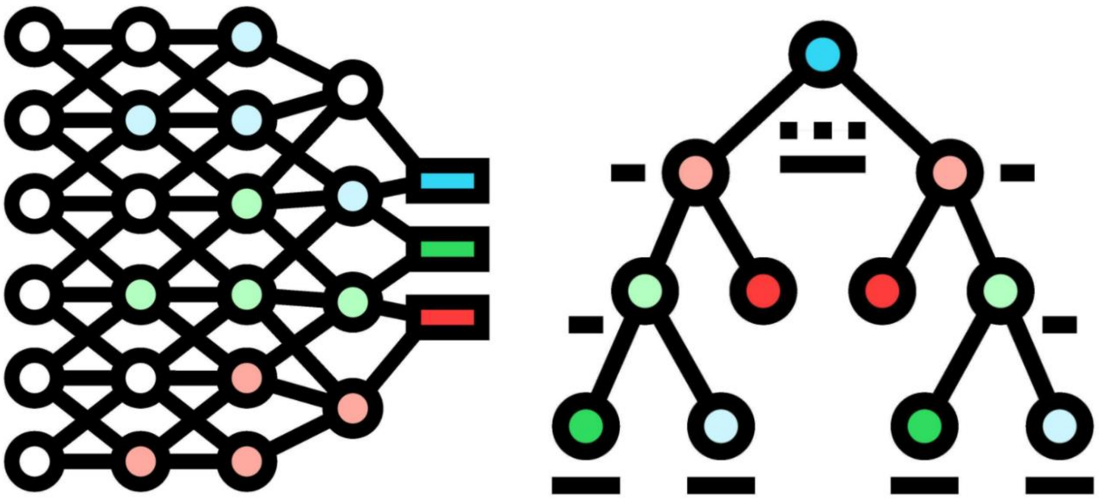
Deep Learning for Higgs Boson Classification: A Comparative Study with Ensemble Models



JUNAID

Data Science & A.I.

Cohort 11

atom**camp**

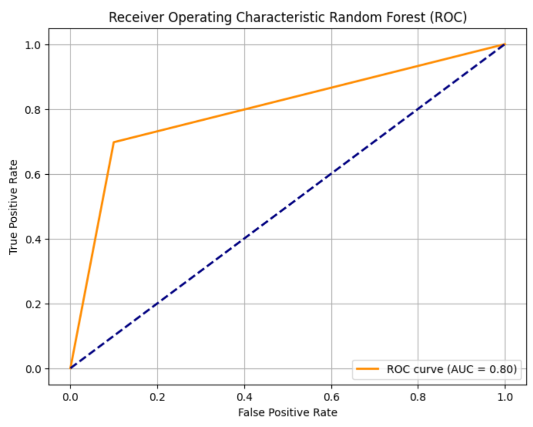
Higgs Boson Classification Model Performance Report

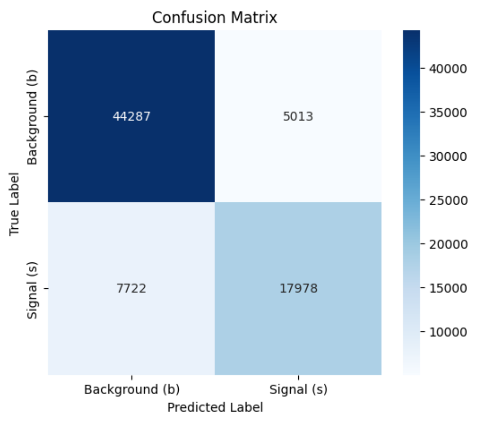
This report analyzes the performance of various machine learning and deep learning models on the Higgs Boson dataset, focusing on classification tasks. The primary goal is to identify the presence of a "Signal (s)" from "Background (b)" events. We will compare ensemble methods (Random Forest and XGBoost) with different configurations of Multi-Layer Perceptrons (MLPs), examining factors such as model depth, activation functions, regularization techniques, optimizers, and training callbacks.

