Final Report

Team 2- Goal Diggers
Jordan Dube, Luke Unterman, Blue Arevalo, Kiersten Kofi Adomfrimpong
435- Data Science
12-01-2022

Process

For our final model, we generated five different sets of features from the initial list of input sequences and merged them into one CSV. These sets were all generated by the Pfeature web server and calculated the Amino Acid Composition, Standard Physico-chemical Properties, Advanced Physico-chemical Properties, Repetitive Residue Information, and Distance Distribution of Residues of each of the amino acid sequences. These sets were merged and designated classes in Pandas using Python. We tested multiple classification algorithms, including Decision Trees and kNN, although we determined through trial and error that generally using Decision Trees would provide us with the best results (meaning higher average accuracy/avgMCCs). The parameters we selected for the Decision Tree operator are listed as follows:

Criterion: information_gain

Maximal Depth: 10

Apply Pruning: True

Confidence: 0.1

Apply Pre-Pruning: True

Minimal Gain: 0.01

Minimal Leaf Size: 2

We also used a custom SMOTE upsampling operator from the RapidMiner marketplace to improve our training set by making the distribution of different classes (DNA, RNA, DRNA, nonDRNA) more proportional. The inspiration for using this approach came from the slides, observing the dataset, and a quick Google search for SMOTE in Rapid Miner; furthermore, we

figured that we needed to provide more objects in the minority classes to increase the Accuracy and MCC for these classes. The parameters we chose for each of the minority classes (DNA, RNA, DRNA) are listed below.

Number of Neighbors: 5

Normalize: True

Equalize Classes: False

Upsampling Size: 1000* (for DRNA this is 1500)

Auto-Detect Minority Class: False

Minority Class: DNA, RNA, or DRNA

Results

Outcome	Quality measure	Baseline result	Design 1	Design 2	Design 3	Best Design
DNA	*Sensitivity*	6.9%	16.4%	17.6%	93.9%	57.3%
	Specificity	99.3%	96.4%	96.4%	96.0%	97.0%
	Accuracy	95.2%	92.8%	92.9%	95.9%	95.3%
	MCC	**0.132**	**0.131**	**0.143**	**0.681**	**0.496**
RNA	*Sensitivity*	39.6%	34.6%	42.6%	94.5%	71.1%
	Specificity	98.9%	96.9%	97.1%	97.7%	98.4%
	Accuracy	95.3%	93.2%	93.9%	97.5%	96.8%
	MCC	**0.501**	**0.343**	**0.421**	**0.814**	**0.706**
DRNA	*Sensitivity*	4.5%	0.0%	0.0%	81.8%	59.1%
	Specificity	100%	99.8%	99.9%	99.8%	99.6%
	Accuracy	99.7%	99.6%	99.6%	99.7%	99.5%
	MCC	**0.122**	**-0.002**	**-0.002**	**0.613**	**0.398**
nonDRNA	*Sensitivity*	98.6%	93.8%	94.0%	93.3%	95.2%

	Specificity	29.8%	35.4%	40.0%	96.2%	69.9%
	Accuracy	91.3%	87.6%	88.2%	93.6%	92.5%
	MCC	**0.428**	**0.309**	**0.354**	**0.748**	**0.625**
averageMCC		**0.296**	**0.195**	**0.229**	**0.714**	**0.556**
accuracy4labels		90.8%	86.6%	87.3%	93.3%	92.0%

As seen in the table, our first two designs underperformed against the baseline by a substantial margin. Our designs made a major improvement in the third design due to the discovery of the SMOTE upsampling operator. By experimenting with this, we were able to vastly improve compared to both our first two designs and the baseline provided in the original table. The 2nd iteration beats our first in accuracy due to us generating many more features, as explained in the Process section. Our first design was super simple, so it makes sense that the values for each of the fields would be rather low. After generating multiple new features from the Pfeature web interface and running this through our simple training model again, every single measure met or exceeded the corresponding value from the first design. We found the 3rd design, which vastly exceeded the 2nd and first designs in almost every aspect. The only values that did not meet or exceed the values from Design 2 were nonDRNA and DRNA sensitivity and DNA specificity. There were also a few fields that did not improve by much over the 2nd iteration, yet this fact is blown out of the water by the MCC values found for the 3rd design, which are about .5 higher on average. We all thought that the 3rd design was very good at the training, but we were concerned about overfitting occurring, so we kept tweaking until we found a decently good model that was better than the baseline and our first two models, but with a shorter tree to avoid some of the occurrences of overfitting that may have been occurring. Our 4th and final model performs

