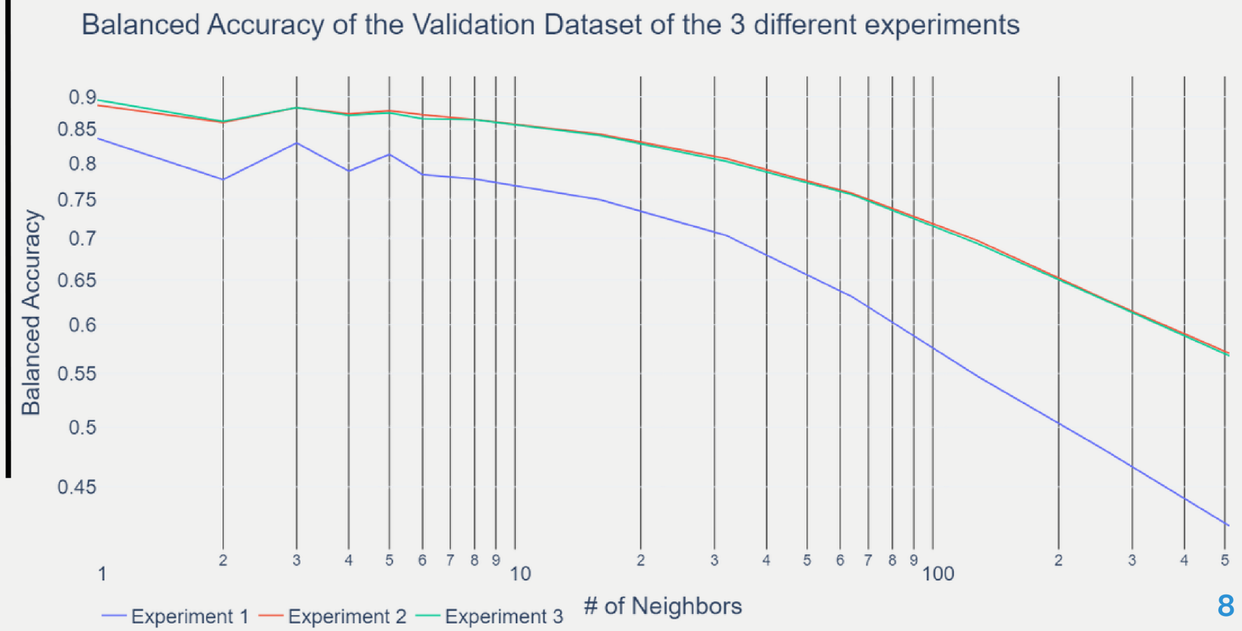
4. Experiments

4.2 K-Nearest Neighbors

Procedure:

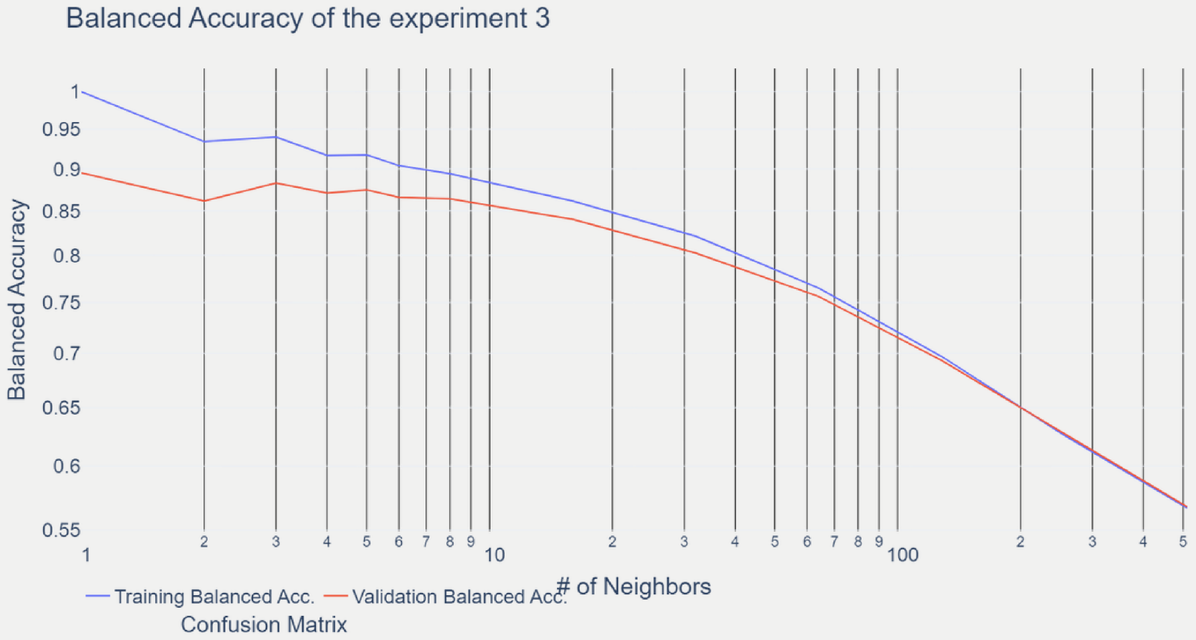
- **Step 1:** We First trained 20 different models using the whole dataset without feature selection, 10 models utilizing Uniform weights and 10 other models using the Euclidean distance weights applied to the predictions, to study the effect of changing the weights function.
- **Step 2:** Automatic Forward Feature Selection Mechanism was adopted, accompanied by our previous knowledge of redundant features, to figure out the best feature sets that fits the model best:
  - 'raw_melspect_mean', 'cln_melspect_mean', 'raw_contrast_mean', 'zcr', 'centroid_mean', 'flatness_mean', 'flatness_std', 'bandwidth_mean', 'flux_std'
- **Step 3:** The Bisection method was then used to estimate the best hyperparameter for the "Number of Neighbors" for the uniform weighted KNN, on three different datasets as follows:
  - Selected Features dataset [12000 x 152]
  - Selected Features with balanced classes dataset [69796 x 152]
  - PCA-reduced Selected Features with balanced classes dataset [69796 x 57]

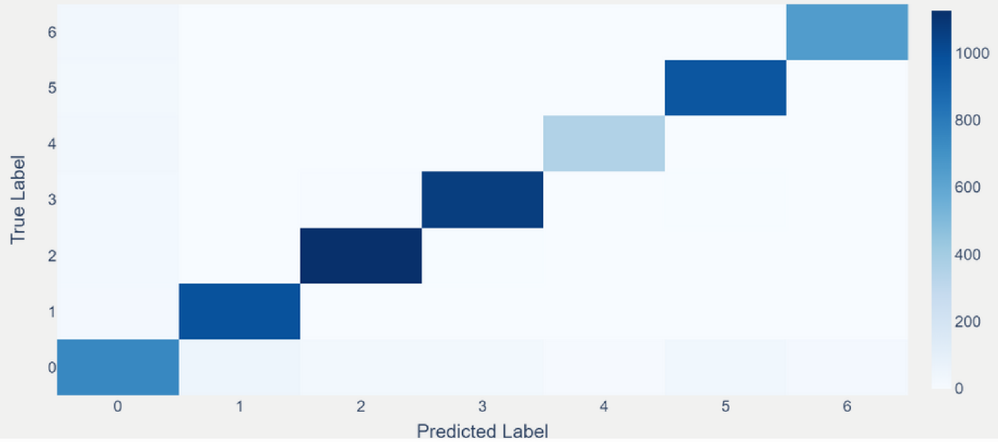




4. Experiments

4.2 K-Nearest Neighbors





Conclusion:

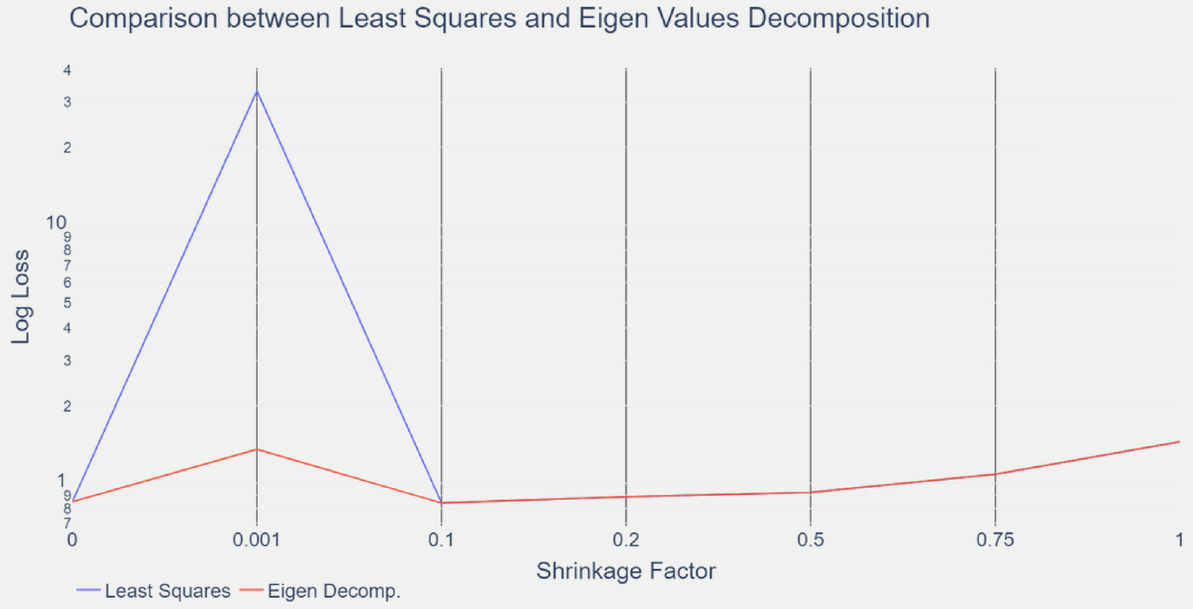
- It was observed that the performance and the behavior between the uniform weighted KNN and Euclidean distance weighted KNN was not noticeably different.
- The First experiment using the skewed dataset showed worse performance than the second and third with a difference of around 10% in the best case.
- Using PCA-reduction on the balanced dataset did not improve the performance, yet it remained almost identical to the second experiment, with much fewer features.
- Analyzing the plot, one can clearly see that there exists overfitting with low number of neighbors, and underfitting with high number of neighbors, which is a very expected behavior for KNN model class.
- The best model performance in our opinion that best mitigates the overfitting and underfitting behavior of the KNN models was the 3 neighbors as hyper parameters sitting at 94% balanced accuracy for the training sets and 89% balanced accuracy for the validation sets.
- Overall the confusion matrix showed some good performance, although the 0 class showed more false positives and false negatives.

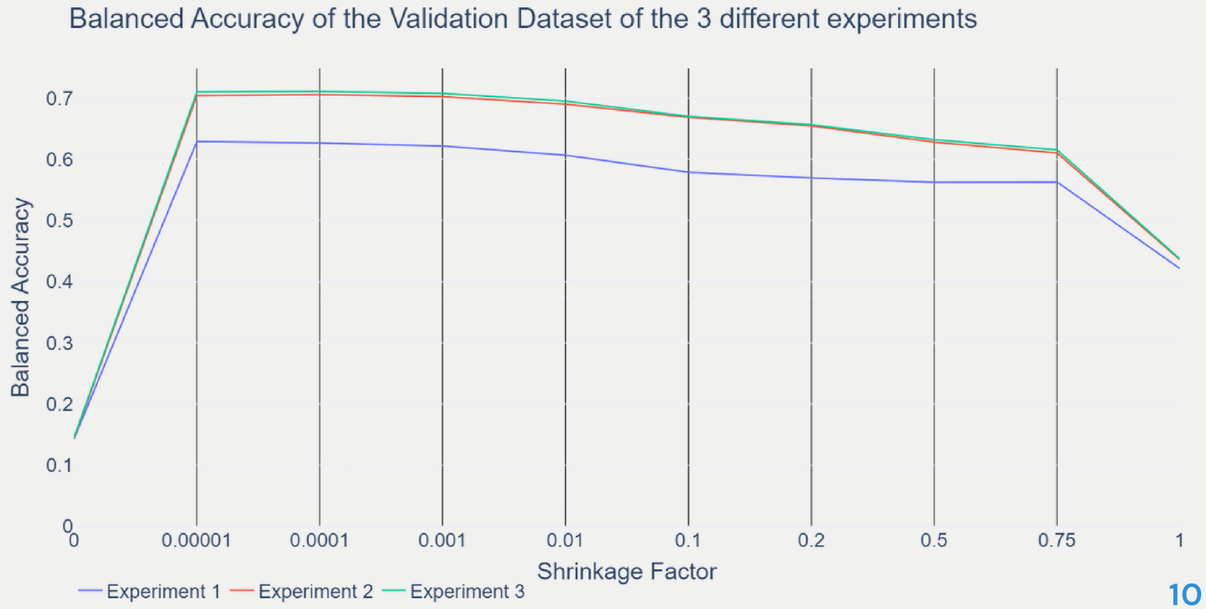
4. Experiments

4.3.1 Discriminant Analysis [Linear]

Procedure:

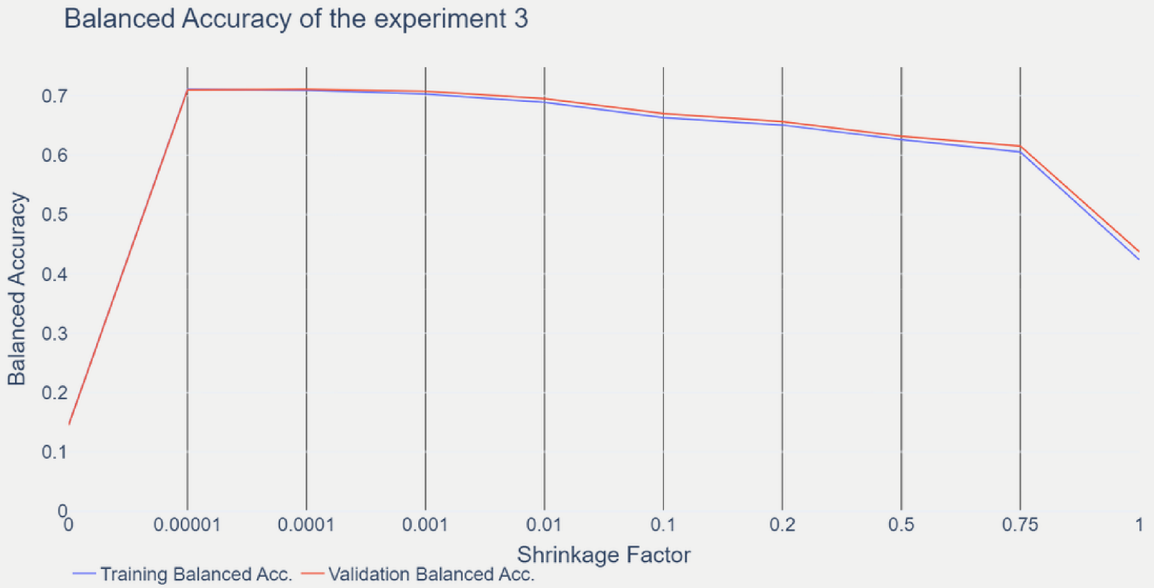
- **Step 1:** We First trained 14 different models using the whole dataset without feature selection, 7 utilizing Least Squares "LSQR" and 7 utilizing the Eigenvalue decomposition as solvers to identify the effect on selecting either of them as solvers, regardless of the shrinkage factor.
- **Step 2:** Automatic Forward Feature Selection Mechanism was adopted to figure out the best feature sets that fits the model best:
  - 'raw_melspect_mean', 'cln_melspect_mean', 'raw_melspect_std', 'cln_melspect_std', 'cln_mfcc_mean', 'cln_contrast_mean', 'raw_mfcc_std', 'cln_mfcc_d2_mean', 'flux_std', 'power_std', 'yin'
- **Step 3:** Then we adopted the Bisection method to estimate the best hyperparameter for the "Shrinkage factor" using the least squares on three different datasets:
  - Selected Features dataset [12000 x 334]
  - Selected Features with balanced classes dataset [69796 x 334]
  - PCA-reduced Selected Features with balanced classes dataset [69796 x 129]