1. Model Performance Overview

Random Forest Classifier

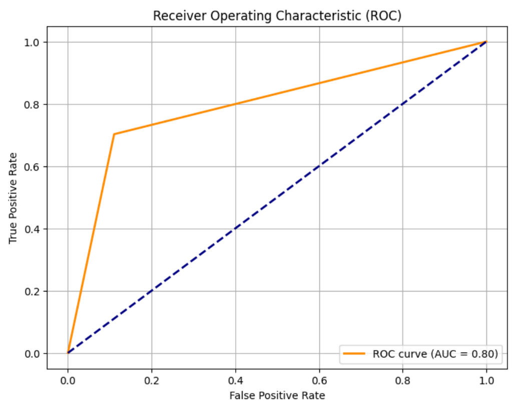
Metric	**Background (b)**	**Signal (s)**
Precision	0.85	0.78
Recall	0.90	0.70
F1-Score	0.87	0.74
Support	49300	25700
Accuracy	**0.83**	
Macro Avg	0.82 (P), 0.80 (R), 0.81 (F1)	
Weighted Avg	0.83 (P), 0.83 (R), 0.83 (F1)	

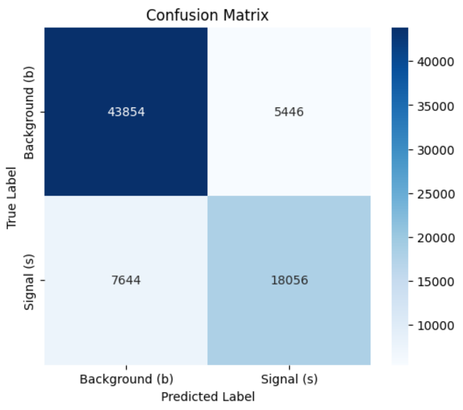




XGBoost

Metric	**Background (b)**	**Signal (s)**
Precision	0.85	0.77
Recall	0.89	0.70
F1-Score	0.87	0.73
Support	49300	25700
Accuracy	**0.83**	
Macro Avg	0.81 (P), 0.80 (R), 0.80 (F1)	
Weighted Avg	0.82 (P), 0.83 (R), 0.82 (F1)	



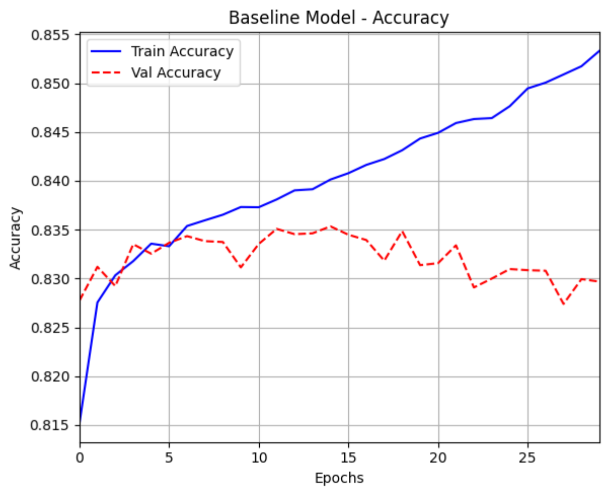


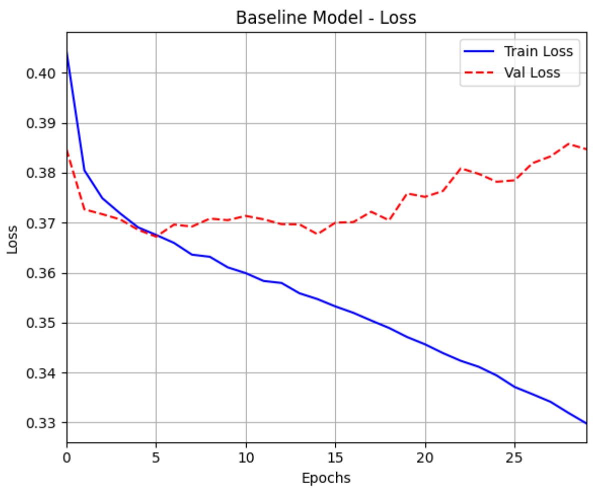
Initial MLP (dl_model)

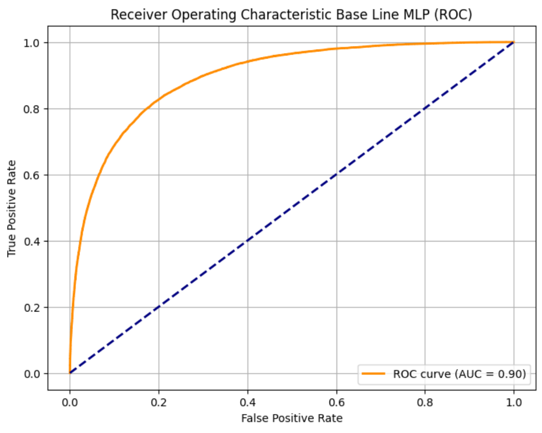
Model Architecture:

