SDSS Galaxy Classification DR18

Assignment No.2: Supervised Learning

Bernardo Gil Alves Salgado (up202004493)
Michal Sztepiuk (up202302748)
Michal Dawid Kowalski (up202401554)

WORK SPECIFICATION

Objective: Compare different supervised learning algorithms in data classification

Database: SDSS Galaxy Classification DR18

"The Sloan Digital Sky Survey (SDSS) has searched about one-third of the sky and found around 1 billion objects and almost 3 million of those are galaxies. It contains 100,000 rows of photometric image data and the galaxy subclass is limited to two types, 'STARFORMING' or 'STARBURST'" [1]

Kaggle database

GitHub repository

RELATED WORKS

	(QUIESCENTLY) STARFORMING GALAXIES	STARBURST GALAXIES
Star Formation Rate (SFR)	Steady and extended	High and rapid
Duration	Long-lived	Short-lived (50-100 Myr)
Gas Depletion Time-Scale	Longer	Significantly shorter
Dominant Star Formation Region	Extended throughout the galaxy	Primarily in nuclear regions
Kennicutt-Schmidt Relation	Standard normalization	Higher normalization (4-10x)
Star Formation Efficiency (SFE)	Lower	Higher
Triggering Mechanism	Normal processes	Often triggered by mergers

(adapted from Hayward 2012 [2])

🔗 Random Forest vs Support Vector Machine vs Neural Network by chaitu_e6 [3]

🔗 Handling Imbalanced Dataset in Machine Learning: Easy Explanation for Data Science Interviews by Emma Ding [4]

🔗 Hyperparameters and tuning strategies for random forest by P. Probst, M. N. Wright, and A. Boulesteix [5]

🔗 Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift by Sergey Ioffe and Christian Szegedy [6]

TOOLS AND ALGORITHMS

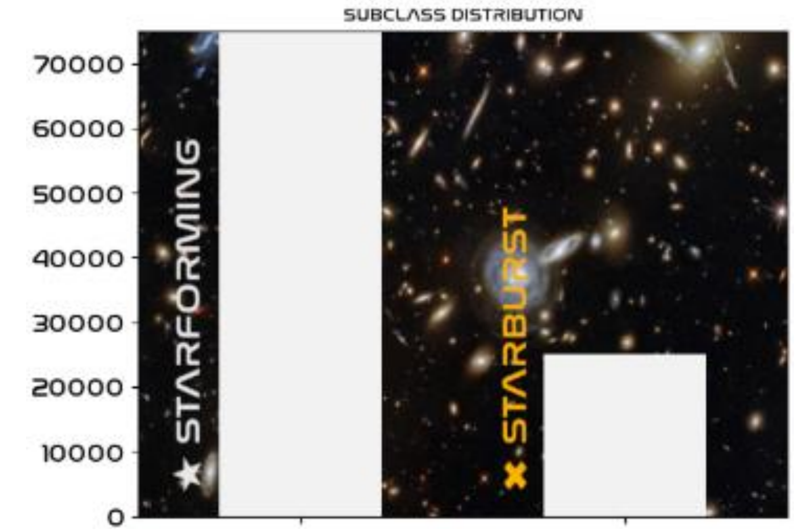
- Random Forest
- Support Vector Machine
- Neural Network

IMPLEMENTATION WORK

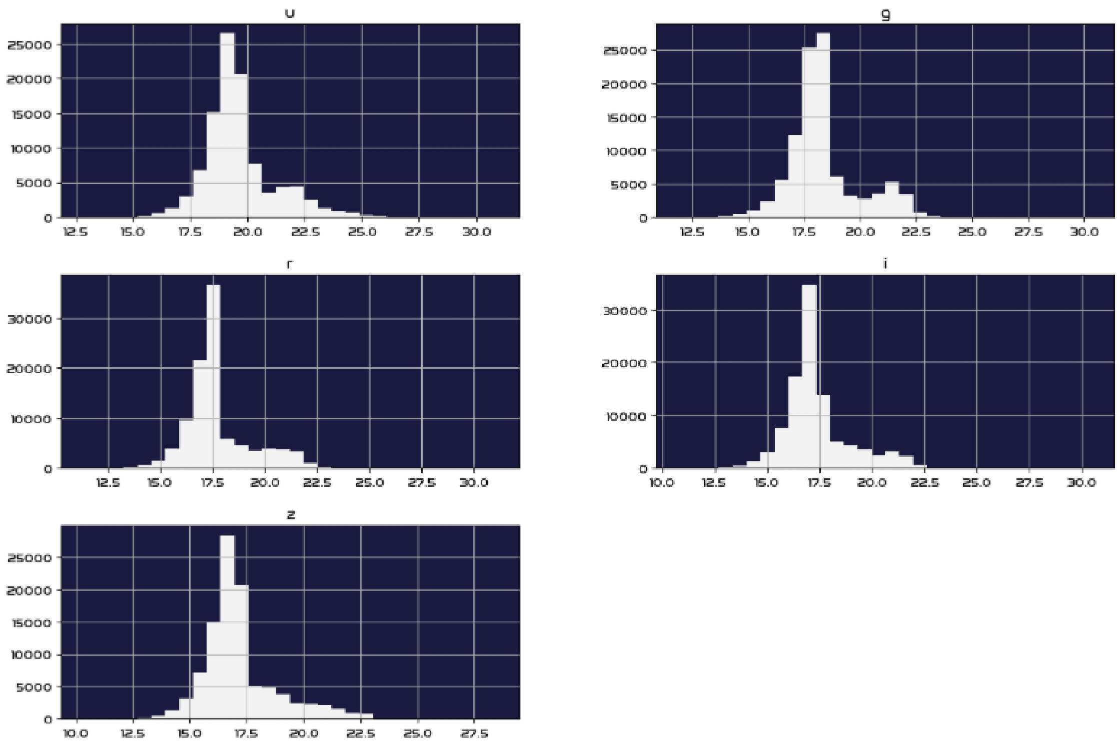
1. Preliminary analysis and dataset visualisation
2. Data preprocessing:
3. Model building
   - ✯ Random Forest
   - ✯ Support Vector Machine
   - ✯ Neural network
4. Training and testing on two datasets: base and reduced.

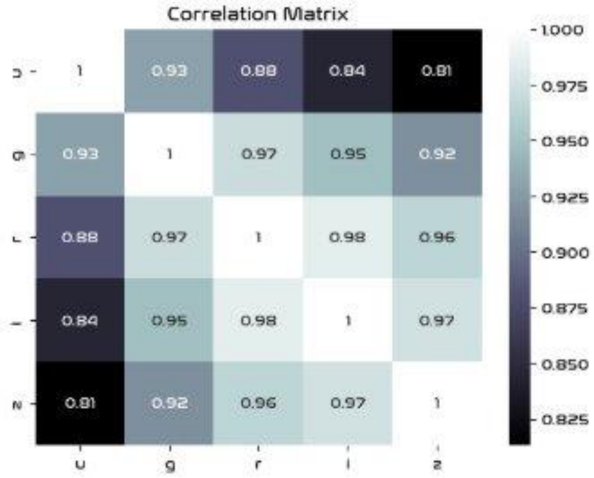
EXPLORATORY DATA ANALYSIS

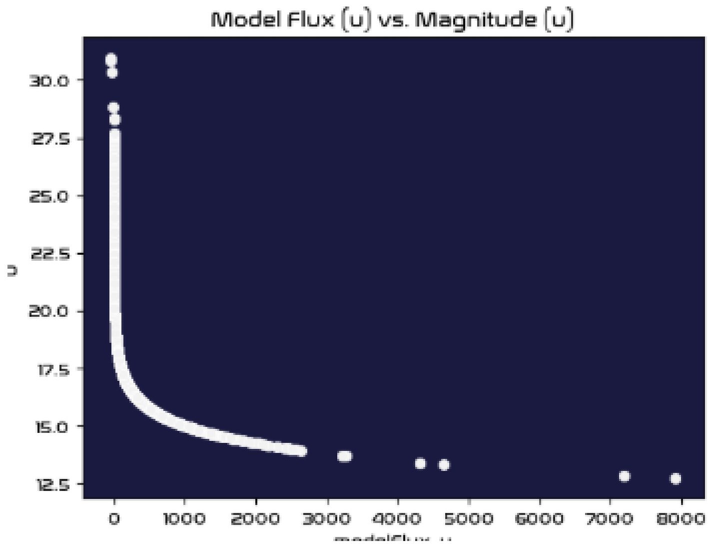
	objid	specobjid	ra	dec	u	g	r	i	z	modelFlux_u	...	psfMag_g	psfMag_i	psfMag_z	expAB_u	expAB_g	expAB_r	expAB_i	expAB_z	redshift	redshift_err
count	1.000000e+05	1.000000e+05	100000.000000	100000.000000	100000.000000	100000.000000	100000.000000	100000.000000	100000.000000	100000.000000	...	100000.000000	100000.000000	100000.000000	100000.000000	100000.000000	100000.000000	100000.000000	100000.000000	100000.000000	100000.000000
mean	1.237659e+18	2.303595e+18	180.577802	23.472475	18.518622	17.258221	16.821739	16.362611	15.850865	30.683321	...	18.834259	18.020203	17.435735	-0.603667	-0.522111	-0.309462	-0.410153	-0.740964	0.116753	0.000179
std	6.103756e+12	2.531359e+18	75.751994	21.140744	105.082004	105.069066	95.035474	100.171155	114.206165	76.552859	...	105.079620	100.181687	114.218604	104.870665	104.871474	94.860919	99.991654	114.005927	0.100169	0.052189
min	1.237646e+18	2.994897e+17	0.008745	-11.244273	-9999.000000	-9999.000000	-9999.000000	-9999.000000	-9999.000000	-47.451720	...	-9999.000000	-9999.000000	-9999.000000	-9999.000000	-9999.000000	-9999.000000	-9999.000000	-9999.000000	-0.000833	0.000002
25%	1.237655e+18	8.130687e+17	138.741880	3.120118	18.762215	17.505868	16.898845	16.527097	16.281327	9.288132	...	19.257783	18.295627	17.991602	0.299999	0.398705	0.418789	0.418656	0.381288	0.055836	0.000008
50%	1.237659e+18	1.457564e+18	181.492972	20.913596	19.349715	18.072640	17.459080	17.091385	16.861105	18.195690	...	19.763915	18.845780	18.563315	0.508688	0.588335	0.604795	0.604254	0.575397	0.085850	0.000011
75%	1.237663e+18	2.367902e+18	223.851863	42.259965	20.079470	18.656182	17.926918	17.592650	17.453848	31.259628	...	20.408775	19.586577	19.299430	0.699907	0.768804	0.773924	0.773119	0.752311	0.135148	0.000015
max	1.237681e+18	1.412691e+19	359.997922	68.695258	30.960000	30.420980	31.173560	30.562360	28.553240	7915.306000	...	26.174400	25.966680	27.043280	1.000000	1.000000	0.999999	1.000000	0.999998	0.572899	16.503710

