CSE 347
Group Project 2 Report

Nicolas Kozachuk, Carl Saba, and Theodore Woodworth

Lehigh University

28 April 2023

For this project, the three algorithms that were chosen to be implemented and analyzed were logistical regression (LR), random forest, and convolutional neural network (CNN). These different algorithms were chosen since each of them have their own unique properties and perform best for different types of datasets. Each of these algorithms have their own strengths and weaknesses and they were applied to three unique datasets to provide a fair comparison between the different algorithms. The datasets selected were the Iyer, Cho, and YaleB datasets. The Iyer and Cho datasets were collected from UCI machine learning repository and are gene sequence datasets. Although the Iyer and Cho datasets have the same type of data, they have different number of features and classes, as the Iyer dataset has 12 features and 11 classes, while the Cho dataset has 16 features and 5 classes. The YaleB dataset contains the data for grayscale images of human faces and has 32x32 features with 2186 rows of training data and 228 rows of testing data, with 38 different classes in total.

The algorithms and models created for this project were coded in R. Many different libraries were needed for this project to preprocess the data, implement the algorithms, and analyze the data, such as pROC, caret, randomForest, MLmetrics, nnet, keras, tensorflow, and dplyr. The first algorithm implemented was the logistic regression algorithm and was implemented using the nnet() function in R. The second algorithm implemented was the random forest algorithm using the randomForest() function in R. The last algorithm implemented was the CNN using the keras library and the keras_model_sequential() function in R. A 2D CNN was implemented for the YaleB dataset and a 1D CNN was implemented for the Iyer and Cho dataset. A 2D CNN had to be used on the YaleB dataset because it was processing 32x32 images, whereas 1D CNN was used for Iyer anc Cho datasets because they are one dimensional. When implementing the logistic regression, random forest, and CNN for the Iyer and Cho datasets cross-validation was performed to do the hyperparameter tuning in order to achieve a better performance. K-fold cross-validation was performed to accomplish this on the datasets with a K=3. Additionally, when implementing the logistic regression and random forest algorithm for the YaleB dataset, a principle component analysis(PCA) was performed to reduce the number of dimensions. The YaleB dataset provided a training and testing dataset, but the Iyer and Cho datasets did not, so the data was split into training and testing datasets.

	Logistical Regression	Random Forest	CNN
Accuracy	0.799	0.791	0.411
F1	0.794	0.790	0.151
AUC	0.882	0.873	0.770

Figure 1: Logistic Regression, Random Forest, and 1D CNN average accuracy, F1, and AUC results for 3 runs of Iyer dataset.

	Logistical Regression	Random Forest	CNN
Accuracy	0.121	0.009	0.055
F1	0.105	0.011	0.027
AUC	0.011	0.012	0.022

Figure 2: Logistic Regression, Random Forest, and 1D CNN standard deviation of accuracy, F1, and AUC results for 3 runs of Iyer dataset.

One can see the average accuracy measure results of the Iyer dataset for 3 runs in Figure 1 for the logistic regression, random forest, and 1D CNN algorithms. The logistic regression algorithm produced accurate results and performed the best on the Iyer dataset and had the highest values for each of the accuracy measures of accuracy, F1, and AUC, as seen in Figure 1. The random forest algorithm also produced accurate results but was only slightly worse than the logistic regression algorithm in terms of accuracy, F1, and AUC. The 1D CNN algorithm did not have very accurate results and performed the worst out of the three algorithms, with a very low F1 and accuracy value, and a decent AUC, but significantly less than that of the logistical regression and random forest algorithm. These results are as expected as the Iyer data set has 12 features of gene sequence data, which is the type of data and has the number of features that would work well in a logistical regression and random forest algorithm, but not a 1D CNN algorithm.

	Logistical Regression	Random Forest	CNN
Accuracy	0.581	0.719	0.469
F1	0.580	0.706	0.311
AUC	0.795	0.879	0.763

Figure 3: Logistic Regression, Random Forest, and 1D CNN average accuracy, F1, and AUC results for 3 runs of Cho dataset.

	Logistical Regression	Random Forest	CNN
Accuracy	0.068	0.042	0.052
F1	0.067	0.040	0.033
AUC	0.019	0.011	0.031

Figure 4: Logistic Regression, Random Forest, and 1D CNN standard deviation of accuracy, F1, and AUC results for 3 runs of Cho dataset.

One can see the accuracy measure results of the Cho dataset in Figure 3 for the logistic regression, random forest, and 1D CNN algorithms. The random forest algorithm produced accurate results and performed the best on the Cho dataset and had the highest values for each of the accuracy measures of accuracy, F1, and AUC, as seen in Figure 3. The logistic regression algorithm also produced accurate results but was only slightly worse than the logistic regression algorithm in terms of accuracy, F1, and AUC. The 1D CNN algorithm did not have very accurate results and performed the worst out of the three algorithms, with a very low F1 and accuracy value, and a decent AUC, but less than that of the logistical regression and random forest algorithm. These results are as expected as the Cho data set has 15 features of gene sequence data, which is the type of data and has the number of features that would work well in a logistical regression and random forest algorithm, but not a 1D CNN algorithm.

	Logistical Regression	Random Forest	CNN
Accuracy	0.031	0.768	0.855
F1	0.026	0.770	0.792
AUC	0.651	0.909	0.953

Figure 5: Logistic Regression, Random Forest, and 2D CNN accuracy, F1, and AUC results for Yale set 1 dataset.