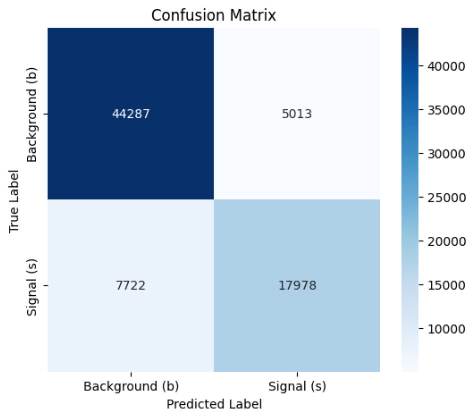
```
dl_model = Sequential([
  Input(shape=(X_train.shape[1],)),
  Dense(256, activation='relu'),
  Dense(128, activation='relu'),
  Dense(64, activation='relu'),
  Dense(1, activation='sigmoid')
])
```

Metric	**Background (b)**	**Signal (s)**
Precision	0.85	0.77
Recall	0.89	0.70
F1-Score	0.87	0.73
Support	49300	25700
Accuracy	**0.83**	
Macro Avg	0.81 (P), 0.80 (R), 0.80 (F1)	
Weighted Avg	0.82 (P), 0.83 (R), 0.82 (F1)	





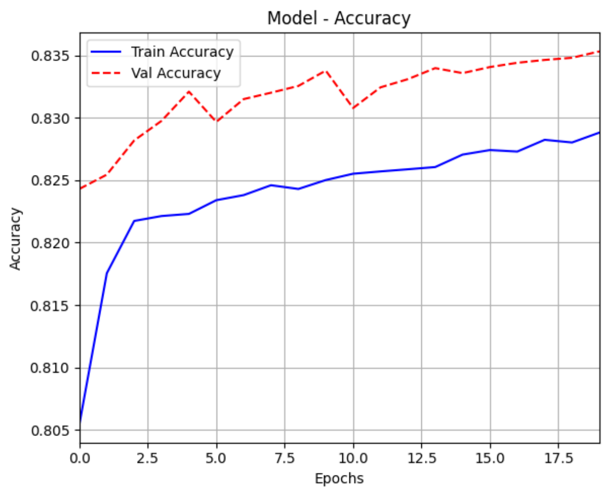


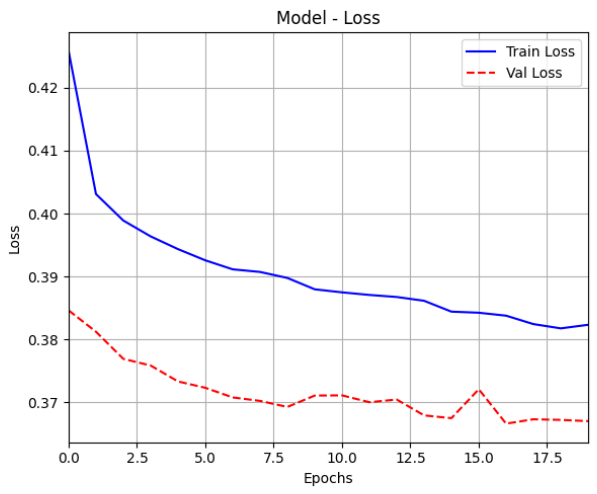


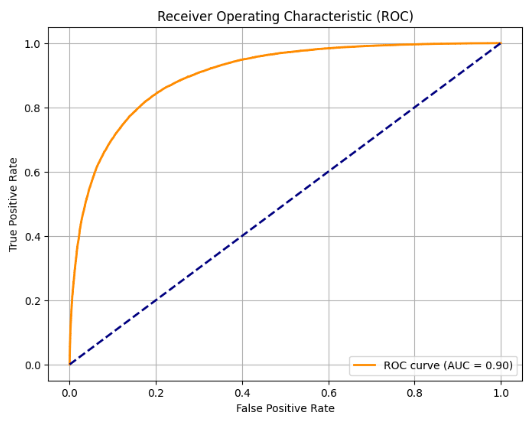
Regularized MLP (model_2)

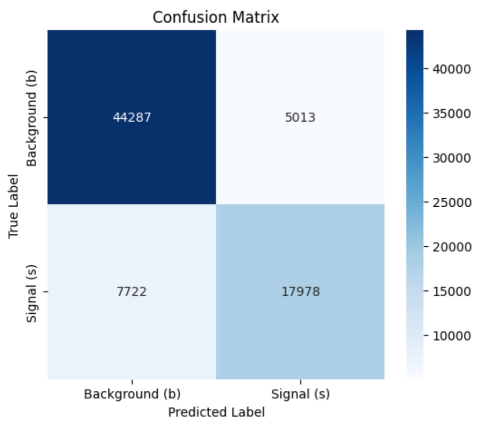
Model Architecture includes BatchNormalization and Dropout layers after each Dense layer (except the output).