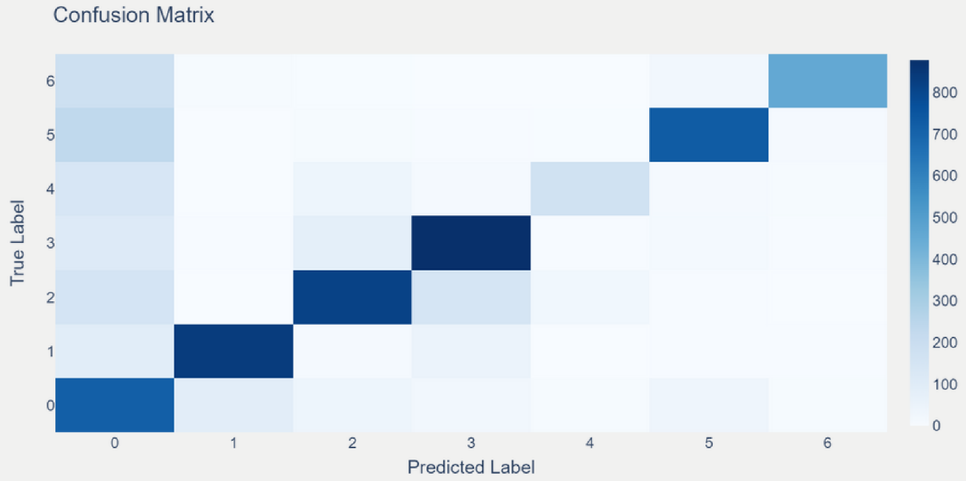




4. Experiments

4.3.1 Discriminant Analysis [Linear]





Conclusion:

- The selection of LSQR and Eigen value decomposition as solvers impacted neither the training nor the validation of any of the selected hyperparameters, except for a shrinkage factor of 0.001.
- In the initial experiment that used a dataset with a skewed distribution, the performance was observed to be comparatively poorer in comparison to the second and third experiments. The discrepancy in performance, measured at its best, amounted to approximately a 10% variation.
- Despite applying PCA-reduction to the balanced dataset, there was no noticeable improvement in performance. However, the performance achieved remained nearly identical to that of the second experiment, even though a significantly reduced number of features were utilized.
- During the analysis of the performance exhibited by the best model on the balanced dataset after applying PCA for dimensionality reduction, an observation was made indicating the presence of underfitting to some extent. However, it is worth noting that despite this underfitting issue, the model managed to surpass the baseline scores.
- The Confusion matrix shows that the 0 class is somehow falsely predicted across all classes, even when training on a non-skewed dataset. further more we can observe the same behavior of a small confusion between predicting the second and the third classes.

4. Experiments

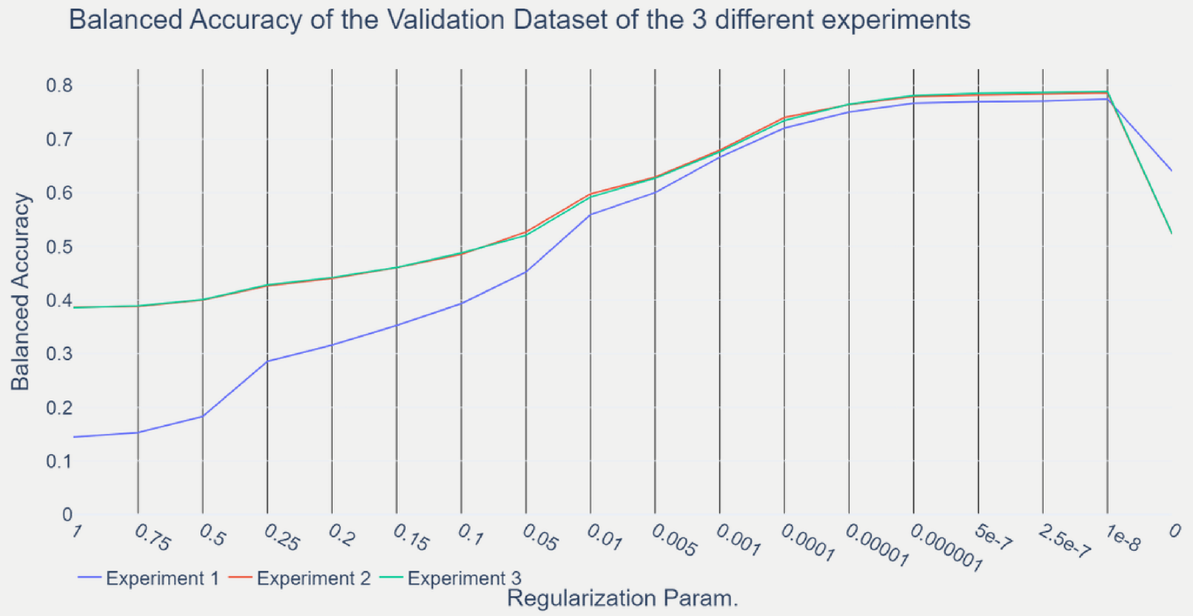
4.3.2 Discriminant Analysis [Quadratic]

