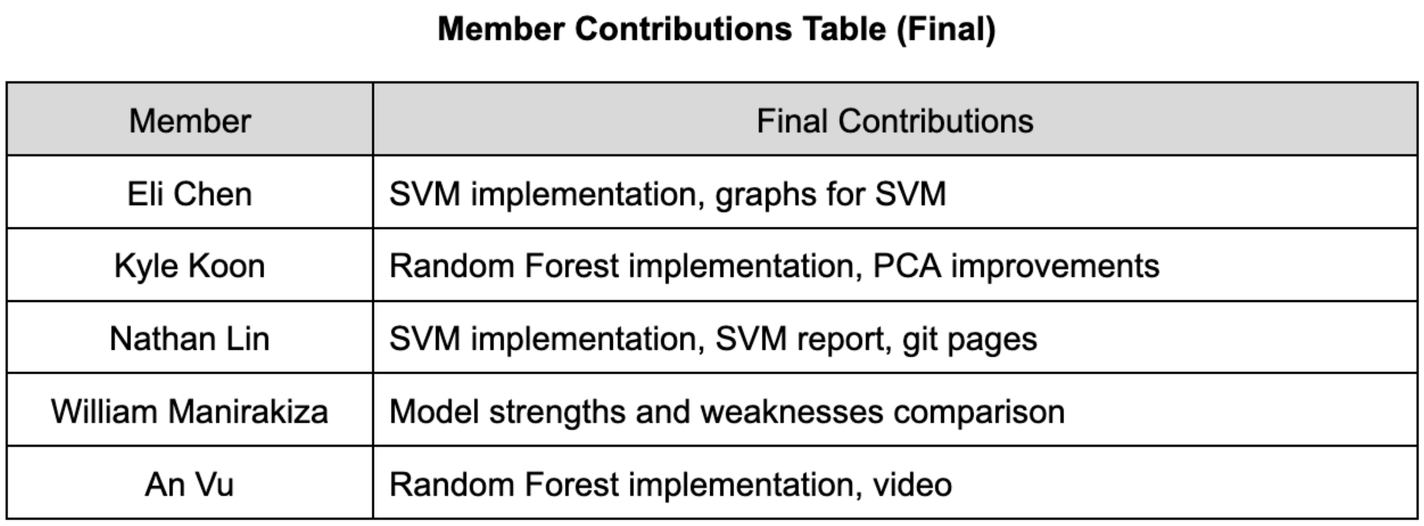
Group 56 - Youtube Virality - Final Report

View on GitHub

Contribution Table



Member Contributions Table (Final)

Member	Final Contributions
Eli Chen	SVM implementation, graphs for SVM
Kyle Koon	Random Forest implementation, PCA improvements
Nathan Lin	SVM implementation, SVM report, git pages
William Manirakiza	Model strengths and weaknesses comparison
An Vu	Random Forest implementation, video

View Gantt Chart

Introduction/Background

Content creators and marketers are constantly aiming to understand what makes a YouTube video go viral. Factors like video titles, tags, and audience interactions, such as likes and comments, are key determinants of virality. Research shows that an audience's emotional engagement, particularly preference over controversy, is key to a video's popularity, while metadata like titles and tags provides a further boost [1]. Additional major contributors to views include producer fame, cross-platform promotion, and consistent audience engagement [2]. Other studies have also demonstrated that metadata features can influence video virality, but with regional differences [3]. Cultural contexts also shape metadata's impact on video virality [4]. In this project, we will focus specifically on how metadata elements like video titles and tags correlate with views, perhaps across different regions, and we will do so by applying machine learning algorithms to Trending YouTube Video Statistics dataset.

Trending YouTube Video Statistics dataset dataset contains information on videos that appeared on YouTube's trending page some time within a range of multiple months. The dataset is separated into 10 files based on the video's country of origin and has 16 features, which include the video and channel titles, likes, views, comments, etc. By analyzing this dataset, we aim to identify patterns that help predict and optimize video virality.