Metric	**Background (b)**	**Signal (s)**
Precision	0.85	0.77
Recall	0.89	0.70
F1-Score	0.87	0.73
Support	49300	25700
Accuracy	**0.83**	
Macro Avg	0.81 (P), 0.80 (R), 0.80 (F1)	
Weighted Avg	0.82 (P), 0.83 (R), 0.82 (F1)	





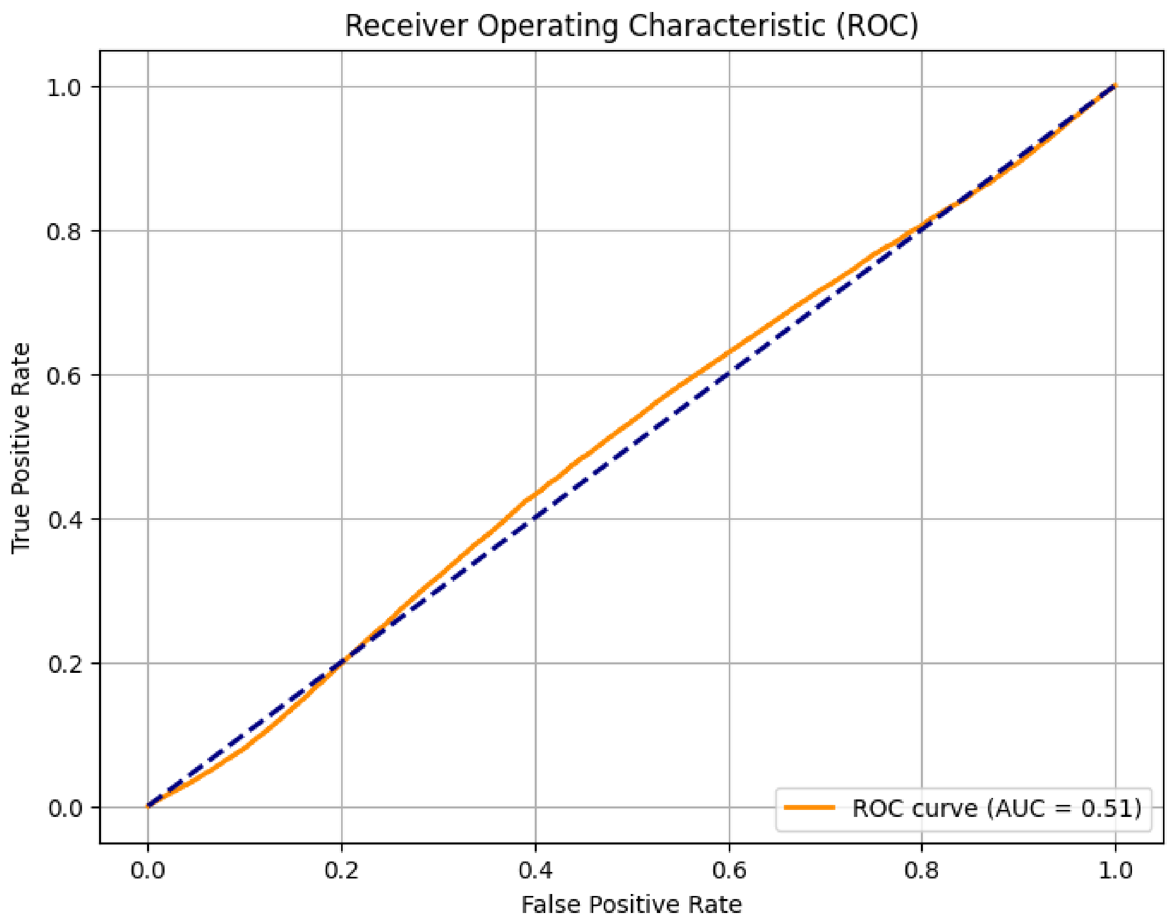


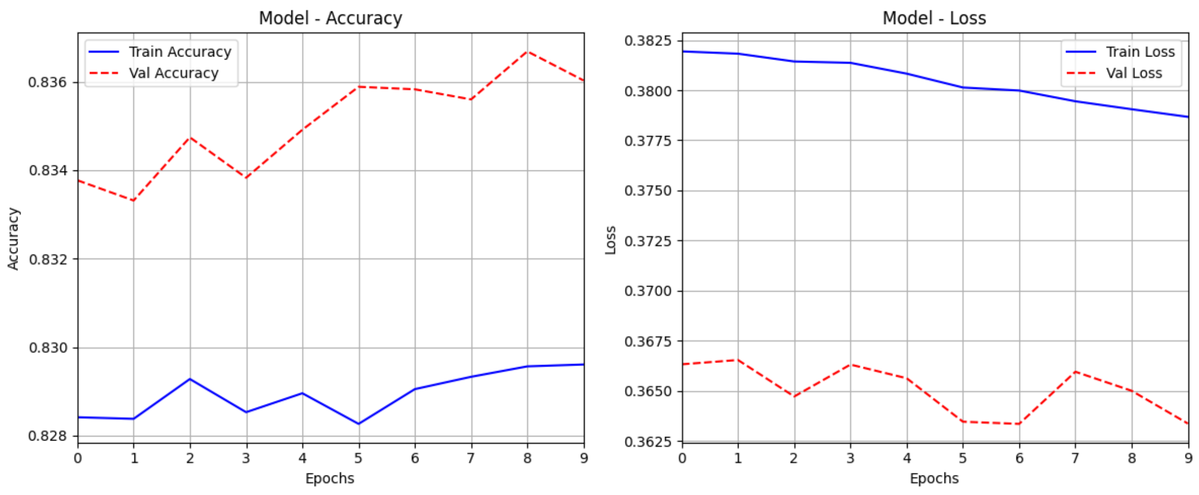