	Logistical Regression	Random Forest	CNN
Accuracy	0.031	0.846	0.895
F1	0.034	0.842	0.878
AUC	0.740	0.908	0.958

Figure 6: Logistic Regression, Random Forest, and 2D CNN accuracy, F1, and AUC results for Yale set 2 dataset.

	Logistical Regression	Random Forest	CNN
Accuracy	0.040	0.781	0.556
F1	0.032	0.773	0.457
AUC	0.691	0.860	0.878

Figure 7: Logistic Regression, Random Forest, and 2D CNN accuracy, F1, and AUC results for Yale set 3 dataset.

One can see the accuracy measure results of the three YaleB datasets in Figures 5, 6, and 7 for the logistic regression, random forest, and 2D CNN algorithms. The worst algorithm for the YaleB data is clearly the logistical regression algorithm. For the three YaleB datasets, the AUC for logistical regression was fairly high, but not nearly as high as the random forest and 2D CNN algorithms. Additionally, the accuracy and F1 measure for the logistic regression algorithm were extremely low for all three of the YaleB datasets. The 2D CNN algorithm performed the best for YaleB datasets 1 and 2, but the random forest algorithm performed the best for YaleB dataset 3. As can be seen in Figure 5 and 6, the 2D CNN has the highest accuracy, F1, and AUC, although the random forest algorithm is very close. As can be seen in Figure 7, the random forest algorithm has the highest accuracy and F1, and has nearly the same AUC as the 2D CNN algorithm.

The random forest algorithm produced accurate results and performed the best on the Cho dataset and had the highest values for each of the accuracy measures of accuracy, F1, and AUC, as seen in Figure 3. The logistic regression algorithm also produced accurate results but was only slightly worse than the logistic regression algorithm in terms of accuracy, F1, and AUC. The CNN algorithm did not have very accurate results and performed the worst out of the three algorithms, with a very low F1 and accuracy value, and a decent AUC, but less than that of the logistical regression and random forest algorithm. These results are as expected as the Cho data set has 15 features of gene sequence data, which is the type of data and has the number of features that would work well in a logistical regression and random forest algorithm, but not a CNN algorithm.

Each of the algorithms implemented have their own set of pros and cons. The pros of the logistic regression algorithm are that it is one of the simplest machine learning algorithms to implement, it gives inference to feature importance due to predicted parameter weights, it is fast to train and test, can handle binary or multi-classification problems, it is less prone to overfitting with low dimensional data and it has easily interpretable results. The cons of logistic regression are that it tends to overfit high-dimensional data causing inaccurate results on test sets, nonlinear problems cannot be solved since it has a linear decision surface, it is sensitive to outliers, and it requires a

large dataset. Overall, logistic regression is an efficient and simple algorithm and is commonly used as a benchmark model to measure performance, but has difficulty capturing complex relationships. These pros and cons can be seen in the accuracy results shown in Figures 1, 3, 5, 6, and 7. The logistic regression algorithm performed well for Iyer and Cho data as they are not high-dimensional data, but it performed horribly for the YaleB datasets. This is because the dataset has high-dimensional complex data with classes that are not ordered, and since logistic regression does not perform well on nonlinear data, the results are poor. For logistic regression, there are not many parameters that can be changed to help fit the data, so hyperparameter tuning was performed to achieve a better performance.

The pros of the random forest algorithm are that it can handle a large number of features and capture linear as well as non-linear relationships, it can handle missing data and outliers well, it can be used for classification and regression, and it provides a measure of feature importance. The cons of the random forest algorithm are that it can be computationally intensive for large data sets, it may suffer from overfitting if the number of trees is too high, and it is difficult to interpret the results. Overall, the random forest algorithm is a powerful algorithm that can handle large and complex datasets and produce accurate results for linear or non-linear relationships, but it is slow to train and may overfit. These pros and cons can be seen in the accuracy results shown in Figures 1, 3, 5, 6, and 7. The random forest algorithm performs well on all of the datasets, as it is a very versatile algorithm and is the second-best algorithm for each of the datasets. Hyperparameter tuning was performed to achieve a better performance and different numbers of trees were tested to achieve the best performance. After testing, it was found that the number of trees to get the best results for the Iyer and Cho dataset was 500 and the YaleB dataset was 5.

The pros of the CNN are that it can handle high-dimensional data, such as images and text, it can capture nonlinear relationships between the features and the output, it can automatically learn features from the data, and it is very powerful in performing classification tasks. The cons of the CNN are that it requires a large amount of training and computing power, its results are difficult to interpret, it may suffer from overfitting if the model is too complex or the number of epochs is too high, and it requires careful tuning of the hyperparameters. Overall, the CNN is an extremely powerful algorithm that performs well with high dimensional data, but it is computationally intensive and requires accurate parameter specification to produce accurate results. These pros and cons can be seen in the accuracy results shown in Figures 1, 3, 5, 6, and 7. The CNN did not perform well for Iyer and Cho since the datasets are not the amount and type of data that work best for CNN. It can be seen in the Figures that the CNN performed significantly better on the YaleB dataset as this dataset is the grey scale human face dataset, which comes from an image. The parameters tuned for the CNN were the batch size and number of epochs in order to achieve the best performance. It was found for all the datasets that the batch size and epoch for the Iyer, Cho, and YaleB dataset should be 32 and 10 to get the best results.