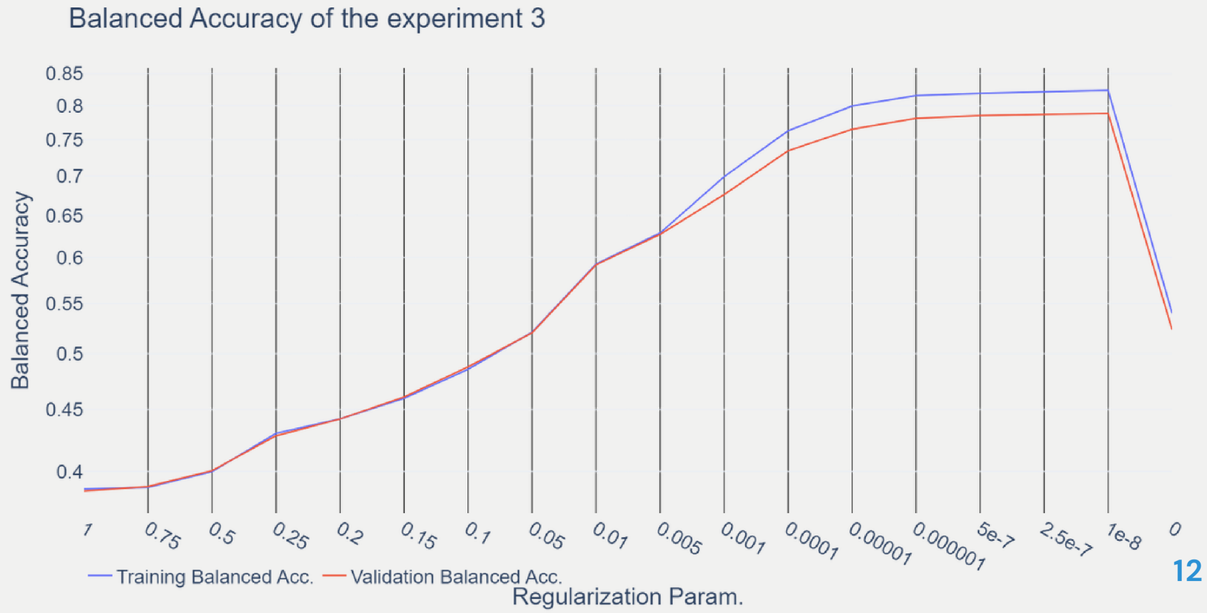
Procedure:

- **Step 1:** Automatic Forward Feature Selection Mechanism was adopted to figure out the best feature sets that fits the model best:
  - 'cln_mfcc_d_std', 'raw_melspect_mean', 'cln_melspect_std', 'raw_mfcc_d_std', 'cln_melspect_mean', 'raw_mfcc_d2_std', 'cln_mfcc_d_std', 'raw_mfcc_std', 'raw_mfcc_mean', 'cln_mfcc_mean', 'cln_mfcc_d_mean', 'zcr', 'yin', 'bandwidth_mean', 'bandwidth_std', 'flatness_mean', 'flatness_std', 'centroid_mean', 'raw_contrast_mean', 'raw_contrast_std', 'cln_contrast_mean', 'cln_contrast_std'
- **Step 2:** Using the Bisection method to select the best "Regularization parameter" on 3 different datasets, with selected features, as follows:
  - Selected Features dataset [12000 x 442]
  - Selected Features with balanced classes dataset [69796 x 442]
  - PCA-reduced Selected Features with balanced classes dataset [69796 x 124]

Conclusion:

- The Best Model performance was found on the balanced PCA=Reduced Selected features dataset, using a regularization parameter of 1e-8 with a balanced accuracy on the validation dataset of 78% which is very close to the non reduced balanced dataset.
- The model started to show overfitting behavior at a regularization parameter of 0.01 for all of the datasets, with the gap widening between the training and validation scores as we decreased the regularization parameter which is the main reason of overfitting.



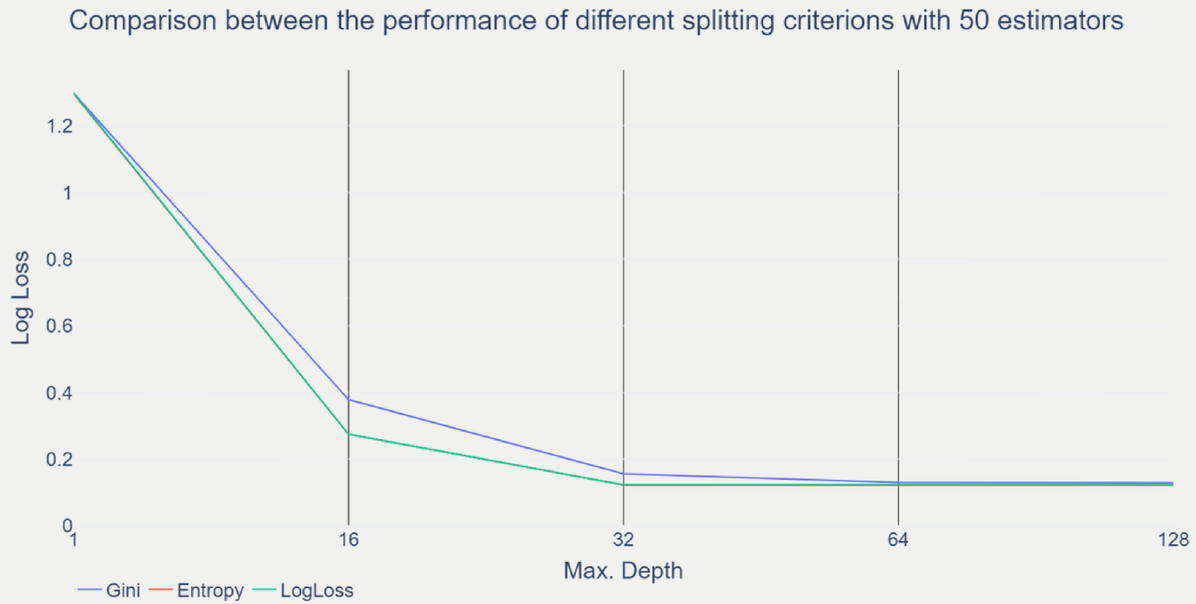


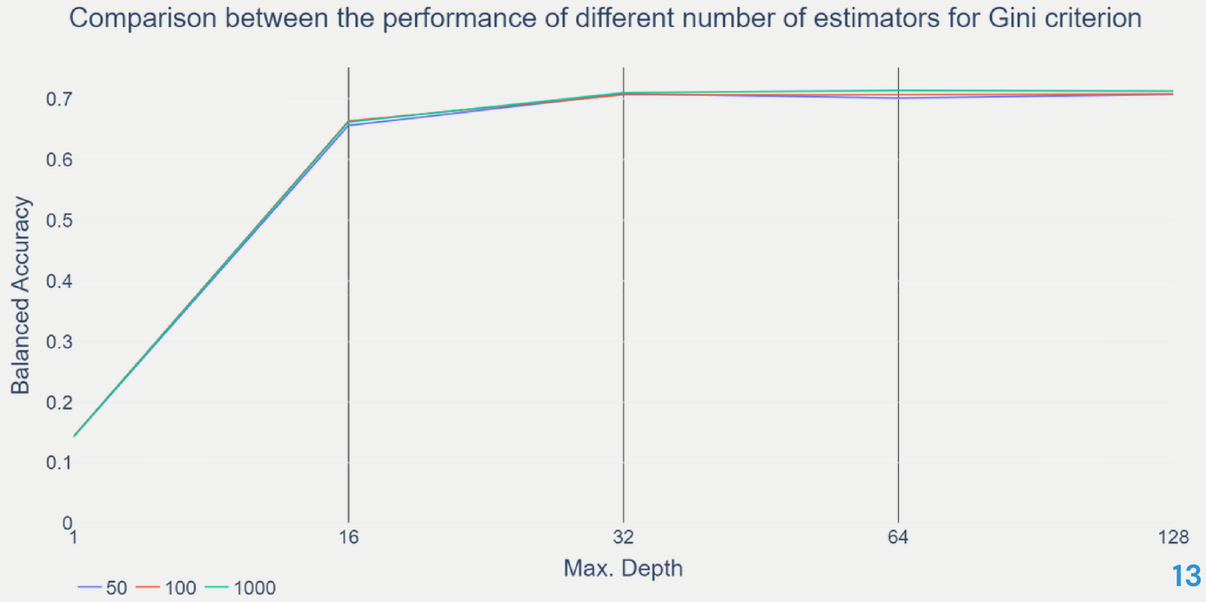
4. Experiments

4.4 Ensemble [Random Forest]

Procedure:

- **Step 1:** We experimented with the effect of choosing 3 different criterion functions "Gini/Entropy/Log Loss" that measure the quality of the splits for the random forest, tested on the whole dataset without any feature selection, to study their effect on the performance.
- **Step 2:** Choosing the Gini criterion we then experimented with the number of the estimators [1, 50, 100, 1000] to study their effect on the performance, disregarding any feature selection mechanism.
- **Step 3:** Applying the Automatic Forward Feature Selection methodology powered by the heuristics mentioned previously of eliminating the redundant features left us with the following feature sets:
  - 'raw_melspect_mean', 'raw_mfcc_mean', 'cln_melspect_mean', 'raw_contrast_mean', 'zcr', 'centroid_mean', 'energy_std', 'energy_mean', 'power_std'
- **Step 3:** We performed a bisection method algorithm to figure out the best "Maximum Depth" parameter to boost the performance to the limit, and these experiments included 3 different datasets:
  - Selected Features dataset [12000 x 170]
  - Selected Features with balanced classes dataset [69796 x 170]
  - PCA-reduced Selected Features with balanced classes dataset [69796 x 60]



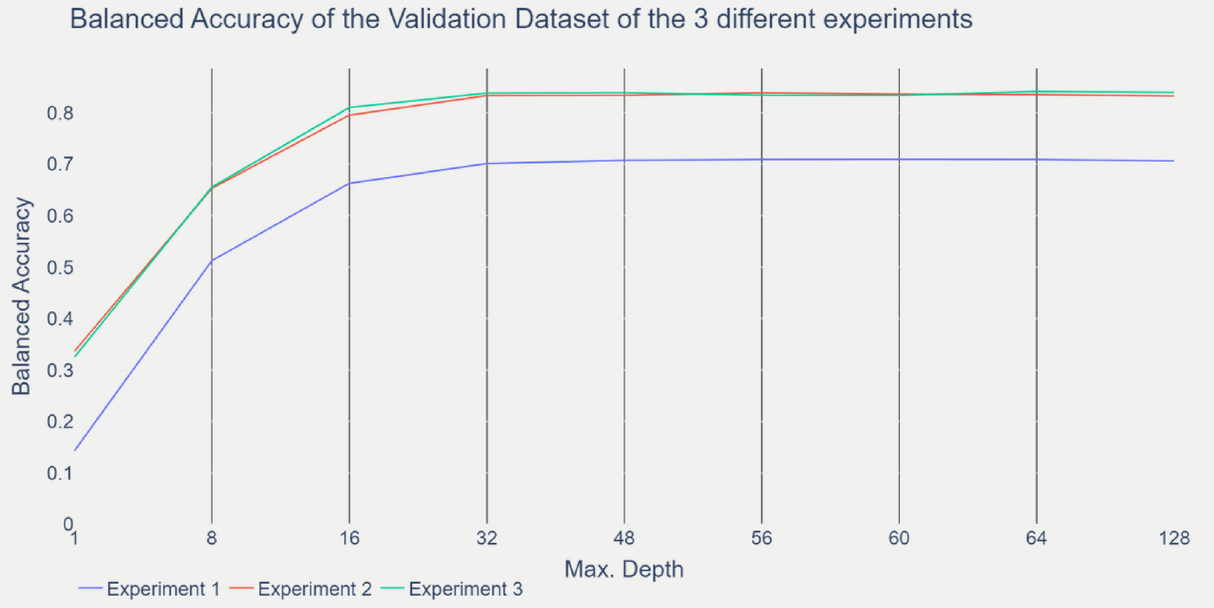


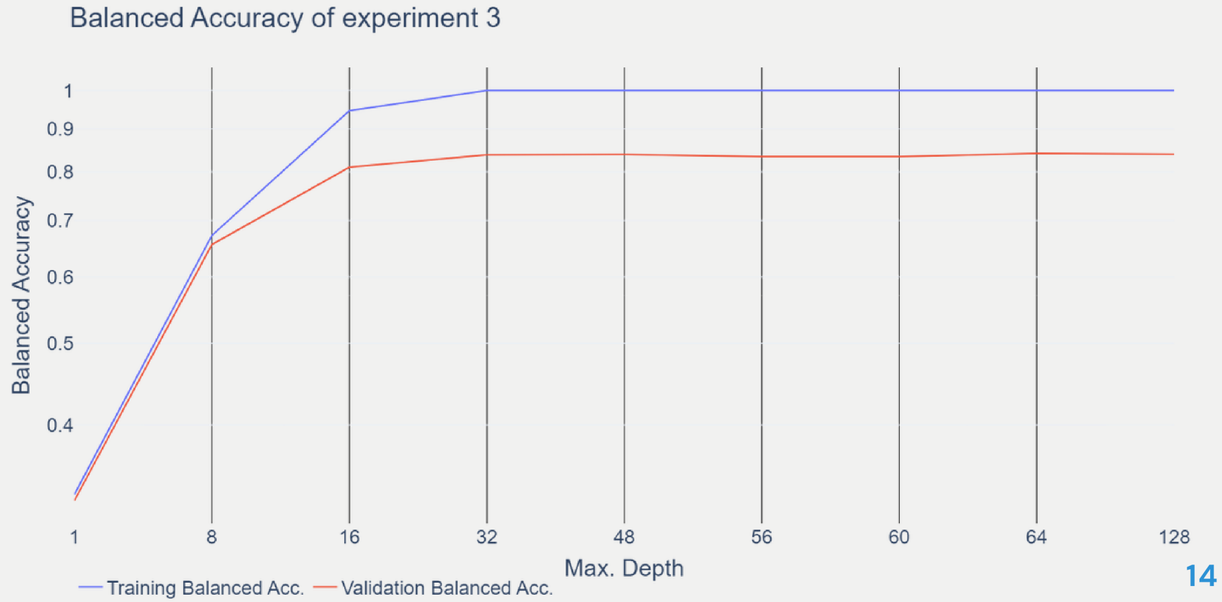
4. Experiments

4.4 Ensemble [Random Forest]

Conclusion:

- Upon careful observation of the plots, it was determined that the selection of the splitting criterion did not exert a substantial influence on the training process or the resultant outcomes of any of the models. The impact, or lack thereof, of the splitting criterion on both training and results was found to be minimal and inconsequential.
- Once the number of estimators surpassed 50, it was observed that there was no impact on the training and validation processes of the models. Increasing the number of estimators beyond this threshold did not result in any noticeable changes in the outcomes for both training and validation.
- Upon careful evaluation, it was observed that the performance exhibited by the First experiment fell short in comparison to both the second and third experiments. The difference in performance, when considering the best-case scenario, amounted to an approximate margin of around 10%.
- Despite applying PCA-reduction to the balanced dataset, performance did not improve noticeably. However, the achieved performance remained nearly identical to that of the second experiment, even with a significantly reduced number of features.
- Upon selecting the model that exhibited the optimal performance and utilized the least number of features, a visual examination of the plot revealed a conspicuous overfitting issue, amounting to an approximate 20% disparity. This overfitting predicament could potentially arise from either inadequate regularization or an insufficient minimum number of samples per leaf.
- It was also observed in the confusion matrix [graph removed for space reasons[that almost all classes were classified well, except for eowl01 which slightly suffered from misclassification.