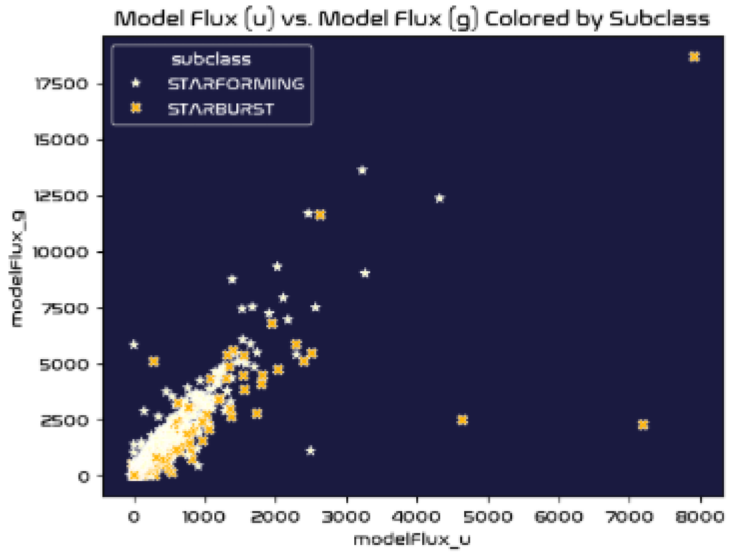


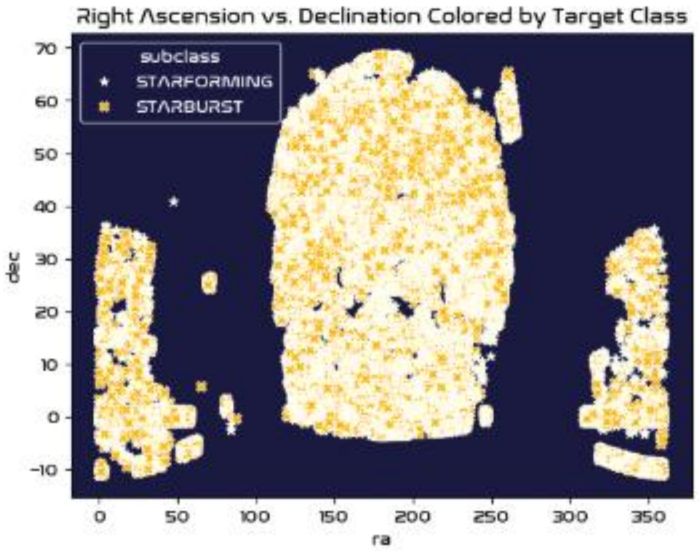
Histograms of u, g, r, i, z Features

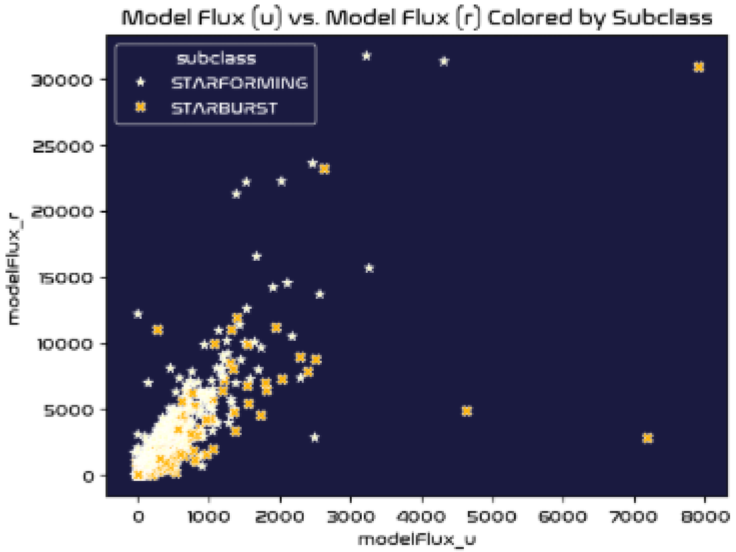


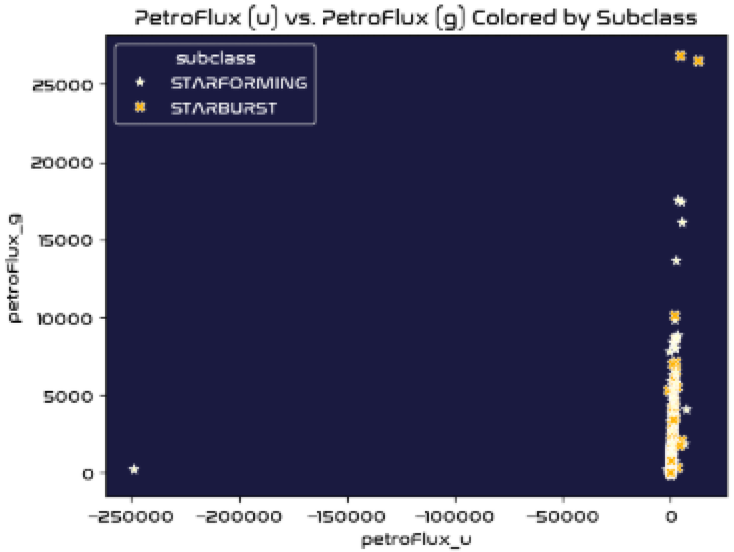


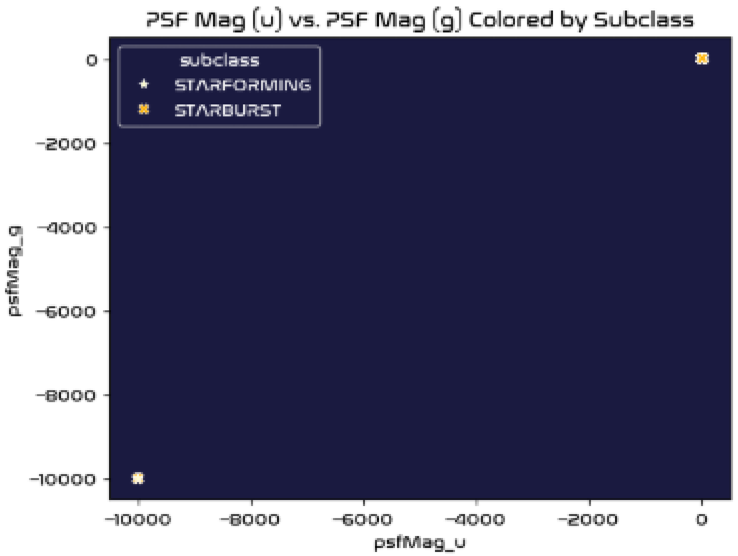

Model Flux (u) vs. Magnitude (u)
u


Model Flux (u) vs. Model Flux (g) Colored by Subclass
subclass
STARFORMING
STARBURST
modelFlux_g
modelFlux_u


Right Ascension vs. Declination Colored by Target Class
subclass
STARFORMING
STARBURST
dec
ra

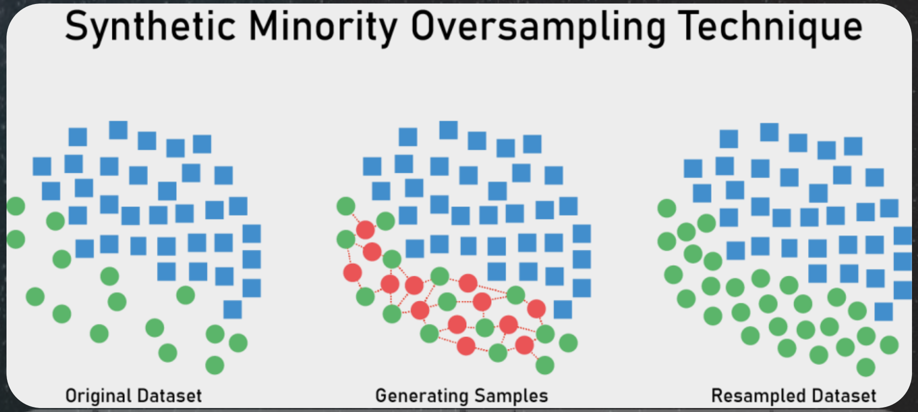

Model Flux (u) vs. Model Flux (r) Colored by Subclass
subclass
STARFORMING
STARBURST
modelFlux_r
modelFlux_u


PetroFlux (u) vs. PetroFlux (g) Colored by Subclass
subclass
STARFORMING
STARBURST
petroFlux_g
petroFlux_u


PSF Mag (u) vs. PSF Mag (g) Colored by Subclass
subclass
STARFORMING
STARBURST
psfMag_g
psfMag_u

DATA PRE-PROCESSING

1. Removing missing values (-9999 placeholders)
2. Dropping irrelevant features: class, objide, specobjid
3. [Selection of best features for problem simplification and avoidance of overfitting]
4. Reming outliers (4 or more features)
5. Scaling values to the same range (Min-max scaling)
6. 80 training / 20 testing split.
7. Handling of imbalance (SMOTE) [4, 5:35]



➡️ Two datasets: base and reduced.

MODEL BUILDING: 1. RANDOM FOREST

Hyperparameters:

n_estimators – Randomly sampled between 100 and 300

max_depth: none, 10, 20, ... , 90, 100

min_samples_split: Random sample between 2 and 20

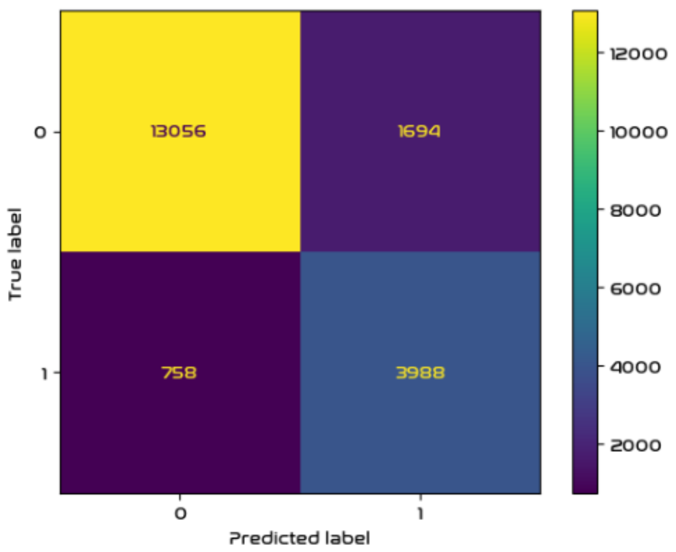
min_samples_leaf: Random sample between 1 and 20