Problem Definition

Going viral may be a life-changing experience for YouTube content creators and marketers, but the key to producing a video that reaches such heights in popularity is not always clear. Content quality is obviously important, but for maximal views, many more video features, such as tags and titles, play a crucial role.

Our goal is to analyze these features to determine which ones most impact view count. By classifying videos based on their performance and analyzing the relationships between variables, we can develop insights that help creators optimize their content for views and audience engagement. Our approach aims to provide practical advice to content creators concentrated on impactful view count-related factors to help them improve their visibility and engagement.

Methods

For the current iteration of our solution, we experimented with a combined United States and Canada dataset. We applied four data preprocessing methods. First, we removed non-English data entries, since we wanted to focus only on English ones for ease of analysis. Next, we removed duplicate videos, keeping only the entry with peak viewership, because entries for the same video with non-peak views are redundant and contribute no extra information to our analysis of the relationship between popularity and metadata. We also bucketed the videos into view count ranges of 0-100k, 100k-500k, 500k-1m, 1m-5m, 5m-20m, and 20m+ to more easily find relationships between videos with a similar view count. Then, we vectorized the tags feature, which we are choosing to focus on first for our analysis, by normalizing the text data (lowercasing, stemming, and removing redundant spaces, special characters, and stop words), then using a bag-of-words method. Because we are dealing with video tags and headlines, the only thing that matters is what words appear and not the order of the words.

Since the resulting bag-of-words vectorization had such massive dimensionality, we needed to apply a dimensionality reduction method to reduce overfitting once we got to training our model, and we decided on principal component analysis (PCA) and t-SNE. With PCA, we determined the number of components needed to maintain 90% variance (in z-space) to simplify our models' learning process. With t-SNE, we attempted to visualize these reduced dimensions to further explore clusters of videos with similar metadata and view count categories; however, this approach yielded little valuable insight and was quickly abandoned. Last, we created a COO (coordinate) matrix, which was highly effective at storing and efficiently handling the sparse data from the bag-of-words vectorization. With it, we were able to represent the large number of features as a sparse matrix, which significantly reduced the memory use and computational time. (The COO matrix was integrated into the training pipeline for our neural network implementation.)

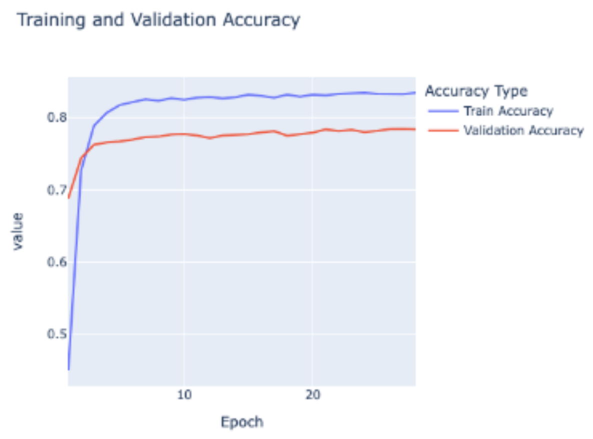
After preprocessing our data, we fed it into a few ML algorithms, the first of which was a neural network. We chose a neural network because this algorithm is very good at learning complex patterns from large datasets with high-dimensional data like ours. The neural network structure consists of four hidden layers with decreasing sizes (128 to 16 neurons), incorporating both batch and layer normalization for training stability. We also utilized the Swish activation function and included dropout layers to prevent overfitting. To address the inherent class imbalance in video view counts, we also implemented a custom Focal Loss function, which helps the model focus on "harder to classify" examples. The model was trained using mini-batch gradient descent with a cyclic learning rate scheduler. The dataset was split into training, validation, and test (70/15/15) sets with features standardized before training. This architecture balances between model complexity and regularization, which makes it good for learning patterns in our high-dimensional video metadata while keepings its generalization capability.

