Objective:

This Python script implements a comprehensive approach to detect phishing websites using various machine learning models. The project aims to compare the performance of different algorithms in classifying URLs as either legitimate or phishing.

Data Preprocessing:

1. The dataset is loaded from a CSV file containing features extracted from URLs.
2. The 'Domain' column is dropped as it's not used for model training.
3. The data is randomly shuffled to ensure even distribution of classes.
4. The dataset is split into features (X) and labels (y), then further divided into training and testing sets with an 80-20 split.

Machine Learning Models:

The script implements and evaluates six different machine learning models:

1. Decision Tree Classifier:
    - Implemented with a max depth of 5 to prevent overfitting.
    - Achieves high accuracy on both training (0.991) and test (0.998) sets.
2. Random Forest Classifier:
    - Also implemented with a max depth of 5.
    - Performs exceptionally well with the highest test accuracy (0.999) among all models.
3. Multilayer Perceptrons (MLP):
    - Uses three hidden layers of 100 neurons each.
    - Matches the Random Forest's performance with 0.999 test accuracy.
4. XGBoost Classifier:
    - Configured with a learning rate of 0.4 and max depth of 7.
    - Shows strong performance with 0.998 test accuracy.
5. Autoencoder Neural Network:
    - Implemented as an unsupervised learning model for feature extraction.
    - Underperforms compared to other models with 0.359 test accuracy.
6. Support Vector Machine (SVM):
    - Uses a linear kernel with C=1.0.
    - Achieves high accuracy (0.998) on par with Decision Tree and XGBoost.

Results and Analysis:

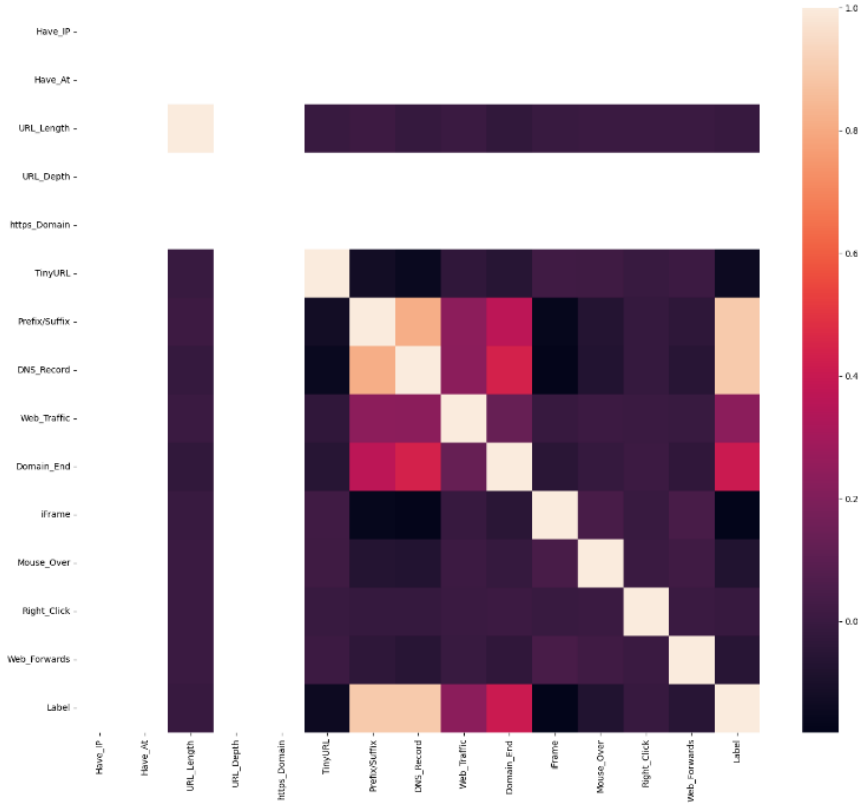
1. Model Performance:

    - Random Forest and MLP show the best performance with 0.999 test accuracy.
    - Decision Tree, SVM, and XGBoost follow closely with 0.998 test accuracy.
    - The Autoencoder significantly underperforms with 0.359 test accuracy.
2. Model Comparison:
    - The results table shows that all models except the Autoencoder perform exceptionally well.
    - The Random Forest model is chosen as the best performer due to its highest test accuracy.
3. Feature Importance:
    - Visualizations for feature importance are generated for Decision Tree and Random Forest models.
    - These plots provide insights into which features are most crucial for classification.
4. Data Distribution and Correlation:
    - Histograms of the dataset features are plotted to understand data distribution.
    - A correlation heatmap is generated to visualize relationships between features.
5. Model Saving and Loading:
    - The XGBoost model is saved to a file and then loaded to demonstrate persistence.