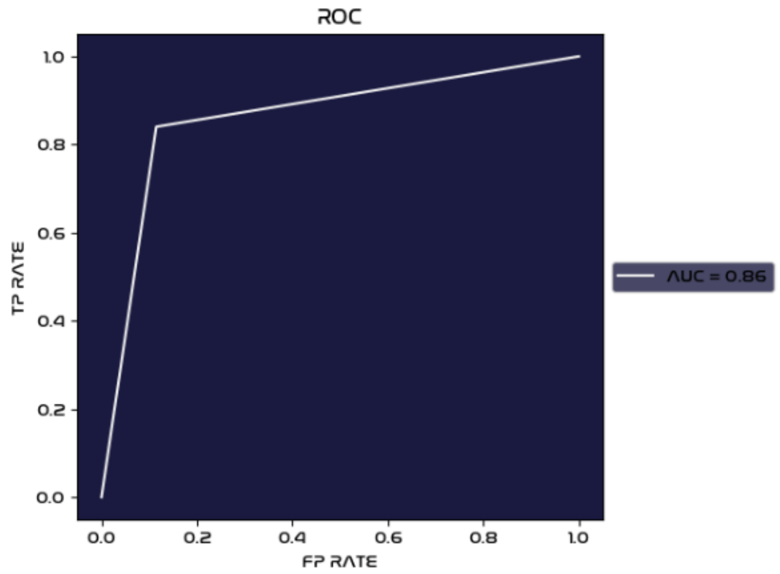
max_features: sqrt / log2 / none

1. **Parameter optimisation (RandomizedSearchCV**) - chosen over grid search to efficiently explore a larger hyperparameter space
2. **3-Fold Cross-Validation** – to avert reliance on a single train-test split and provide a better estimate of the model's generalisation [5]
3. **F1-Score** – selected as scoring metric – useful for imbalanced datasets
4. **random_state** – the parameter to ensure reproducibility
5. **Boostrap** – the parameter to use different subsets for each tree to enhance the diversity of trees and improve performance
6. **Comprehensive evaluation** – classification reports, confusion matrices, ROC-AUC curves to understand the model's strengths and weaknesses and not just accuracy
7. **Fitting on full dataset** – after defininig the best hyperparameters, retraining the model on the entire dataset (after SMOTE) before making predictions on the test set

RF PERFORMANCE ON ENTIRE DATASET

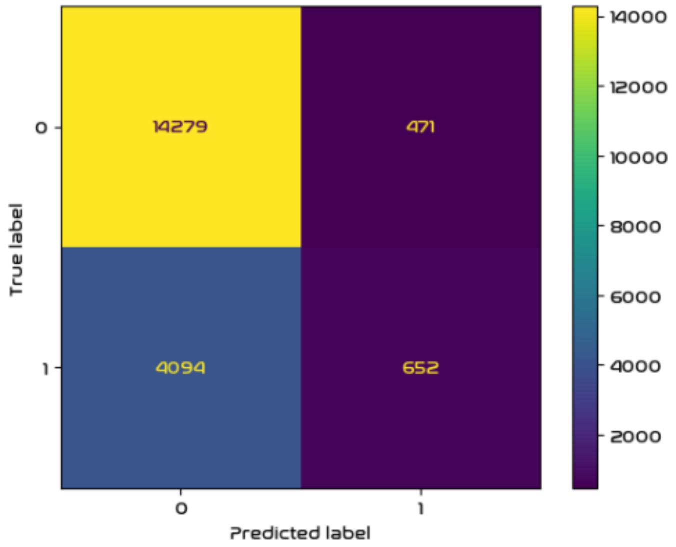
	precision	recall	f1-score	support
0	0.95	0.89	0.91	14750
1	0.70	0.84	0.76	4746
accuracy			0.87	19496
macro avg	0.82	0.86	0.84	19496
weighted avg	0.89	0.87	0.88	19496

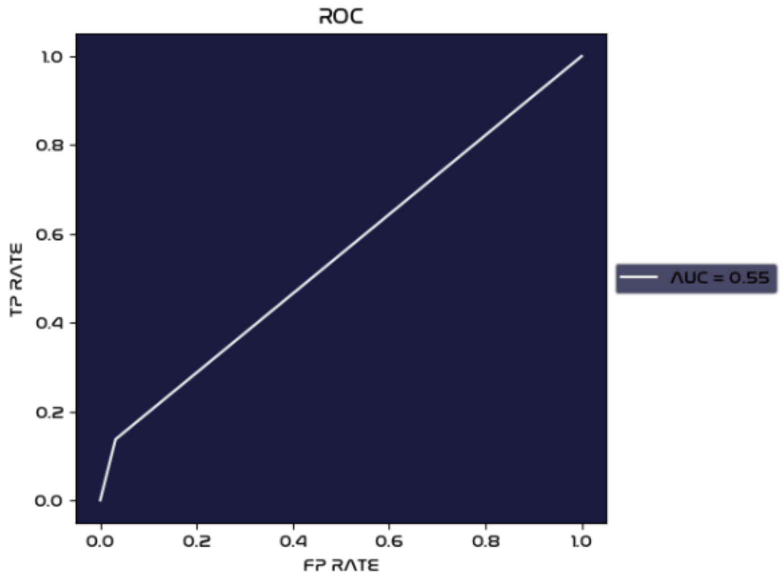




RF PERFORMANCE ON REDUCED DATASET

	precision	recall	f1-score	support
0	0.78	0.97	0.86	14750
1	0.58	0.14	0.22	4746
accuracy			0.77	19496
macro avg	0.68	0.55	0.54	19496
weighted avg	0.73	0.77	0.71	19496





2. NEURAL NETWORK

No. 1: Baseline model with a straightforward architecture ↓

No. 2: Enhanced with L2 Regularisation ↓

No. 3: Most refined, enhanced further with Batch Normalisation to improve stability and speed. ↓

ARCHITECTURE & HYPERPARAMETERS	MODEL 1	MODEL 2	MODEL 3	
1st Hidden Layer	128 neurons, ReLU, Dropout (0.3)	128 neurons, ReLU, Dropout (0.1)	128 neurons, ReLU, Dropout (0.1)	← lower dropout rates in 2 and 3 may help retain more information during training while still providing some regularisation
2nd Hidden Layer	64 neurons, ReLU, Dropout (0.3)	64 neurons, ReLU, Dropout (0.2)	64 neurons, ReLU, Dropout (0.2)	
3rd Hidden Layer	32 neurons, ReLU, Dropout (0.3)	32 neurons, ReLU, Dropout (0.3)	32 neurons, ReLU, Dropout (0.3)	
Output Layer	1 neuron, Sigmoid			← standard for binary classification
Regularisation	None	L2 (λ = 0.001)	L2 (λ = 0.001)	← "imposes a penalty on the sum of squared feature coefficients (...) offers enhanced stability and mitigates the risk of overfitting"[6, p. 10][1]
Batch Normalization	No	No	Yes	← Can accelerate training and improve model stability by normalizing the inputs to each layer [6]
Optimizer	Adam (learning rate = 0.001)			← Adjusts weights during training, combining the benefits of both AdaGrad and RMSProp for faster convergence
Loss Function	Binary Crossentropy			← measures how well the predicted probabilities match the actual binary outcomes
Batch Size	64			← model updates weights after processing 64 samples – balances memory efficiency and convergence speed
Epochs	50			← sufficient iterations for the model to learn data patterns while allowing early stopping to prevent overfitting
Validation Split	20%			← a portion of the training data is put to the side to evaluate the model's performance, monitoring generalisation
Early Stopping	Yes (patience = 10)			← helps prevent overfitting by monitoring validation loss (training stops when the models starts to generalise badly)

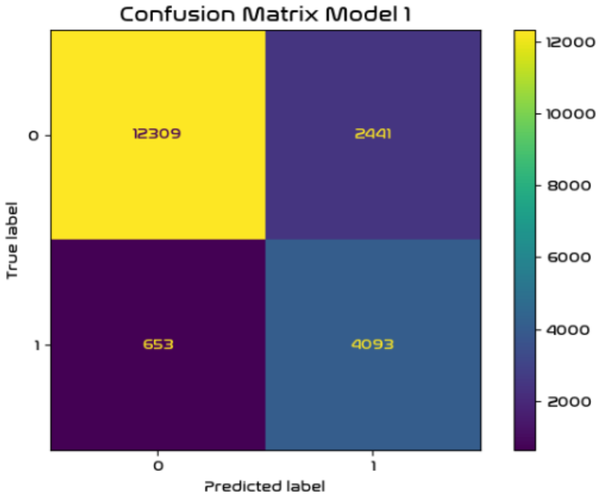
[1]Primary source: A. Arkan and M. Ahmadi, "An unsupervised and hierarchical intrusion detection system for software-defined wireless sensor networks," J. Supercomput., vol. 2023, pp. 1–27.

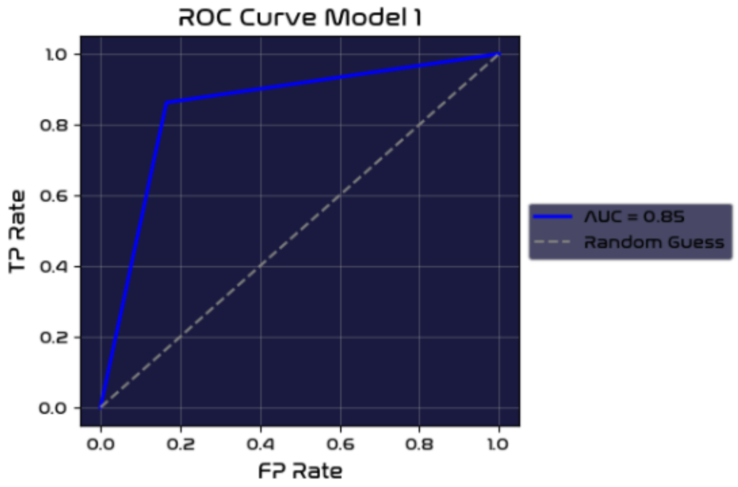
MODEL 1 PERFORMANCE

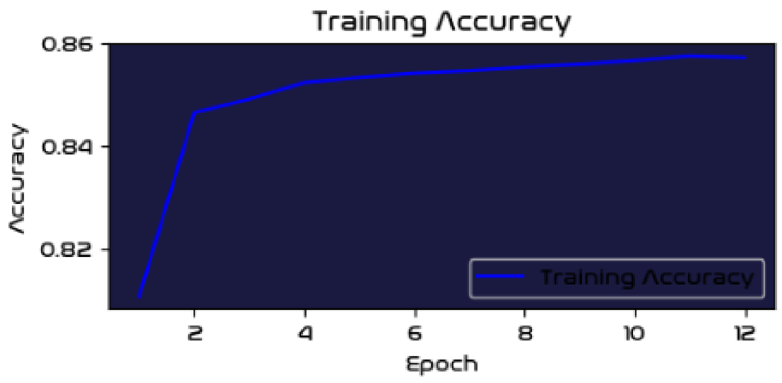
Epoch 1/50 **1470/1470** **6s** 3ms/step - accuracy: 0.7605 - loss: 0.5039 - val_accuracy: 0.7375 - val_loss: 0.5875
Epoch 2/50 **1470/1470** **3s** 2ms/step - accuracy: 0.8441 - loss: 0.3687 - val_accuracy: 0.8954 - val_loss: 0.3076
Epoch 3/50 **1470/1470** **3s** 2ms/step - accuracy: 0.8482 - loss: 0.3592 - val_accuracy: 0.8213 - val_loss: 0.4239
Epoch 4/50 **1470/1470** **4s** 3ms/step - accuracy: 0.8527 - loss: 0.3544 - val_accuracy: 0.7858 - val_loss: 0.4716
Epoch 5/50 **1470/1470** **4s** 3ms/step - accuracy: 0.8522 - loss: 0.3508 - val_accuracy: 0.8177 - val_loss: 0.4192
Epoch 6/50 **1470/1470** **4s** 2ms/step - accuracy: 0.8532 - loss: 0.3498 - val_accuracy: 0.7292 - val_loss: 0.6031
Epoch 7/50 **1470/1470** **4s** 3ms/step - accuracy: 0.8550 - loss: 0.3462 - val_accuracy: 0.8683 - val_loss: 0.3517
Epoch 8/50 **1470/1470** **4s** 3ms/step - accuracy: 0.8553 - loss: 0.3435 - val_accuracy: 0.8188 - val_loss: 0.4224