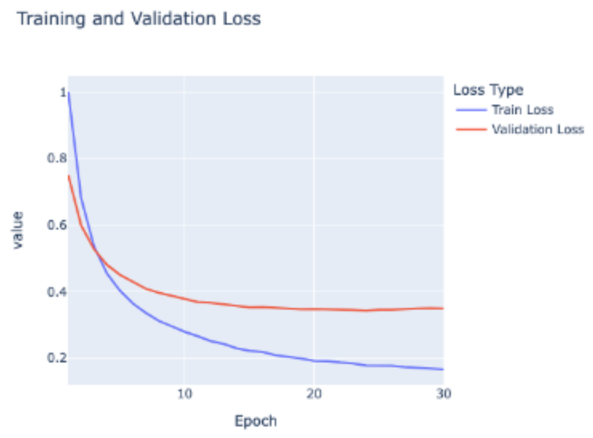
We then implemented a Support Vector Machine (SVM) to classify videos based on their metadata and view count categories. We chose SVM because it is good at handling non-linearly-separable data and high-dimensional datasets, both of which describe our chosen dataset. The pipeline included feature normalization using StandardScaler to standardize feature magnitudes and SelectKBest to select the top 20 most informative features, reducing dimensionality and computational complexity. To address class imbalances, we applied SMOTE-Tomek, combining synthetic oversampling of minority classes with undersampling of majority classes. An RBF kernel was chosen for the SVM to capture non-linear relationships, and hyperparameters, including C, γ, and the number of selected features, were optimized using grid search with stratified cross-validation. The dataset was split into training, validation, and test sets (70/15/15) to evaluate model performance, with accuracy, precision, recall, F1-score, and confusion matrices generated to assess the results.

Last, we implemented a Random Forest Classifier to perform the same task. We chose random forest because this method's bootstrap aggregation helps lower the risk of overfitting while generally maintaining the accuracy of a basic decision tree algorithm. Before training the model, we first normalized our preprocessed data using StandardScaler, then split into testing and training datasets (80/20), then applied SMOTE-Tomek sampling to achieve a better class balance (like we did for SVM). Our random forest currently uses default values for all hyperparameters, except for max_depth, which is set to 12 through manual selection. Like for SVM, we also generated plots showing the random forest's accuracy, precision, recall, and F-1 score as well as a confusion matrix.

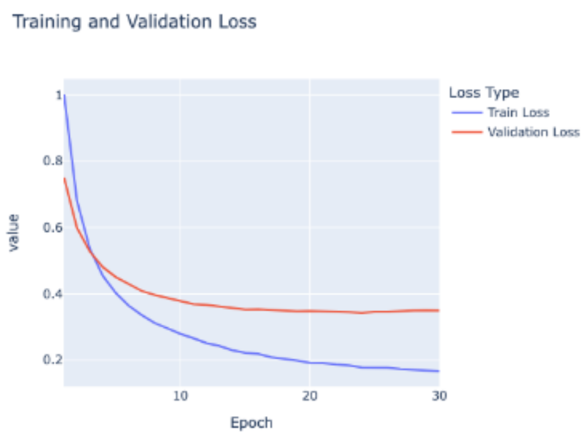
Results and Discussion

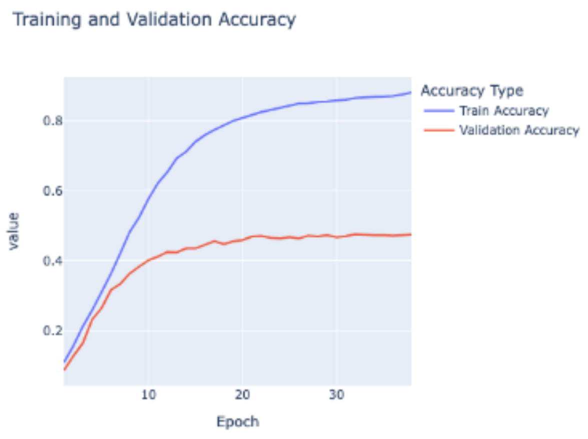
After experimenting with our various chosen methods, we were able to achieve the following results. First, we analyze our neural network:





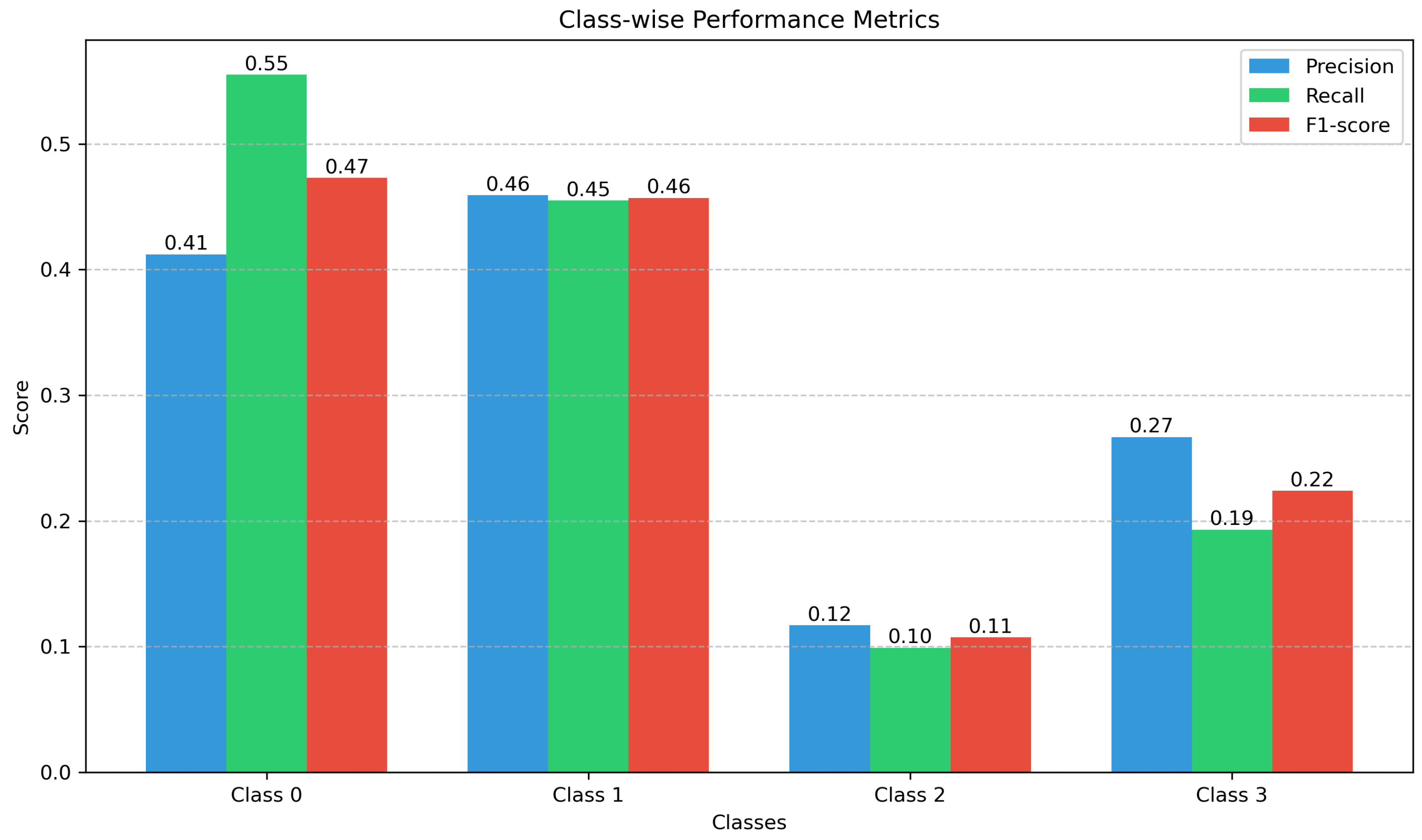
The above plots show the training and validation accuracy and loss over time for our first optimized neural network (not using sparse tensors). We were able to achieve a final test accuracy of around 80%. This is above our expected accuracy of 70% but below our desired accuracy of 90%, which is decent. Some other metrics we calculated were a precision of 0.7972, recall of 0.7916, F1-score of 0.7911, which are all relatively okay. However, we soon realized that we had been "cheating" in our training: our video dataset contains many duplicate entries (same videos but over different days and with different view counts), and our testing data was most likely contaminated with the same videos that were in our training set. As a result, we had to remove the duplicate entries and retrain our model. We also converted to using sparse tensors. With our new neural network, we produced the following accuracy and loss visualizations:

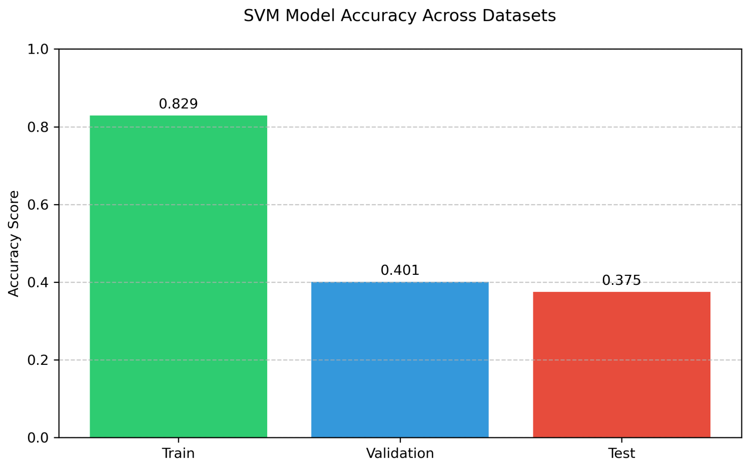


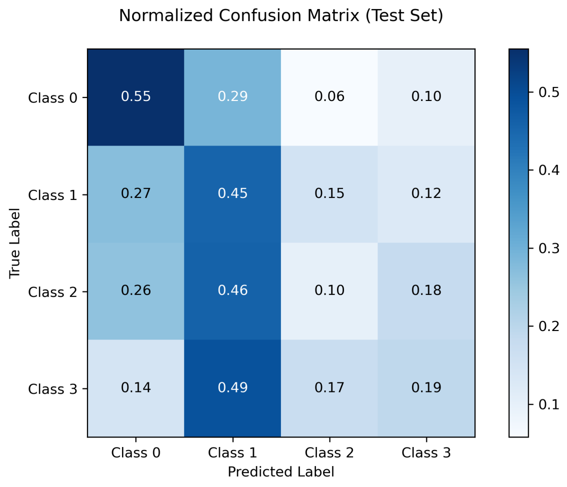


After retraining using non-duplicate data, adding data from the Canada videos, and using sparse tensors, our model achieved a final accuracy of around 50%. This new accuracy is much lower than our expected accuracy, so we have a bit of work to do. However, we also see that our training accuracy is extremely high. This is a clear indication of overfitting. We had already been concerned about overfitting due to having too many features, so we began to integrate principal component analysis (PCA) into our data preprocessing. Unfortunately, even after using a PCA-reduced dataset, we saw little change in our test results. Besides accuracy, our new neural network produced a precision of 0.4499, recall of 0.4555, and F1-Score of 0.4466. These are all far below our desired metrics, so again, we have something to work on.