4. Experiments

4.5.1 Neural Network [Linear Models]]

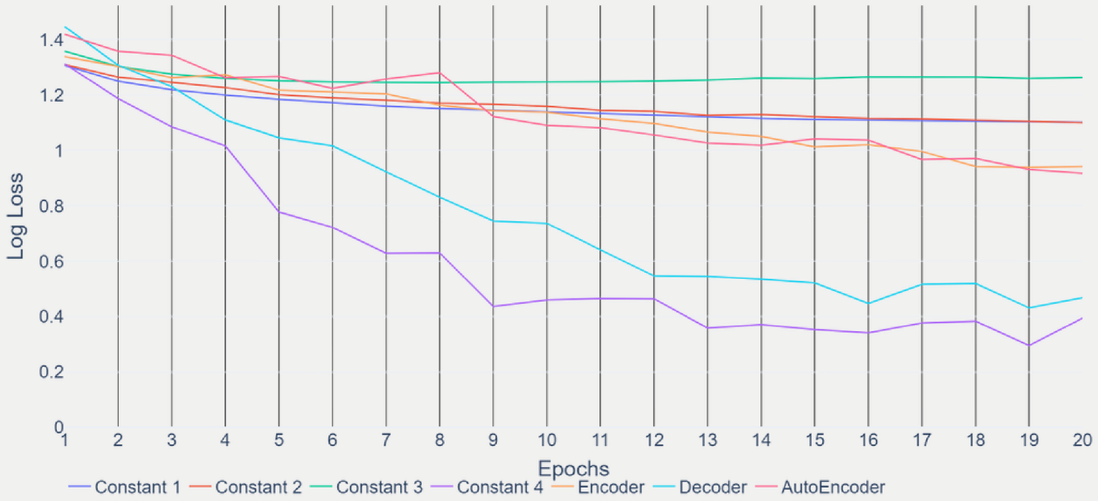
Procedure:

- Disregarding the feature selection methodology at first, we experimented with different model architectures to identify the most appropriate ones that fits best to our task:
  - Architecture 1 [Constant Layers]:
    - Step 1: A single layer for feature reduction from d features to 64 features, followed by 4 layers with equal input and output features of 64 features each. Finally, a single layer is used to reduce the output to 7 classes.
    - Step 2: Increased the number of internal features to 128 and keeping 4 layers as internal layers
    - Step 3: Same Architecture as the constant layers but increased the number of layers to 8 internal layers while keeping 64 as the number of features.
    - Step 4: Adding a Rectified linear unit "ReLU" activation between each internal layer while keeping 4 internal layers and 64 features.
- Cont. Architectures:
  - Architecture 2 [Funnel Auto Encoder]:
    - Encoder [Pre-Funnel]; gradually reduced the number of features from 548 to 64 using 5 linear layers [548, 512, 256, 128, 64] followed by the previous constant architecture: 4 layers of 64 features with "ReLU" activations and a single layer for the output
    - Decoder [Post Funnel]: Starting with the constant architecture of 1 layer from d features to 64 then 4 layers with 64 features and activations followed by a gradual decrease from 64 feature to 7 outputs
    - Auto Encoder: Using the previous encoder, constant decoder blocks we ended up with around 14 layers with inputs as follows [(548, 512, 256, 128), (64, 64, 64, 64), (32, 16, 8, 7)]
- Comparing all the architectures we found that the best performance was achieved on Architecture 1 Step 4
- Selecting The Constant Architecture [Step 4] we performed a forward feature selection methodology guided by our heuristics to reduce the number of features and a better bias-variance tradeoff, we performed an extended training on 3 different datasets: [120000x180], [69796x180], [69796x53]

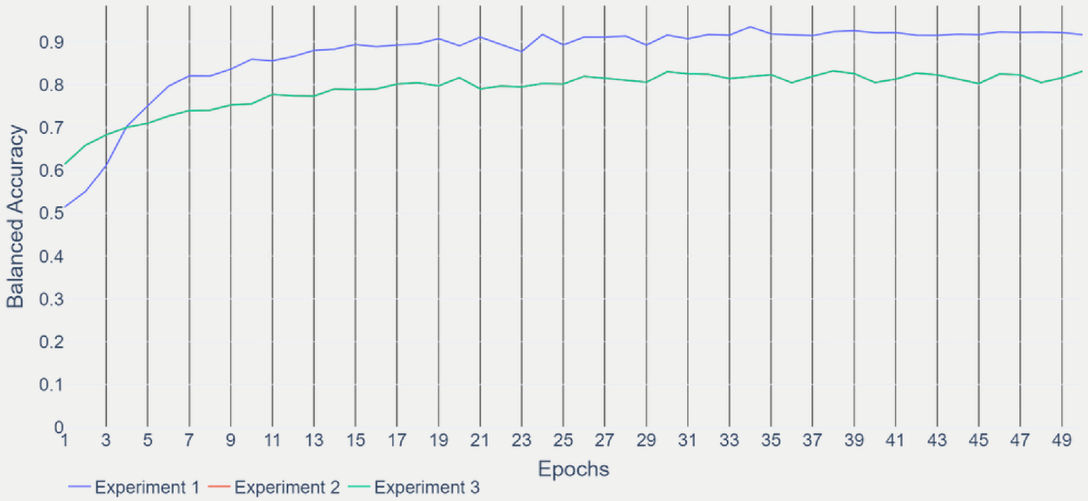
4. Experiments

4.5.1 Neural Network [Linear Models]

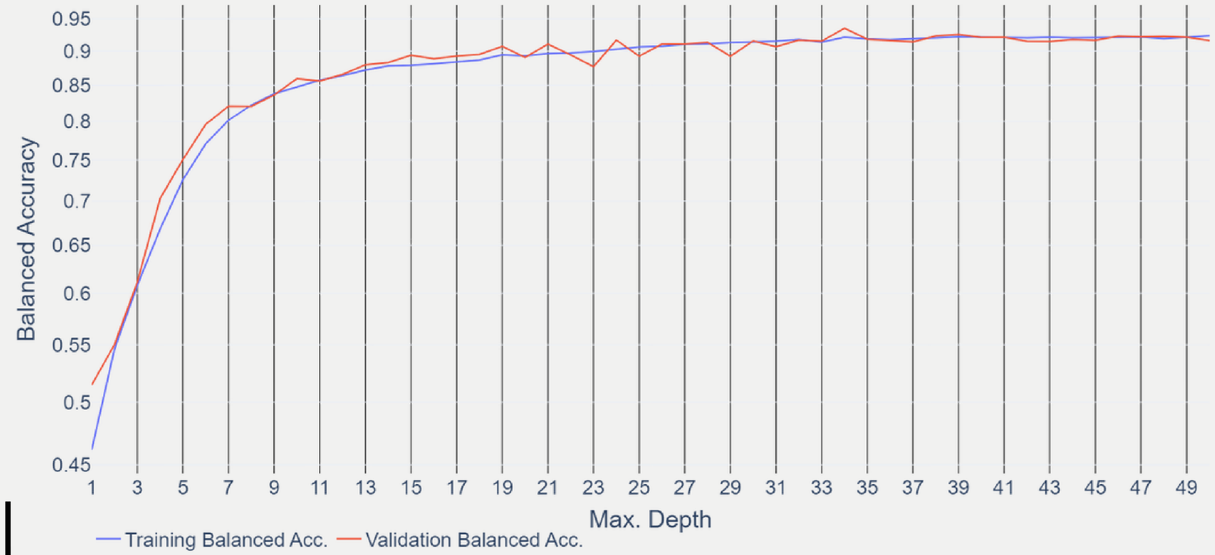
Comparison between the performance of different architectures