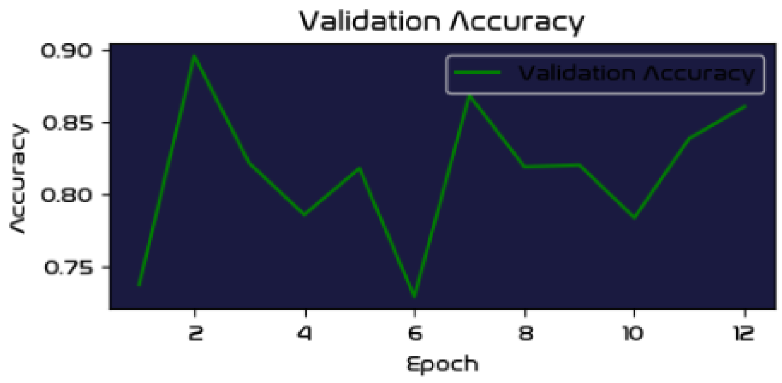
Classification Report:

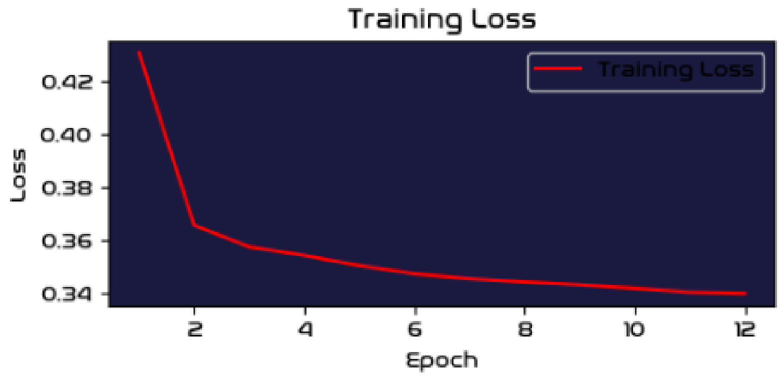
	precision	recall	f1-score	support
0	0.95	0.83	0.89	14750
1	0.63	0.86	0.73	4746
accuracy			0.84	19496
macro avg	0.79	0.85	0.81	19496
weighted avg	0.87	0.84	0.85	19496









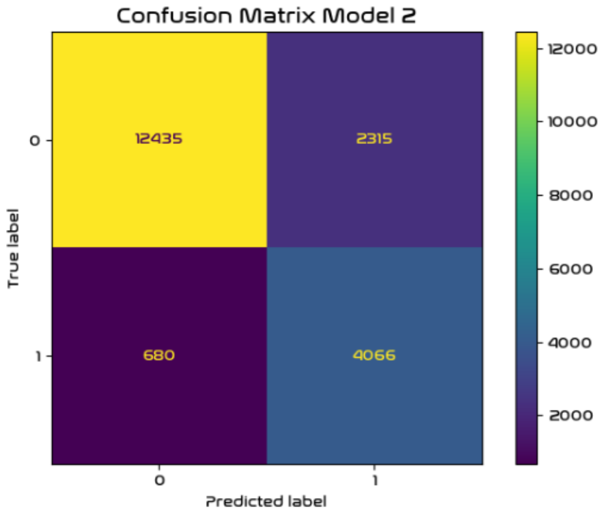


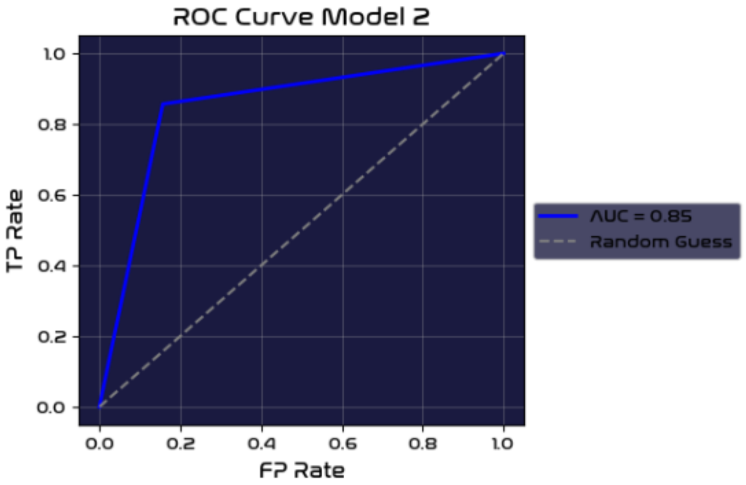
MODEL 2 PERFORMANCE

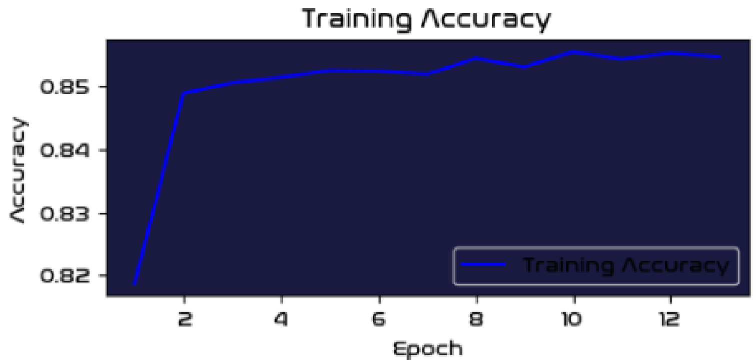
Epoch 1/50 **1470/1470** ━━━━━━━━━━━━━━━━━━━━ **5s** 2ms/step - accuracy: 0.7684 - loss: 0.5647 - val_accuracy: 0.7770 - val_loss: 0.5447
Epoch 2/50 **1470/1470** ━━━━━━━━━━━━━━━━━━━━ **4s** 2ms/step - accuracy: 0.8479 - loss: 0.4032 - val_accuracy: 0.8436 - val_loss: 0.4154
Epoch 3/50 **1470/1470** ━━━━━━━━━━━━━━━━━━━━ **3s** 2ms/step - accuracy: 0.8483 - loss: 0.3914 - val_accuracy: 0.8894 - val_loss: 0.3484
Epoch 4/50 **1470/1470** ━━━━━━━━━━━━━━━━━━━━ **4s** 3ms/step - accuracy: 0.8499 - loss: 0.3852 - val_accuracy: 0.7979 - val_loss: 0.4922
Epoch 5/50 **1470/1470** ━━━━━━━━━━━━━━━━━━━━ **5s** 3ms/step - accuracy: 0.8548 - loss: 0.3757 - val_accuracy: 0.8099 - val_loss: 0.4668
Epoch 6/50 **1470/1470** ━━━━━━━━━━━━━━━━━━━━ **4s** 3ms/step - accuracy: 0.8531 - loss: 0.3759 - val_accuracy: 0.7691 - val_loss: 0.5346
Epoch 7/50 **1470/1470** ━━━━━━━━━━━━━━━━━━━━ **5s** 3ms/step - accuracy: 0.8506 - loss: 0.3794 - val_accuracy: 0.7828 - val_loss: 0.5105
Epoch 8/50 **1470/1470** ━━━━━━━━━━━━━━━━━━━━ **5s** 3ms/step - accuracy: 0.8560 - loss: 0.3701 - val_accuracy: 0.8897 - val_loss: 0.3519

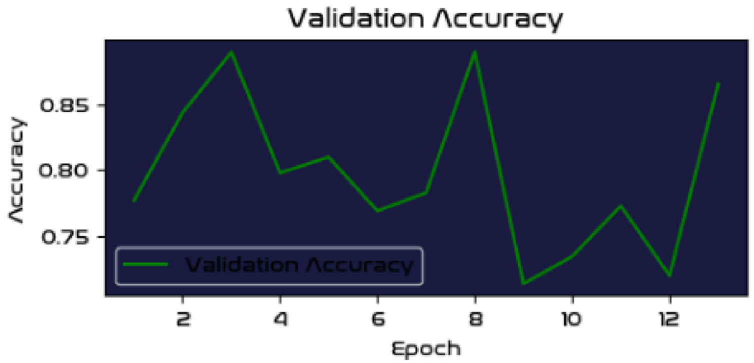
Classification Report:

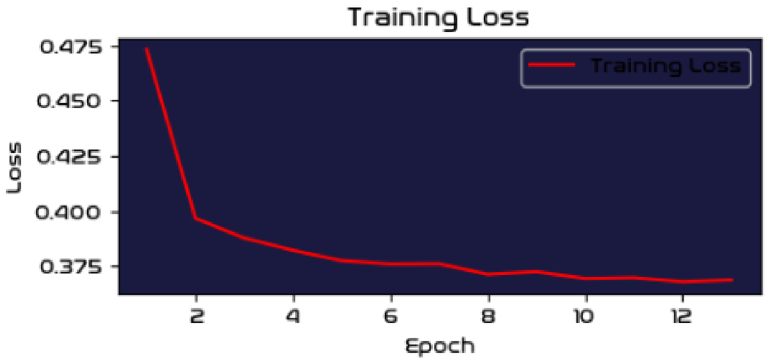
	precision	recall	f1-score	support
0	0.95	0.84	0.89	14750
1	0.64	0.86	0.73	4746
accuracy			0.85	19496
macro avg	0.79	0.85	0.81	19496
weighted avg	0.87	0.85	0.85	19496