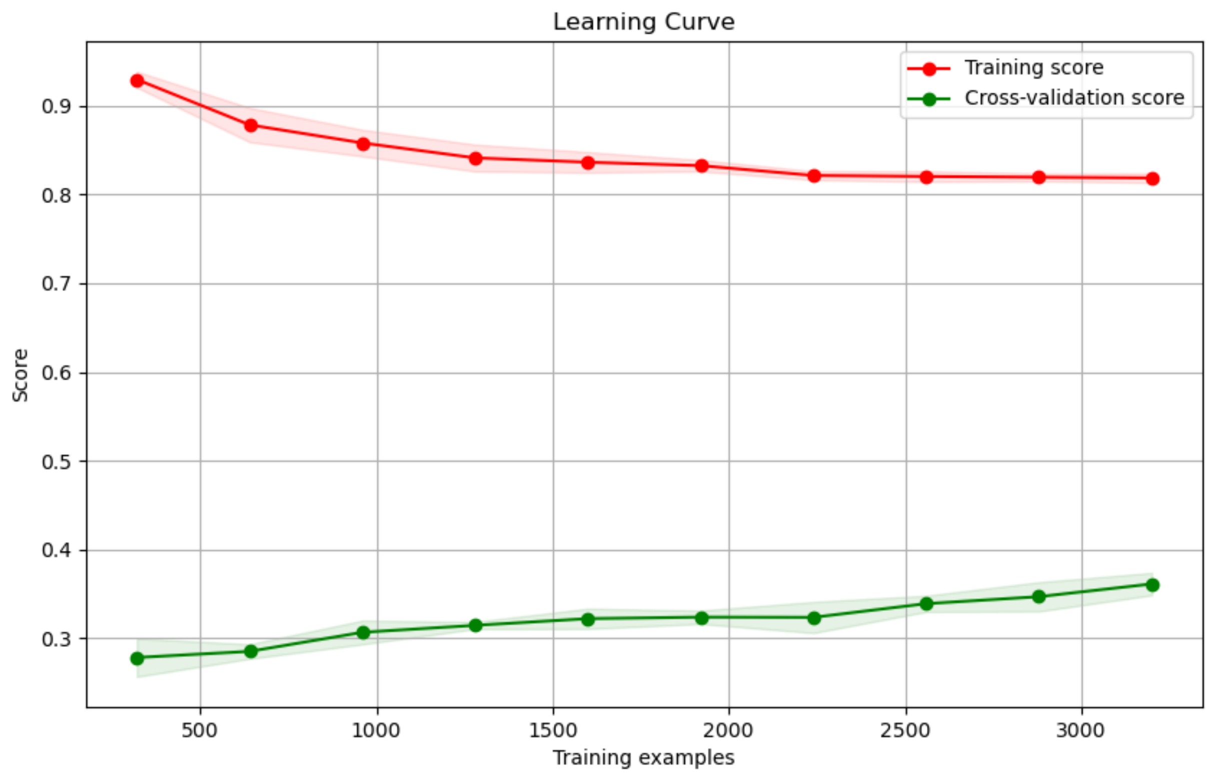
Meanwhile, the SVM achieved a training accuracy of 82.88%, but validation and test accuracies dropped significantly to 40.08% and 37.47%, indicating severe overfitting and poor generalization. The model struggled to handle the dataset's high dimensionality and class imbalances, with minority classes showing particularly poor precision, recall, and F1-scores. The confusion matrix highlighted frequent misclassifications, especially for less frequent categories. To improve performance, better dimensionality reduction techniques, in addition to PCA, could help reduce noise and overfitting. Alternative models like Random Forests or Gradient Boosting may also handle imbalanced data more effectively. Further hyperparameter tuning, advanced resampling techniques, and improved feature engineering are essential to make the model more robust and reliable for unseen data.