Balanced Accuracy of the Validation Dataset of the 3 different experiments



Balanced Accuracy of experiment 1



Conclusion:

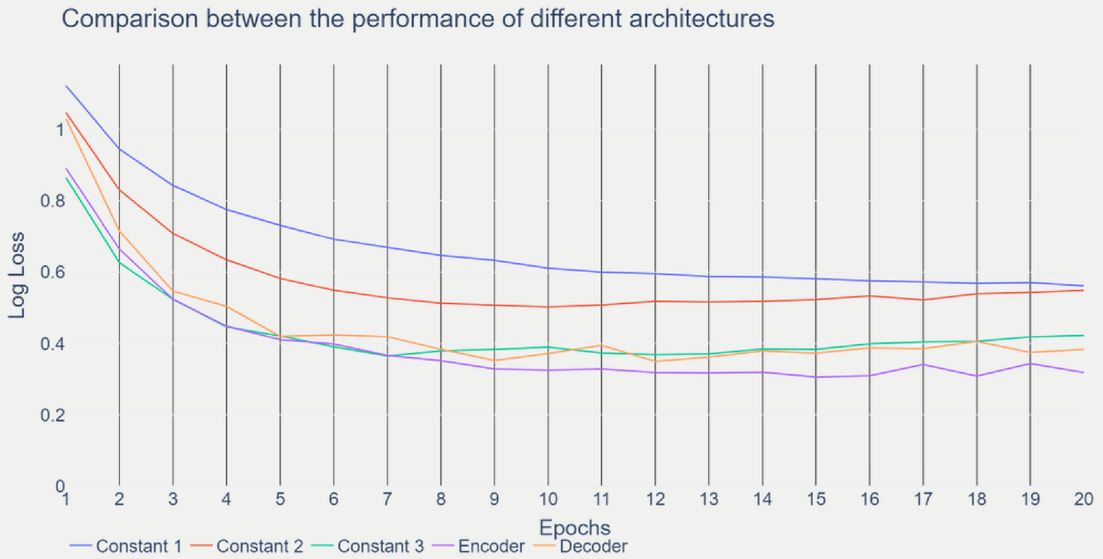
- Based on the information presented in the initial plot, it can be inferred that the Constant 4 model architecture exhibits superior performance in terms of log loss. In contrast, both the shallower and deeper architectures clearly demonstrate signs of underfitting.
- To our surprise, the first experiment exhibited superior performance in terms of balanced accuracy compared to both the second and third experiments. However, it is noteworthy that the performance of the second and third experiments remained indistinguishable from each other.
- The initial experiment exhibited highly promising outcomes, devoid of any indications of overfitting or underfitting, surpassing the best baseline we had established.
- It is worth to mention that the confusion matrix showed very healthy prediction rates among all classes.

4. Experiments

4.5.2 Neural Network [Convolutional Models]

Procedure:

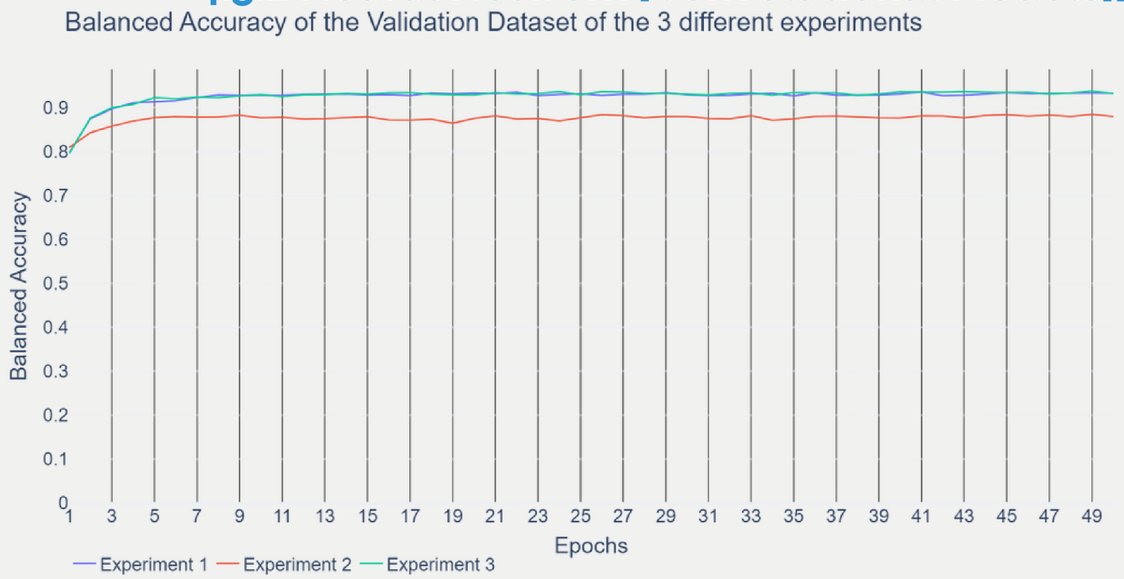
- Similar to the Linear Models, At first we started to experiment with different Architectures in order to find the most appropriate one that fits well to our data, by disregarding the feature selection methodology in the beginning,
  - Architecture 1:
    - Step 1: a Single 1-D Convolution Layer transforming the input of 1 layer x d features to 64 layers with n features then 4 1D convolution layers with 64 filters each followed by a batch normalization and a relu activation and lastly a final layer from 64 filters to a single layer with 7 outputs.
    - Step 2; Increased the number of internal layers' filters to 128 while keeping the number of layers = 4
    - Step 3: Increased the number of internal layers to 8 and the number of internal filters to 128
  - Architecture 2;
    - Step 4; Introduced the encoder mechanism by adding 3 Layers in the beginning one from [548 -> 512, 512 -> 256, 256 => 128] Followed by all the layers from step 3.
  - Architecture 3
    - Step 5: Introduced the decoder mechanism by adding 4 layers in the end preceded by the constant layers from step 3.