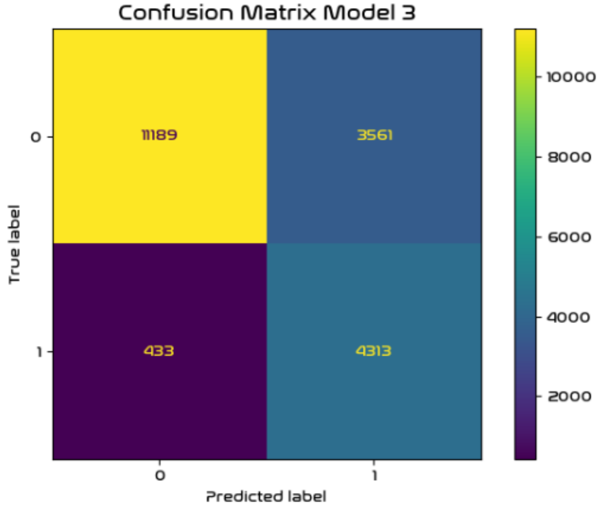
MODEL 3 PERFORMANCE

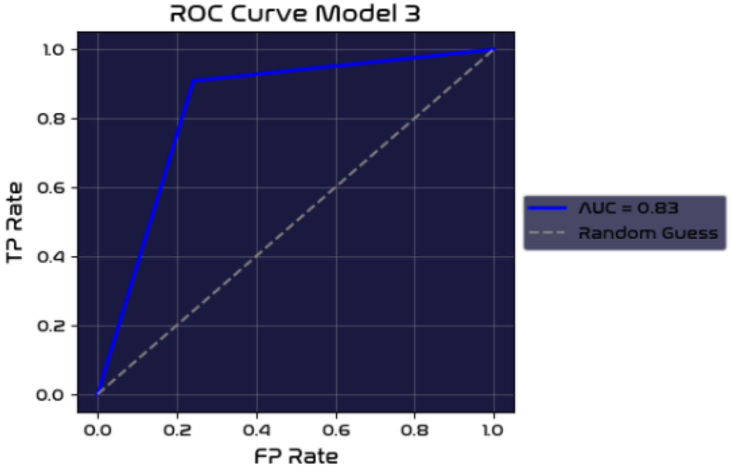
Epoch 1/50 **1470/1470** ━━━━━━━━━━━━━━━━━━━━ **10s** 4ms/step - accuracy: 0.7903 - loss: 0.6022 - val_accuracy: 0.8431 - val_loss: 0.4588
Epoch 2/50 **1470/1470** ━━━━━━━━━━━━━━━━━━━━ **8s** 5ms/step - accuracy: 0.8547 - loss: 0.4006 - val_accuracy: 0.9327 - val_loss: 0.2710
Epoch 3/50 **1470/1470** ━━━━━━━━━━━━━━━━━━━━ **8s** 5ms/step - accuracy: 0.8554 - loss: 0.3675 - val_accuracy: 0.7696 - val_loss: 0.5863
Epoch 4/50 **1470/1470** ━━━━━━━━━━━━━━━━━━━━ **7s** 5ms/step - accuracy: 0.8574 - loss: 0.3585 - val_accuracy: 0.9086 - val_loss: 0.3034
Epoch 5/50 **1470/1470** ━━━━━━━━━━━━━━━━━━━━ **7s** 4ms/step - accuracy: 0.8572 - loss: 0.3552 - val_accuracy: 0.7069 - val_loss: 0.6141
Epoch 6/50 **1470/1470** ━━━━━━━━━━━━━━━━━━━━ **7s** 5ms/step - accuracy: 0.8593 - loss: 0.3500 - val_accuracy: 0.9168 - val_loss: 0.2722
Epoch 7/50 **1470/1470** ━━━━━━━━━━━━━━━━━━━━ **8s** 6ms/step - accuracy: 0.8578 - loss: 0.3481 - val_accuracy: 0.9368 - val_loss: 0.2123
Epoch 8/50 **1470/1470** ━━━━━━━━━━━━━━━━━━━━ **9s** 6ms/step - accuracy: 0.8581 - loss: 0.3486 - val_accuracy: 0.8908 - val_loss: 0.2996

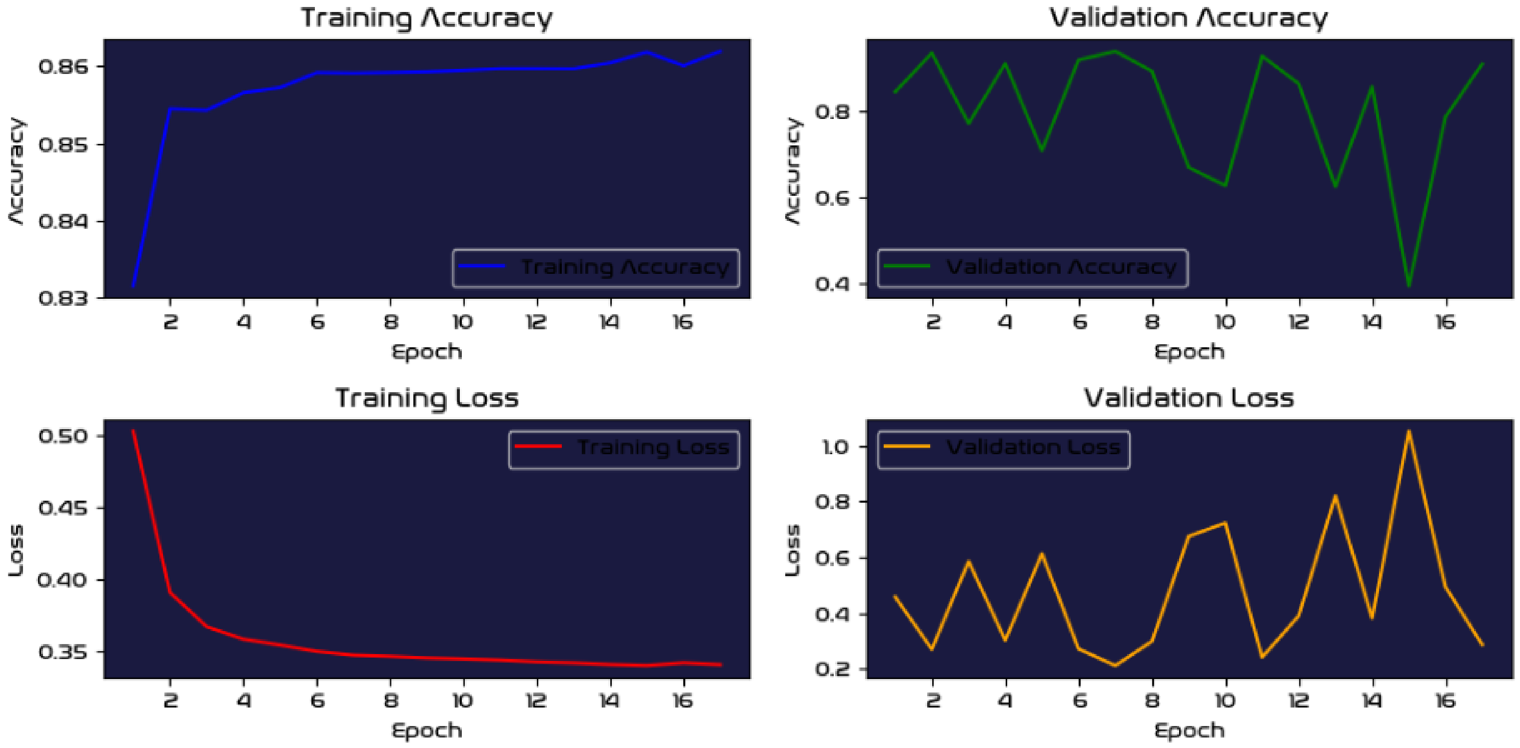
```
Classification Report:
              precision    recall  f1-score   support

           0       0.96      0.76      0.85     14750
           1       0.55      0.91      0.68      4746

    accuracy                           0.80     19496
   macro avg       0.76      0.83      0.77     19496
weighted avg       0.86      0.80      0.81     19496
```







3. SUPPORT VECTOR MACHINE (SVM)

Hyperparameter candidates:

C (regularisation parameter): [0.1, 1, 10, 100]

Kernel: ['linear', 'rbf', 'poly']

Gamma (kernel coefficient): ['scale', 'auto']

Degree (of the polynomial kernel): [2, 3, 4]

Best Hyperparameters after GridSearchCV:

C: 100

Kernel: 'poly'

Gamma: 'scale'

Degree: 4

GridSearchCV – used to explore combinations of hyperparameters, optimising for the F1-score with 3-fold cross-validation

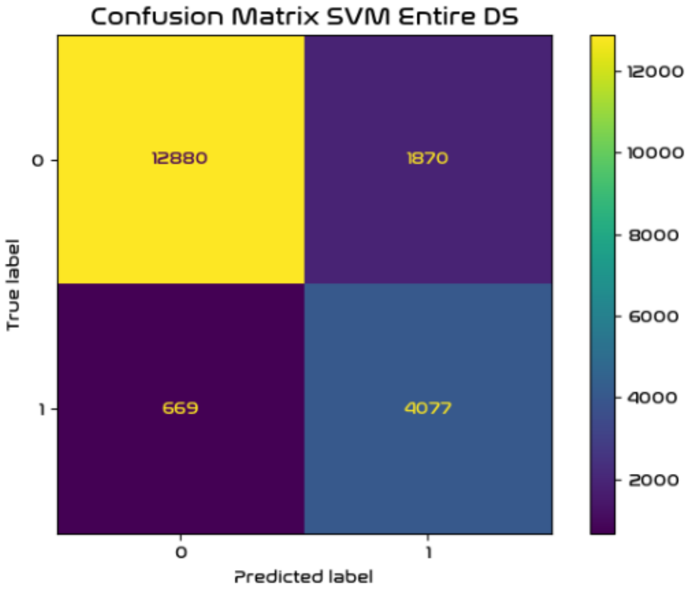
Model Fitting – The best SVM model was fitted using the entire dataset after tuning to ensure that the data is maximally utilised

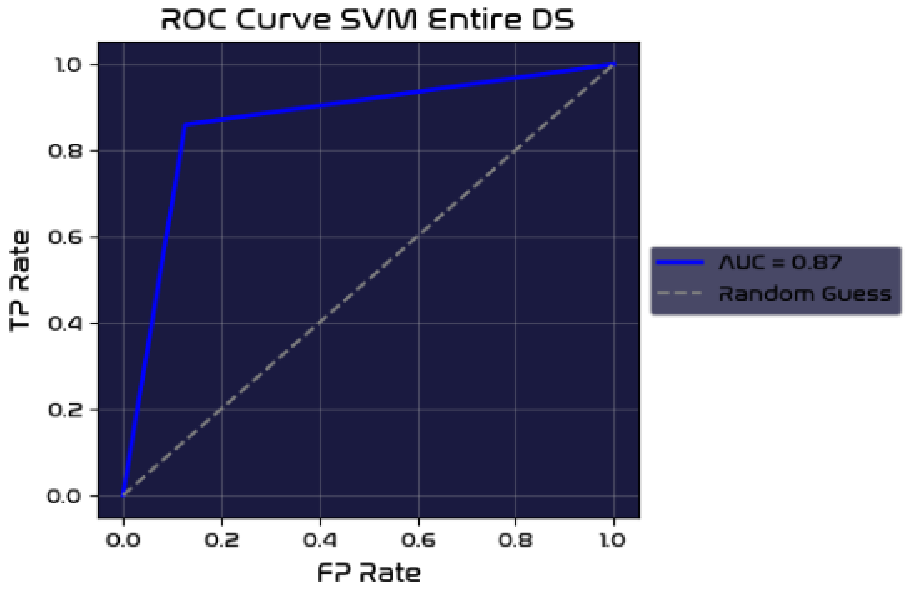
Permance evaluation – Predictions made on the test set, then performance was assessed using classification reports, confusion matrices, and ROC curves