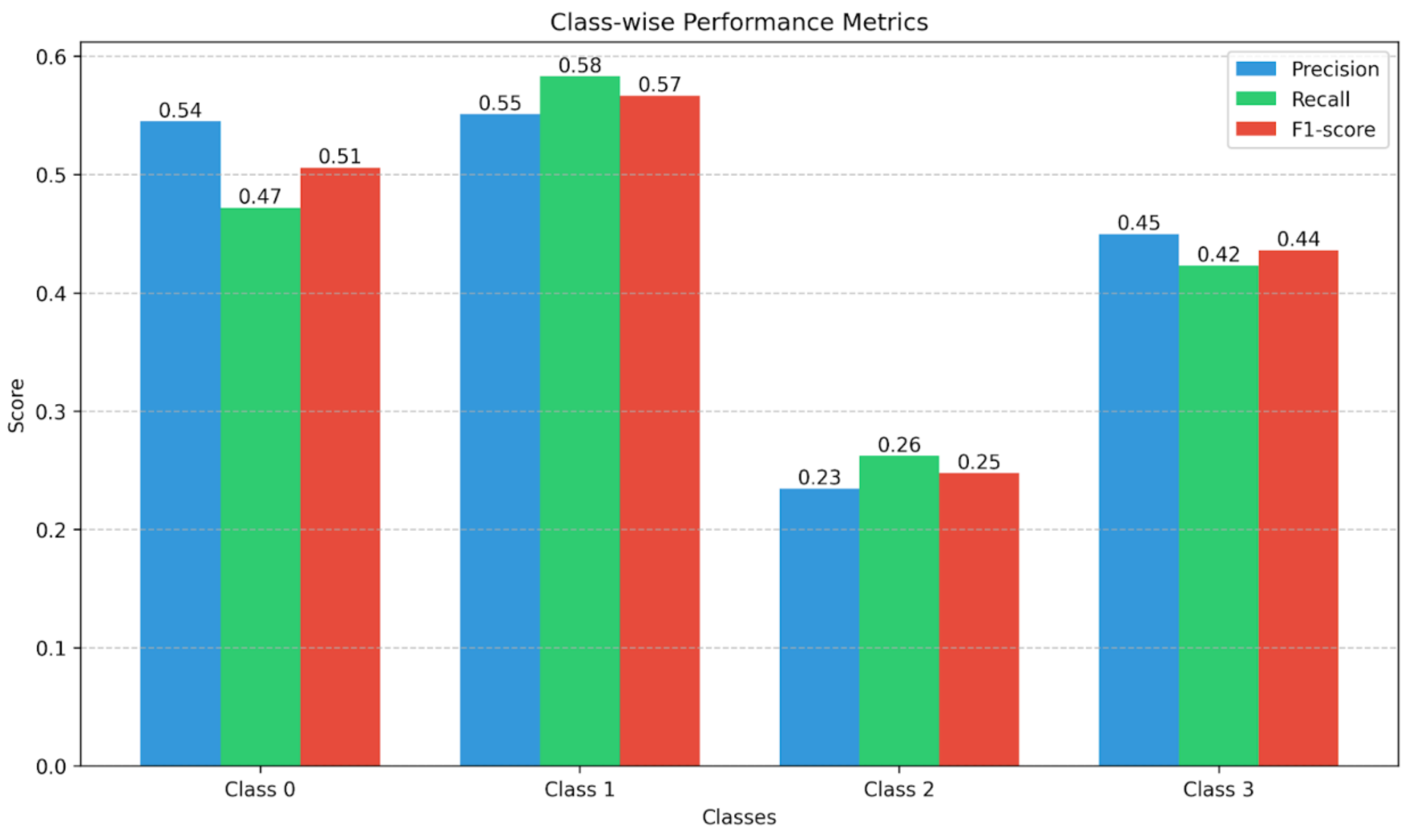


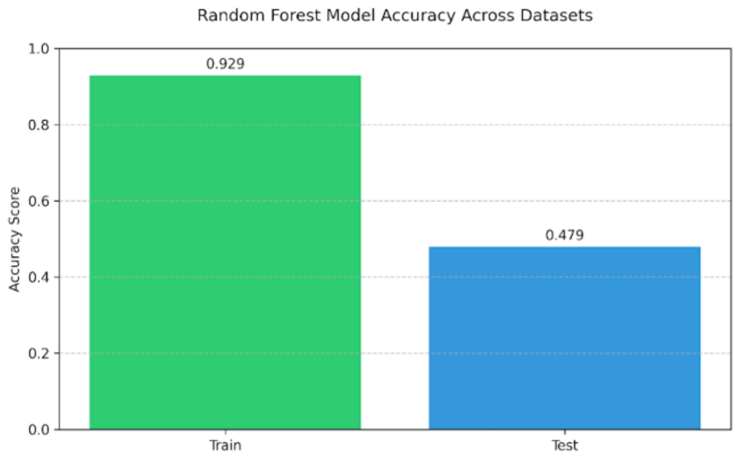
Left: Accuracy of the SVM model across the training, validation, and test datasets, demonstrating overfitting with significantly higher training accuracy compared to validation and test accuracies. Right: Normalized confusion matrix for the SVM model, showing frequent misclassification between similar categories.