Regularized MLP with RMSprop (model_2 variant)

This model, using RMSprop for 10 epochs, showed poor convergence.

- Test accuracy: 64.45%
- Test loss: 68.17%

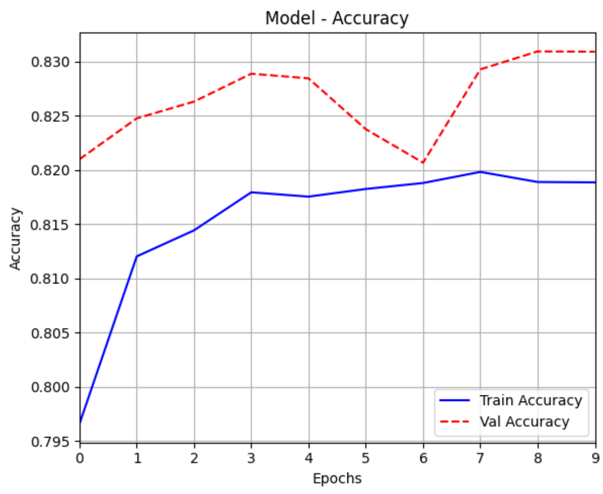


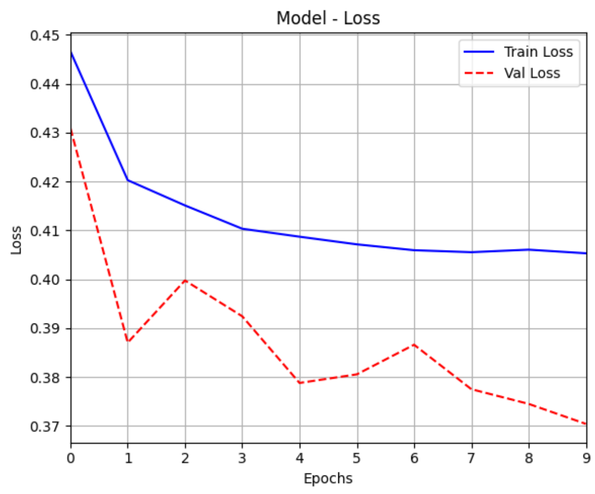


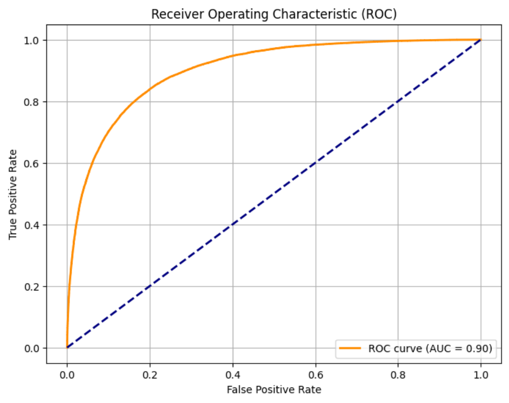
Regularized MLP with Callbacks (model_4)