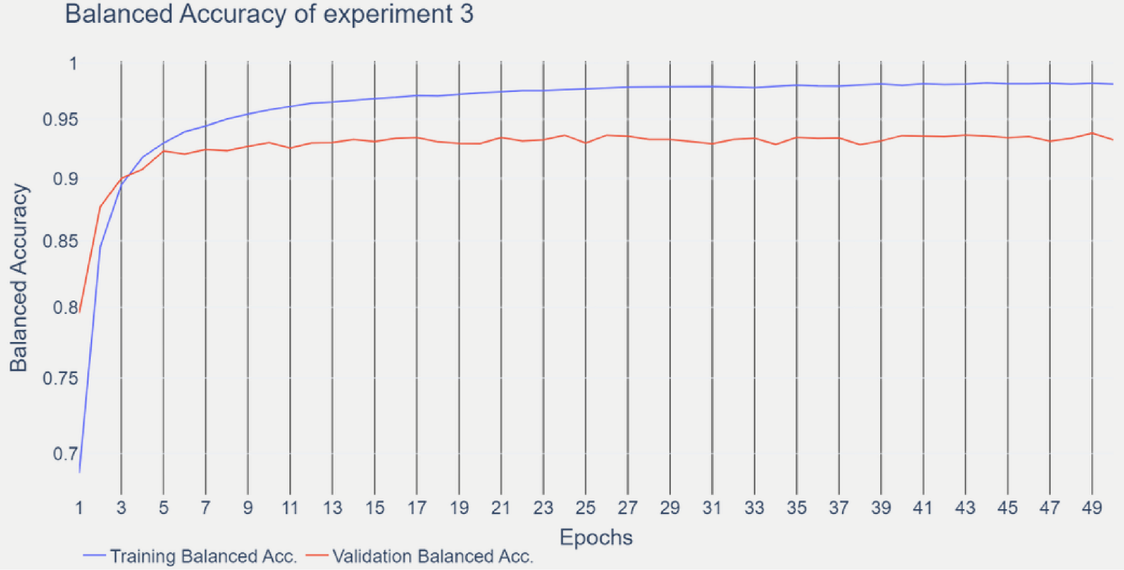


- Comparing the performance of each of the architectures we found that the best model was Architecture 1 step 3 [8 layers with 128 filters], yet it suffered from slight overfitting.
- We then applied automatic forward feature selection to further boost the performance and reduce the overfitting, and trained the model on 3 different datasets:
  - Selected Features [Dataset size: 12000 x 212]
  - Selected Features - Balanced [Dataset size: 69796 x 212]
  - PCA Reduced Selected Features [Dataset size: 120000 x 77]

4. Experiments

4.5.2 Neural Network [Convolutional Models]]





- Based on the observations, it was noted that the first and third experiments demonstrated superior overall performance when utilizing both the full selected feature sets and the PCA-reduced feature sets. However, it is worth mentioning that the attempt to balance the dataset did not yield as satisfactory results as the other experiments.
- Upon examining the plot representing the results of the third experiment, it becomes evident that the model exhibited indications of overfitting, specifically around 5% deviation in balanced accuracy. This overfitting phenomenon can be attributed to the utilization of a more intricate model than necessary to address the given problem, or alternatively, it suggests that incorporating additional datapoints could potentially be advantageous in this particular scenario.
- In summary, the obtained results can be deemed satisfactory due to the achieved balanced accuracy rates. Specifically, the validation set demonstrated a commendable accuracy of 93.5%, while the training set exhibited an impressive accuracy of 98.5%. These findings indicate that employing the 2D Convolution layers within this architecture for the purpose of classifying "Whole files" instead of individual rows proved to be a viable approach.

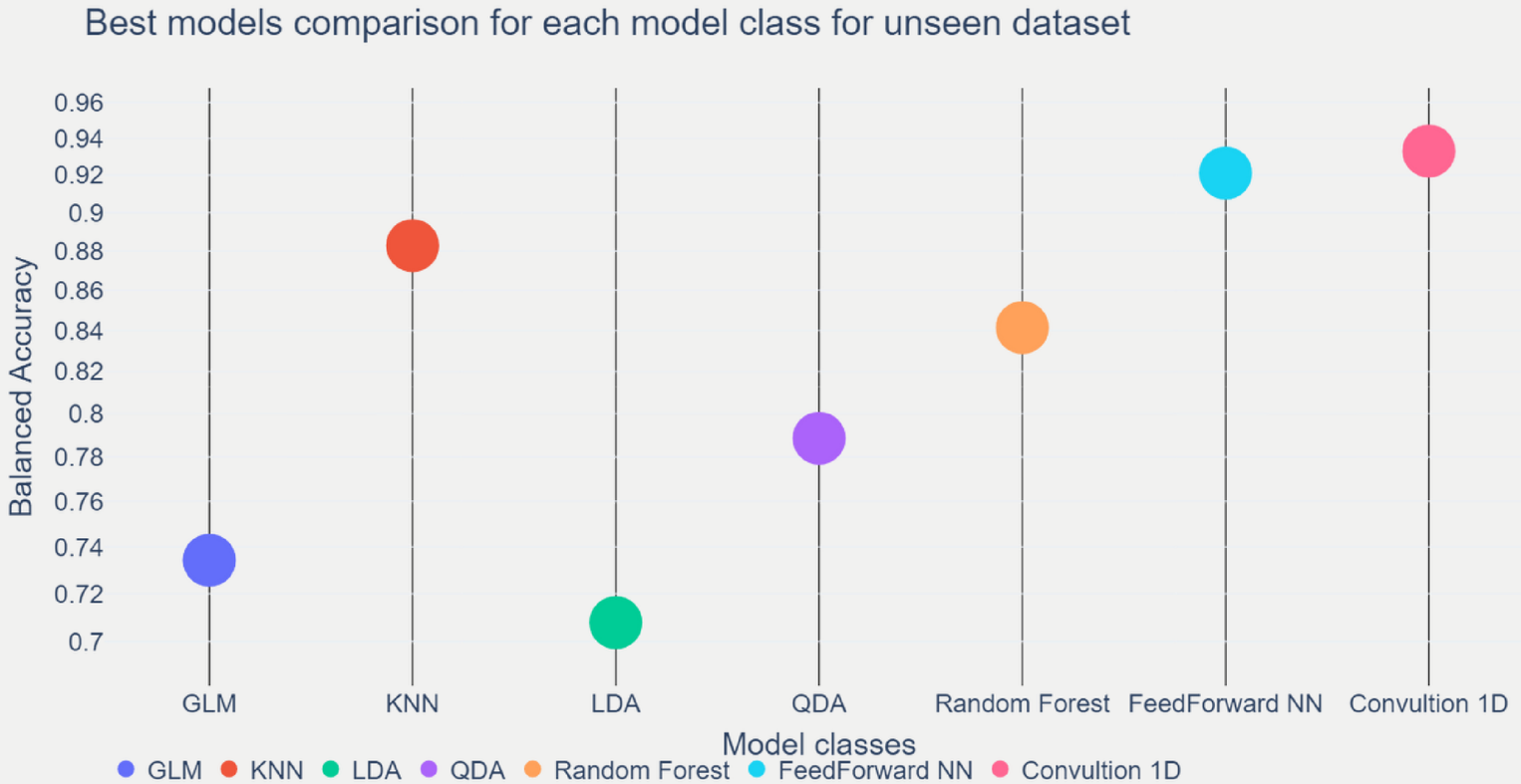
5. Conclusion

Most Common Feature sets used:

- raw_melspect_mean, cln_melspect_mean, cln_melspect_std
- raw_mfcc_mean, cln_mfcc_mean
- raw_contrast_mean, cln_contrast_mean
- centroid_mean,
- yin

Best Model-classes based on different metrics:

- 1-D Convolutional neural networks
- Linear Feed forward neural networks
- K-Nearest Neighbors
- Ensemble Methods [Random Forests]



General Observations:

- In cases of underfitting models, some of the classes specifically the "eueowl1" class suffered, slightly more then other classes, from misclassification which further confirms the under-representation of the class in the dataset as mentioned in the previous report.
- In cases of overfitting models, the misclassification percentages of all of the classes were almost the same for the unseen test-sets, which can open the door of combining different models in a voting ensemble in order to mitigate this problem, as further experiments.
- In models with balanced accuracy less than 85% we have encountered a some lenience of false positive and false negative predictions towards the 0 class even when training on a balanced dataset, but this issue completely disappears after that threshold.