Robustness – The SVM model is effective for high-dimensional spaces, particularly so for binary classification tasks, making it suitable for this dataset

SVM PERFORMANCE ON ENTIRE DATASET

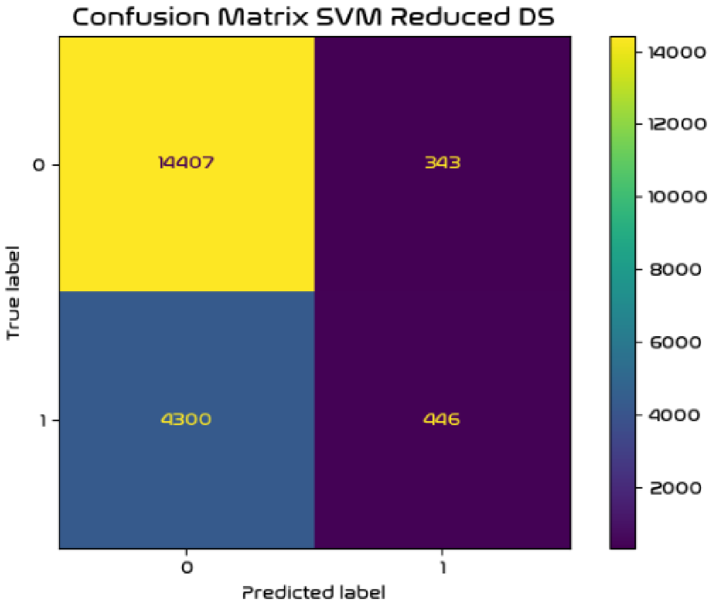
	precision	recall	f1-score	support
0	0.95	0.87	0.91	14750
1	0.69	0.86	0.76	4746
accuracy			0.87	19496
macro avg	0.82	0.87	0.84	19496
weighted avg	0.89	0.87	0.87	19496

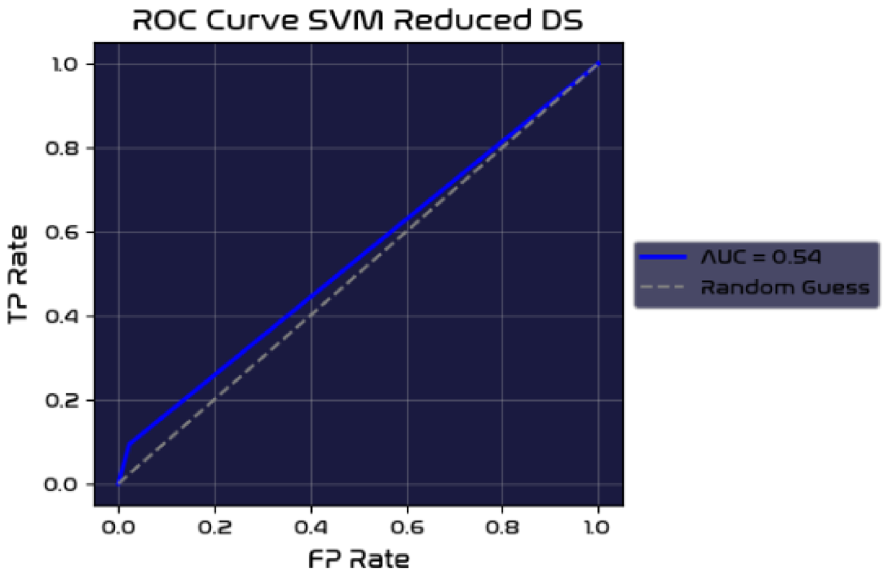




SVM PERFORMANCE ON REDUCED DATASET

	precision	recall	f1-score	support
0	0.77	0.98	0.86	14750
1	0.57	0.09	0.16	4746
accuracy			0.76	19496
macro avg	0.67	0.54	0.51	19496
weighted avg	0.72	0.76	0.69	19496





COMPARISON: CLASSIFICATION REPORT

RANDOM FOREST

RF ON ENTIRE DATASET

	precision	recall	f1-score	support
0	0.95	0.89	0.91	14750
1	0.70	0.84	0.76	4746
accuracy			0.87	19496
macro avg	0.82	0.86	0.84	19496
weighted avg	0.89	0.87	0.88	19496

RF ON REDUCED DATASET

	precision	recall	f1-score	support
0	0.78	0.97	0.86	14750
1	0.58	0.14	0.22	4746
accuracy			0.77	19496
macro avg	0.68	0.55	0.54	19496
weighted avg	0.73	0.77	0.71	19496

NEURAL NETWORK

NN MODEL 1

Classification Report:

	precision	recall	f1-score	support
0	0.95	0.83	0.89	14750
1	0.63	0.86	0.73	4746
accuracy			0.84	19496
macro avg	0.79	0.85	0.81	19496
weighted avg	0.87	0.84	0.85	19496

NN MODEL 2

Classification Report:

	precision	recall	f1-score	support
0	0.95	0.84	0.89	14750
1	0.64	0.86	0.73	4746
accuracy			0.85	19496
macro avg	0.79	0.85	0.81	19496
weighted avg	0.87	0.85	0.85	19496

NN MODEL 3

Classification Report:

	precision	recall	f1-score	support
0	0.96	0.76	0.85	14750
1	0.55	0.91	0.68	4746
accuracy			0.80	19496
macro avg	0.76	0.83	0.77	19496
weighted avg	0.86	0.80	0.81	19496

SUPPORT VECTOR MACHINE

SVM ON ENTIRE DATASET

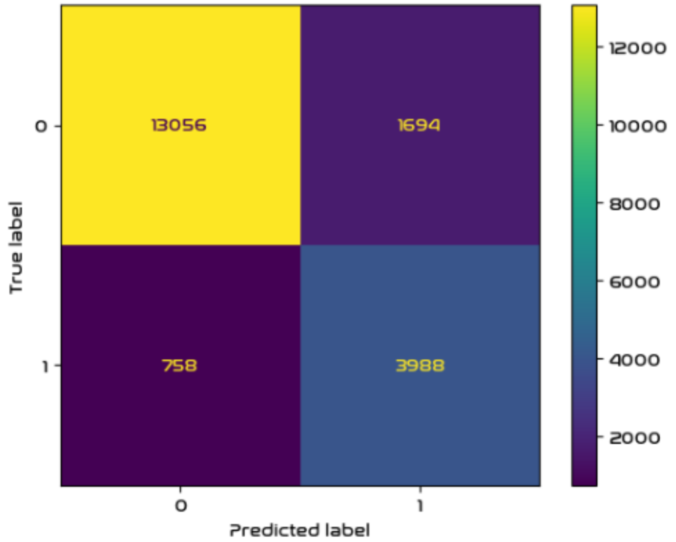
	precision	recall	f1-score	support
0	0.95	0.87	0.91	14750
1	0.69	0.86	0.76	4746
accuracy			0.87	19496
macro avg	0.82	0.87	0.84	19496
weighted avg	0.89	0.87	0.87	19496

SVM ON REDUCED DATASET

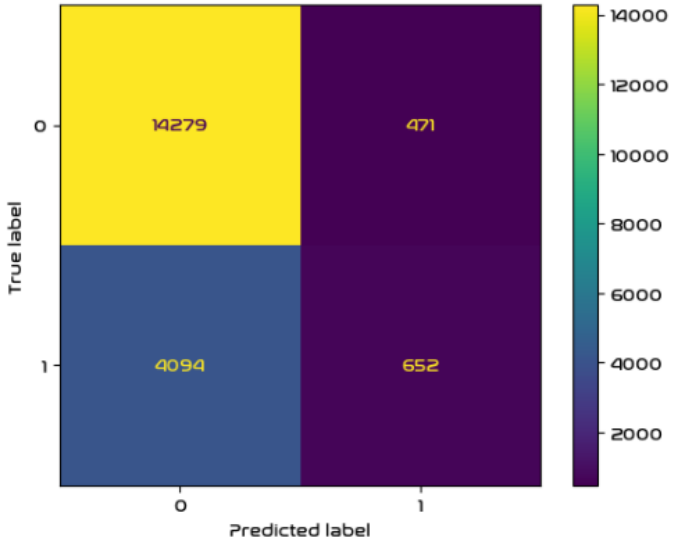
	precision	recall	f1-score	support
0	0.77	0.98	0.86	14750
1	0.57	0.09	0.16	4746
accuracy			0.76	19496
macro avg	0.67	0.54	0.51	19496
weighted avg	0.72	0.76	0.69	19496

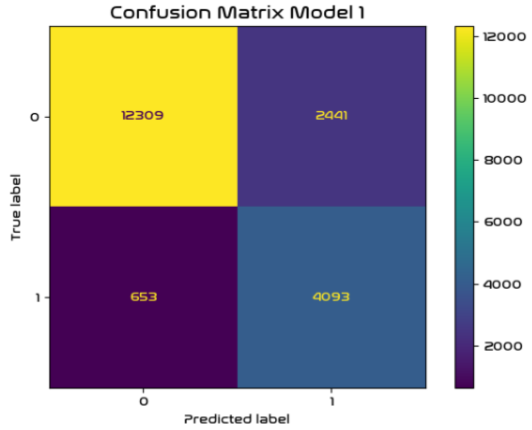
COMPARISON: CONFUSION MATRIX (RF | NN | SVM)