slightly worse than the 3rd design except for nonDRNA sensitivity and RNA and DNA specificity. Our confusion matrix for our final design is found below:

	true DNA	true RNA	true DRNA	true nonDRNA
pred. DNA	224	26	0	224
pred. RNA	16	372	0	118
pred. DRNA	1	2	13	32
pred. nonDRNA	150	123	9	7484

Our predictions on the blind test set can be summarized as follows:

Class	**Absolute Count (test set, training set)**	**Representation in Dataset (test set, training set)**
nonDRNA	7558, 7859	85.9%, 89.4%
RNA	594, 523	6.8%, 5.9%
DNA	542, 391	6.2%, 4.4%
DRNA	100, 22	1.1%, 0.3%

Conclusion

In conclusion, the best model we produced was the 4th iteration, which we created by utilizing the SMOTE operator to create a more balanced training data set. We find that our chosen best design compares favorably with the baseline results from Table 1. As a result of the oversampling techniques we implemented and the wide variety of different features we used for our training set, the average MCC values and average accuracy values recorded from our test set are markedly higher than those from the baseline results. The main reasons we performed better

on the final iteration than on both our initial designs and the baseline were our usage of the SMOTE operator to balance out the class proportions and our generation of many different features on the training dataset to feed into the Decision Tree operator. These two combined factors greatly impacted our final model's performance, as seen in our confusion matrix and metrics table. Our final model performed better on the DNA samples than the baseline result. We saw higher percentages in sensitivity and accuracy, yet slightly lower in specificity. The average MCC was also higher, which shows our predictions had more correlated results. Finally, we saw marginally higher percentages and MCC values in the other aspects for RNA, nonDRNA, and DRNA.

While analyzing our results, we identified that a disadvantage in our model was the frequent hurdle of our results overfitting. This disadvantage is evident in the difference in the average MCC values between the third and the final designs. However, we were able to turn this to our advantage by using those trials as stepping stones in the design process, which led to our Best Design case. After using the first three models, we saw that, across the board, the results were most consistent and gave us the highest accuracy compared to the baseline results. Further advantages of our model can be summarized by a list: Uses SMOTE to oversample underrepresented classes from the dataset to increase capacity to identify the minority classes, Uses a multitude of features generated from Pfeature each of the different characteristics of amino acid sequences to allow the model to use the optimal features at each branch, and the Decision Tree parameters have been fined tuned to ensure decent performance/prediction on the blind dataset. Another disadvantage of our model is that the model uses the somewhat arbitrary feature selection from Pfeature, which results in an incomplete understanding of what features

would be best to use and what the features represent (without additional research); additionally, a small number of classification models have been explored, so it is possible that we could obtain better results using other algorithms like Support Vector Machines or Random Forest.

Overall, we are pleased with our results during this project. We are hopeful that our results are of relatively high quality since we have attempted to adjust for any overfitting, underrepresentation, or noise in our model. We are relatively happy with our results because they show a not insignificant increase in different metrics like MCCs and Accuracy in comparison to the baseline results from Table 1. Additionally, it is comforting to see that we have steadily improved our results over time (as shown in Table 1) as a result of our choices to generate more features and use SMOTE. It was interesting to see how using SMOTE greatly bridged the gap between any inaccuracies we faced during the project. We are confident that we came close to the desired outputs without facing more instances of oversampling. We felt that completing this project was an overall positive experience, as it successfully taught us to process, generate, and analyze real-world data. This project undoubtedly helped prepare us for (potential) future careers in Data Science.