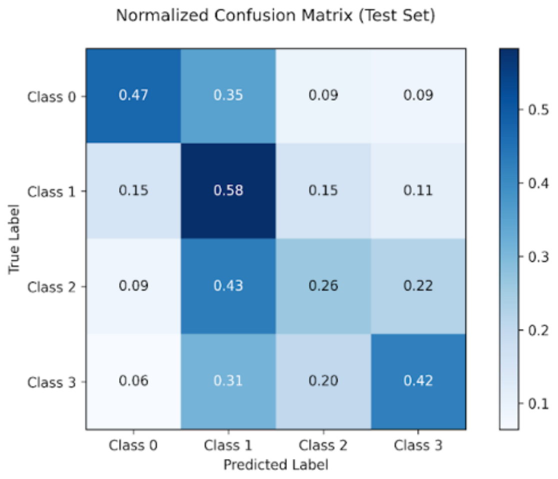


Learning curve for the SVM model, illustrating the relationship between training dataset size and model performance. The gap between training and cross-validation scores indicates overfitting and poor generalization, especially for larger datasets.

Our random forest classifier performed similarly to the neural network and SVM, this similarity being a high training accuracy of 92.90% and a much lower testing accuracy of 47.92%. The visualizations for random forest mirror the issues seen with SVM, with poor performance in classes with less data, like classes 2 and 3. However, it is worth noting that our random forest classifier outperforms SVM on accuracy and precision, recall, and F1-score for almost every class. This is likely due to the fact that the random forest algorithm trains several decision trees on a subset of the dataset and its features, then combines the results of every decision tree generated to arrive at the final conclusion. By using an ensemble method, random forests are, by nature, more robust to noise and generally less likely to overfit.







Now is a good point to summarize the strengths and weaknesses of each of our chosen ML models. Neural networks are very effective at uncovering patterns in large datasets, but tend to overfit on small, high dimensional datasets. These strengths and weaknesses explain why our neural network did much better in training than testing since there were about 20,000 unique data points and thousands of features (even after applying PCA). SVM is very effective for high dimensional datasets, but if the number of data points is much lower than the number of features (like in our case), SVM might pick too many support vectors (causing overfitting). Similar to neural networks, this could explain why SVM had high training accuracy and poor cross validation scores. Finally, random forests have the advantage of ensemble learning, incorporating the 'vote' of multiple models. However, the individual models have the same drawback as a decision tree: trees with large depth tend to lead to overfitting. These strengths and weaknesses explain why random forest outperformed SVM in testing but achieved results similar to the neural network.

Overall, our three models performed rather consistently, though not necessarily in a desirable manner. All three presented classic symptoms of overfitting, with high training accuracy (>=80%) and low validation and test accuracy (around half the training accuracy), indicating that they were struggling with our chosen dataset, which, unfortunately, had very high dimensionality but comparatively limited entry count. Additionally, all three models struggled to classify certain classes; this is a problem with unevenly distributed data, which we could fix by reassigning class boundaries.

In summary, our next steps would be to work on reducing overfitting for all three models, perhaps by implementing additional data preprocessing methods to further reduce dimensionality, increasing the size of our dataset (in terms of data entries) with data augmentation to improve generalizability, and further tuning hyperparameters to achieve optimal model performance.

References

[1] Y. Hu, “Preference or Controversy: What Predicts Virality Most?” Wenzhou Kean University, 2019.

[2] G. Feroz Khan and S. Vong, “Virality over YouTube: an empirical analysis,” Internet Research, vol. 24 no. 5, pp. 629-647, 2014. [Online Serial] Available: https://doi.org/10.1108/IntR-05-2013-0085 [Accessed October 1, 2024].

[3] Z. Halim, S. Hussain, and R. H. Ali, “Identifying content unaware features influencing popularity of videos on YouTube: A study based on seven regions,” Expert Systems with Applications, vol. 206, 2022. [Online Serial] Available: https://doi.org/10.1016/j.eswa.2022.117836 [Accessed October 1, 2024].

[4] A. Abdihakim, “Characterizing Feature Influence and Predicting Video Popularity on YouTube,” KTH Royal Institute of Technology, 2021. [Online Serial] Available: https://kth.diva-portal.org/smash/get/diva2:1603397/FULLTEXT01.pdf [Accessed October 3, 2024]