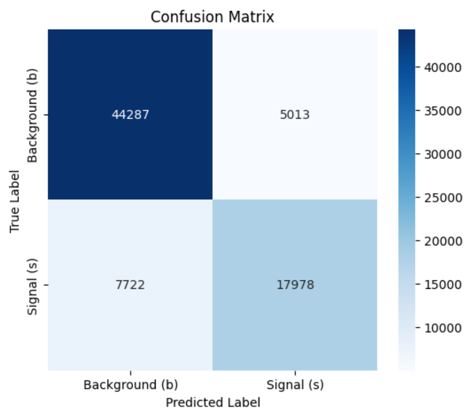
Model Architecture identical to model_2, but trained with ReduceLROnPlateau, ModelCheckpoint, and EarlyStopping callbacks.

Metric	**Background (b)**	**Signal (s)**
Precision	0.85	0.78
Recall	0.90	0.70
F1-Score	0.87	0.74
Support	49300	25700
Accuracy	**0.83**	
Macro Avg	0.82 (P), 0.80 (R), 0.81 (F1)	
Weighted Avg	0.83 (P), 0.83 (R), 0.83 (F1)	









2. Model Performance Comparison and Analysis

Observing the classification reports, it's evident that the Random Forest Classifier and the final Regularized MLP with Callbacks (model_4) achieved the best and very similar performance across most metrics, both reaching an overall accuracy of 83%. XGBoost also performed comparably but showed slightly lower precision and F1-score for the "Signal" class.

The **"Signal (s)" class** consistently proves harder to predict than "Background (b)," as indicated by lower precision, recall, and F1-scores across all successful models (e.g., Signal F1-score of 0.73-0.74 compared to Background F1-score of 0.87). This is expected in highly imbalanced datasets where the minority class (Signal) is more challenging to isolate.

How did model depth and activation affect performance?

Deeper models (model_2, model_4 with 5 hidden layers) with ReLU activations could capture more complex patterns. The initial shallower MLP (dl_model) overfitted, suggesting depth alone isn't enough; regularization is crucial for deep models to generalize.

What helped mitigate overfitting?

Batch Normalization stabilized training, and **Dropout** layers (randomly deactivating neurons) prevented co-adaptation. **Early Stopping** (in model_4) further prevented overfitting by halting training when validation performance ceased improving.

How did the learning rate and optimizer affect convergence?

RMSprop (on model_2 variant) led to poor convergence (64.45% accuracy). The adaptive learning rate from **ReduceLROnPlateau callback** (in model_4), which adjusts the learning rate during training, was crucial for better convergence, allowing model_4 to achieve optimal performance.

What would you improve with more time or compute?

With more resources, I would focus on:

1. **Extensive Hyperparameter Tuning** for all models.
2. **Advanced Ensemble Techniques** (e.g., stacking) to combine model strengths.
3. **Feature Engineering and Selection** to create more discriminative features.
4. **Cross-Validation** for more robust performance estimates.
5. **Addressing Class Imbalance** to improve Signal class prediction.

Conclusion

The analysis demonstrates that both traditional ensemble methods (Random Forest, XGBoost) and a well-regularized Multi-Layer Perceptron can achieve strong and comparable performance on the Higgs Boson classification task, with an accuracy of approximately 83%. The "Signal" class remains the more challenging one to predict.

The key takeaways from the MLP experiments are:

- **Overfitting is a significant challenge** in deep learning models, especially without proper regularization.
- **Batch Normalization and Dropout are highly effective** in mitigating overfitting and stabilizing training in deep neural networks.
- **Adaptive learning rate schedulers** like ReduceLROnPlateau are crucial for efficient and robust convergence, preventing models from getting stuck or diverging, as observed with the unoptimized RMSprop run.
- **Callbacks like Early Stopping and Model Checkpointing** are indispensable tools for managing the training process, ensuring that the best model weights are saved and training is halted when generalization performance no longer improves.

With further computational resources, extensive hyperparameter tuning, advanced ensemble strategies, and more rigorous cross-validation would likely yield marginal but valuable improvements in model robustness and potentially performance, particularly for the critical "Signal" class.