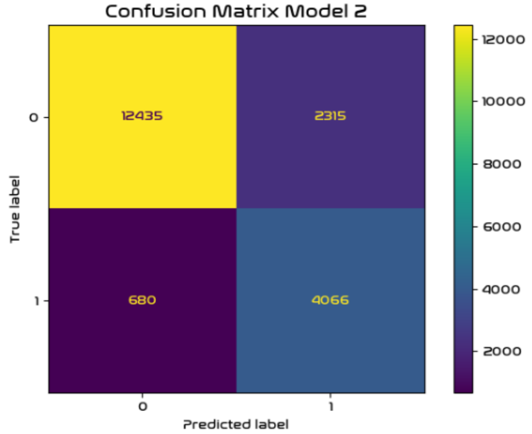
RF ON ENTIRE DATASET

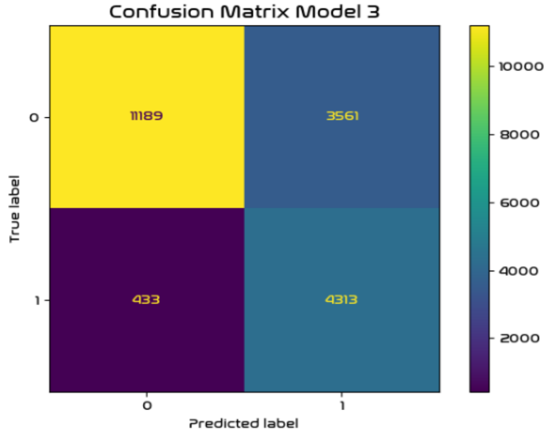


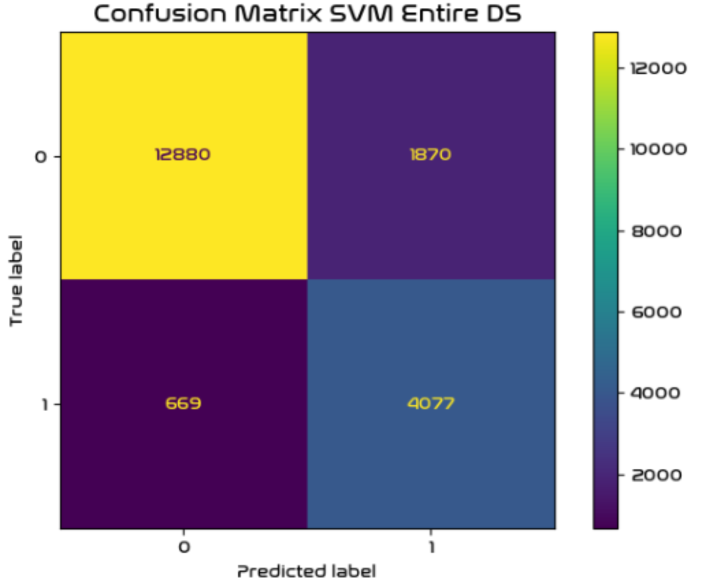
RF ON REDUCED DATASET

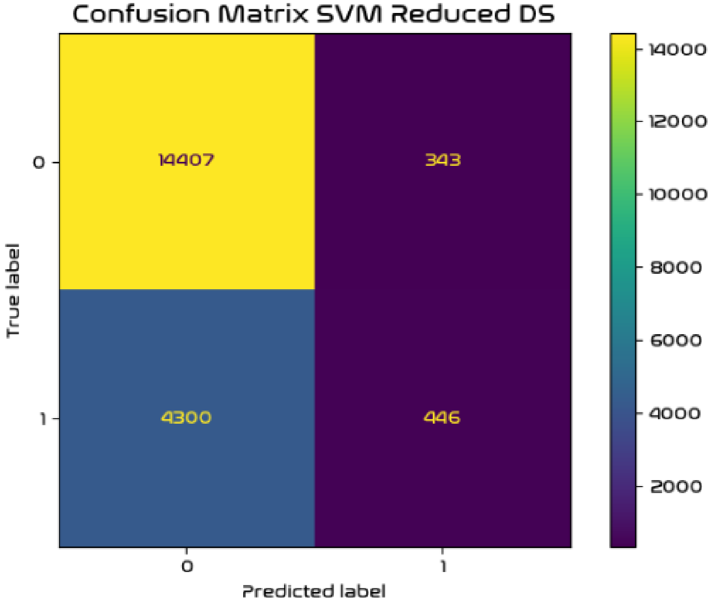






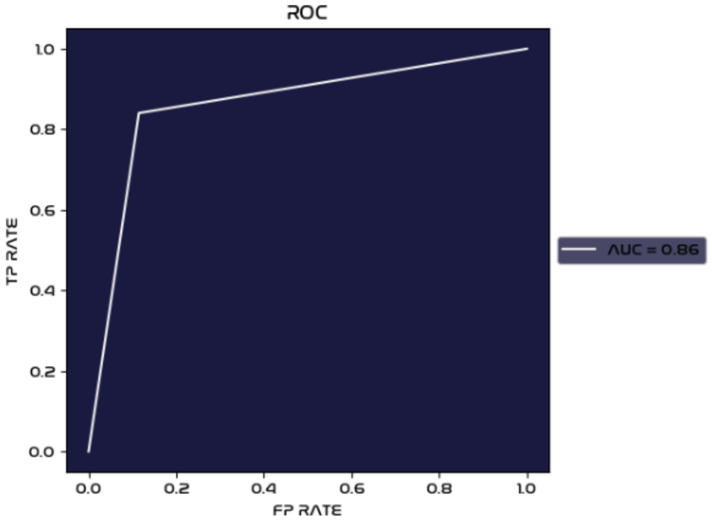




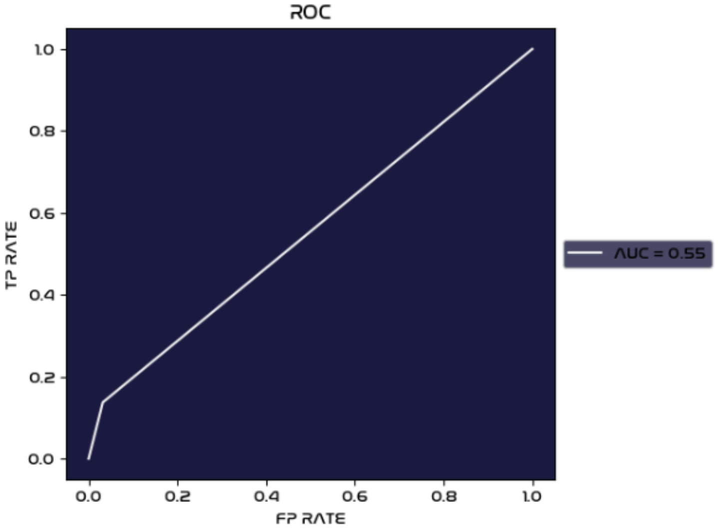


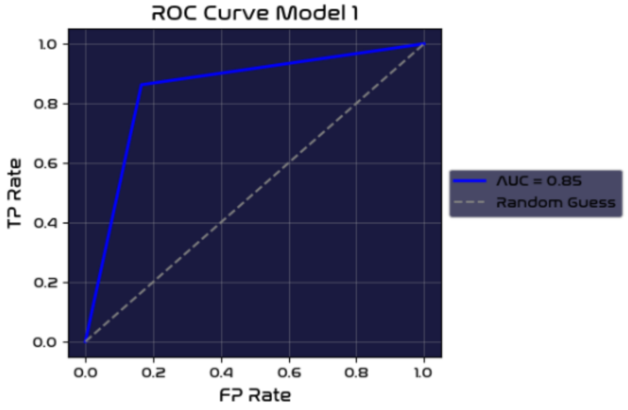
COMPARISON: ROC CURVE (RF | NN | SVM)

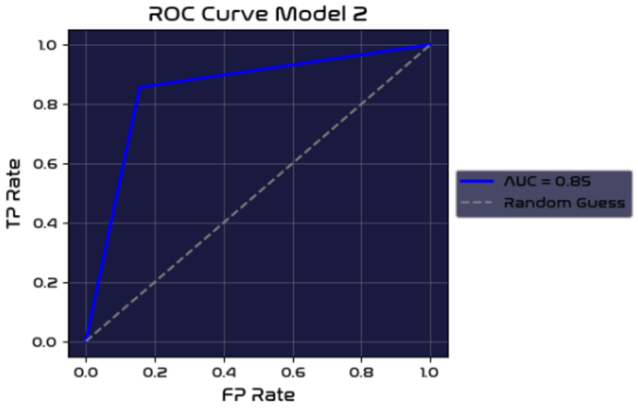
RF ON ENTIRE DATASET

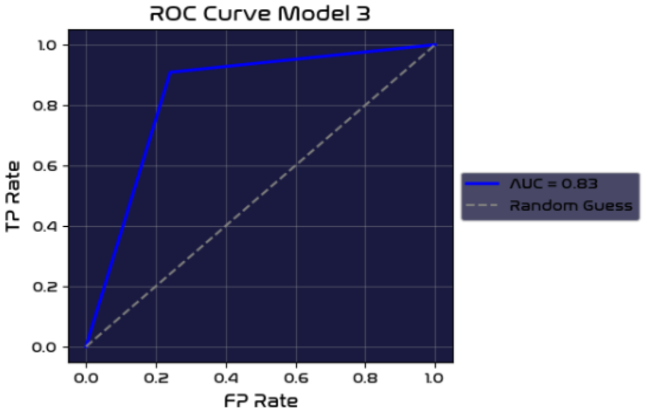


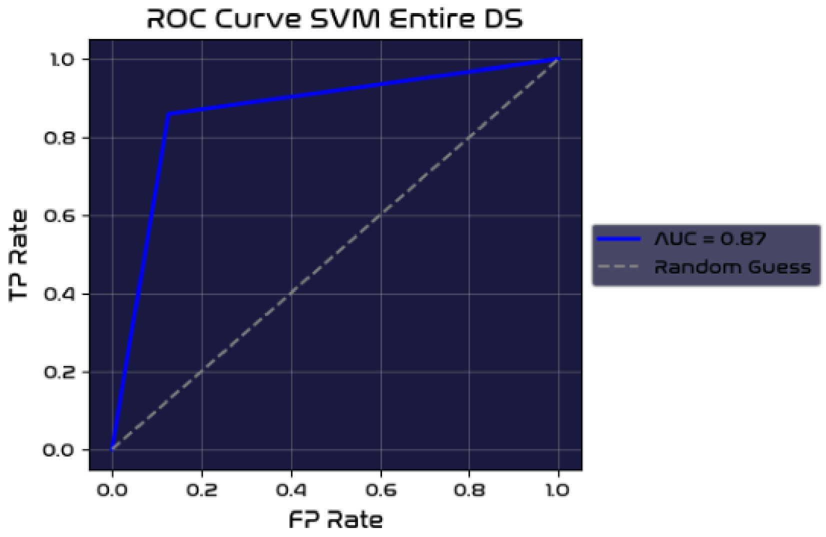
RF ON REDUCED DATASET

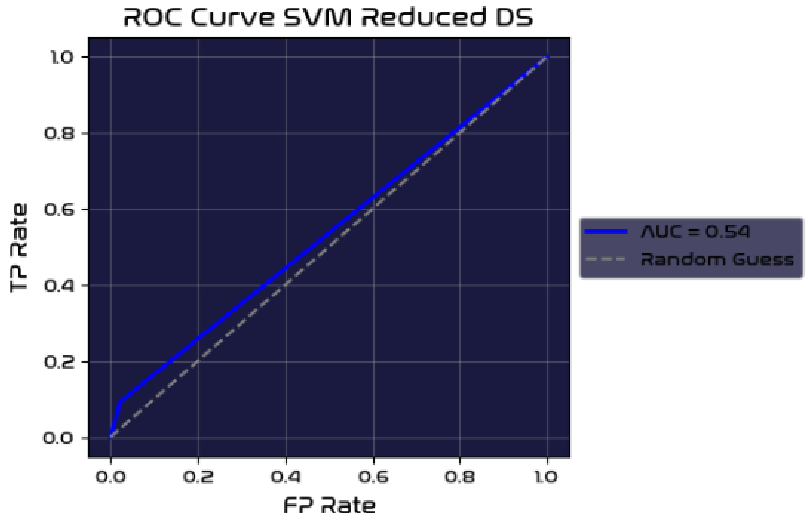






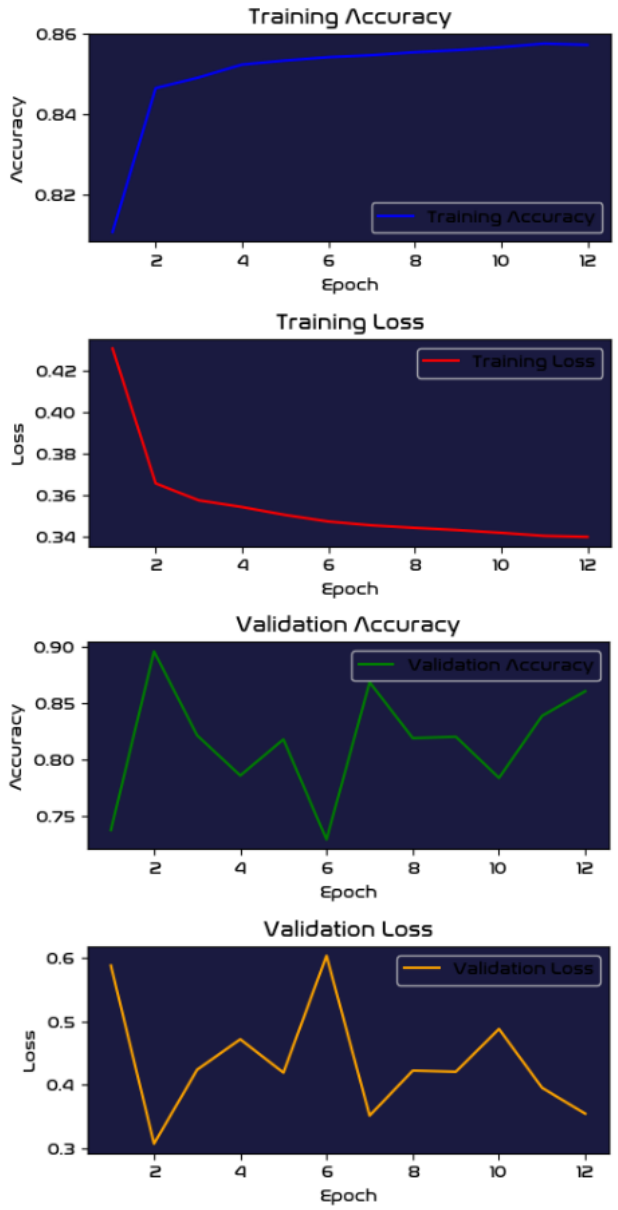




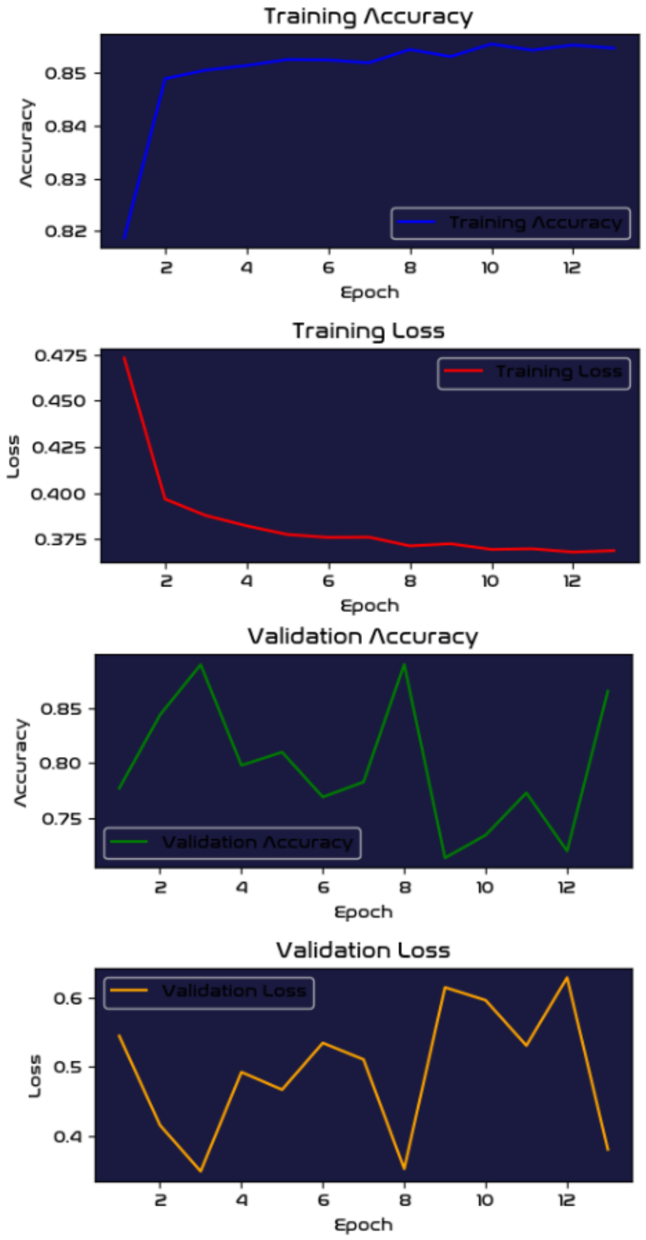


EXTRA: NEURAL NETWORK PERFORMANCE

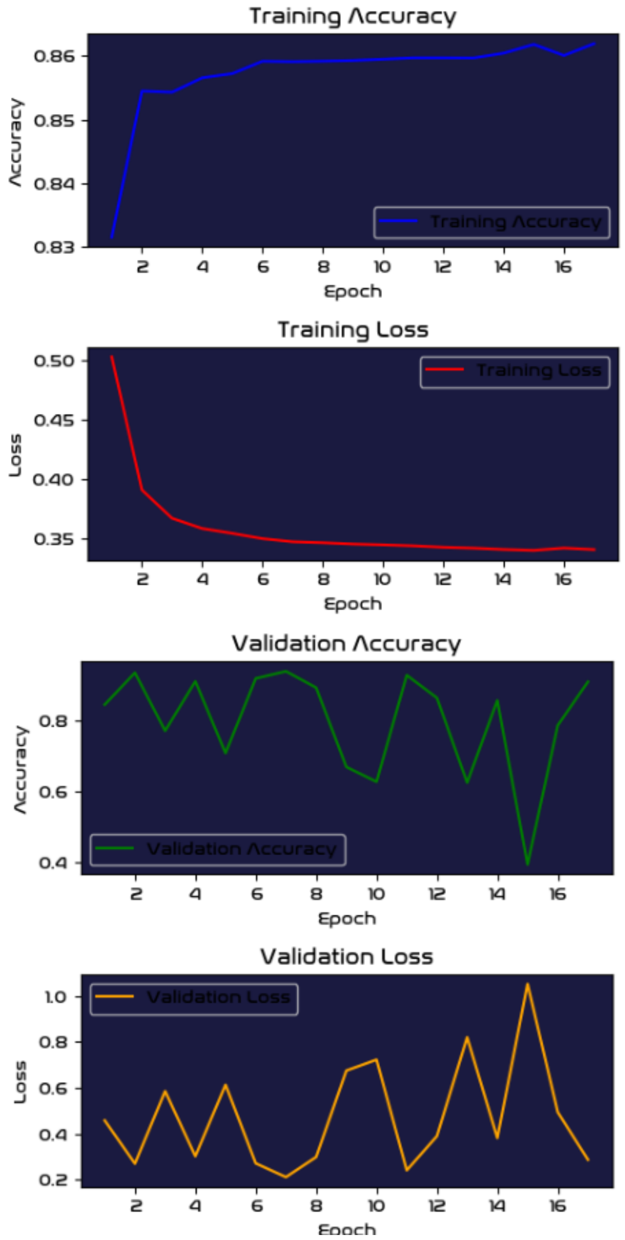
MODEL 1



MODEL 2



MODEL 3



Conclusion

Random Forest (RF):

- **Entire Dataset:** Highest accuracy (87%) and robust f1-scores across both classes.
- **Reduced Dataset:** Significant drop in performance (77%), particularly for class 1.

Neural Network (NN):

- **Model 1 & 2:** Similar performance (accuracy ~85%), but class 1 f1-scores remain lower.
- **Model 3:** Slight improvement with accuracy ~86%, strongest recall and f1 for class 0.

Support Vector Machine (SVM):

- **Entire Dataset:** Comparable performance to RF, but slightly lower accuracy (87%) and f1-score for class 0.
- **Reduced Dataset:** Largest performance drop, accuracy drops to 76%.

Key takeaways:

- **Random Forest** performs best overall on the full dataset.
- **Neural Networks** are stable but underperform for class 1.
- **SVM** is sensitive to dataset reduction.

REFERENCES

[1] C. Bryan, 'SDSS Galaxy Classification DR18'. 2024. Accessed: Dec. 03, 2024. [Online]. Available: https://www.kaggle.com/datasets/bryancimo/sdss-galaxy-classification-dr18

[2] C. C. Hayward, P. Jonsson, D. Kereš, B. Magnelli, L. Hernquist, and T. J. Cox, 'How to distinguish starbursts and quiescently star-forming galaxies: the "bimodal" submillimetre galaxy population as a case study: Star formation modes and the SMG bimodality', Monthly Notices of the Royal Astronomical Society, vol. 424, no. 2, pp. 951–970, Aug. 2012, doi: 10.1111/j.1365-2966.2012.21254.x.

[3] chaitu_e6, 'Random Forest vs Support Vector Machine vs Neural Network', GeeksforGeeks. Accessed: Dec. 17, 2024. [Online]. Available: https://www.geeksforgeeks.org/random-forest-vs-support-vector-machine-vs-neural-network/

[4] Emma Ding, Handling Imbalanced Dataset in Machine Learning: Easy Explanation for Data Science Interviews, (Dec. 05, 2022). Accessed: Dec. 18, 2024. [Online Video]. Available: https://www.youtube.com/watch?v=GR-OW5asKlk

[5] P. Probst, M. N. Wright, and A. Boulesteix, "Hyperparameters and tuning strategies for random forest," Wiley Interdiscip. Rev.: Data Min. Knowl. Discov., vol. 9, 2018.

[6] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," arXiv, abs/1502.03167, 2015.

[7] C. Kishor, K. Reddy, I. VijayaSindhooriKaza, P. R. Anisha, I. MousaMohammedKhubrani, M. Shuaib, I. ShadabAlam, and S. Ahmad, "Optimising barrier placement for intrusion detection and prevention in WSNs," PLOS ONE, vol. 19, 2024.