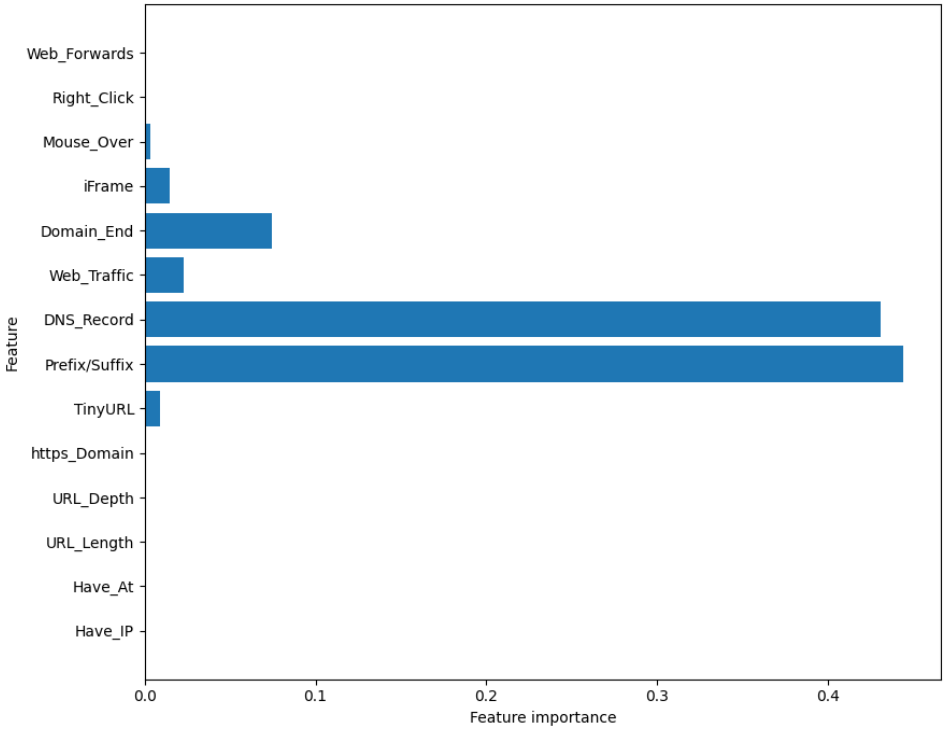
Output

	ML Model	Train Accuracy	Test Accuracy	Loss	Val_loss
1	Random Forest	0.991	0.999	0.3391	0.2384
2	Multilayer Perceptrons	0.991	0.999	0.2217	0.1040
0	Decision Tree	0.991	0.998	0.1723	0.1506
5	SVM	0.991	0.998	0.1535	0.1362
3	XGBoost	0.990	0.998	0.1252	0.1230
4	AutoEncoder	0.364	0.359	0.1098	0.1134

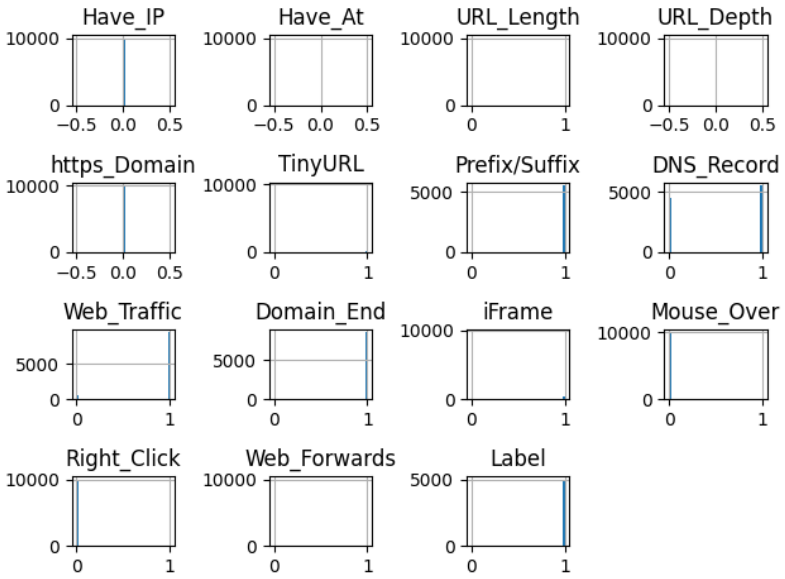
correlation_heatmap



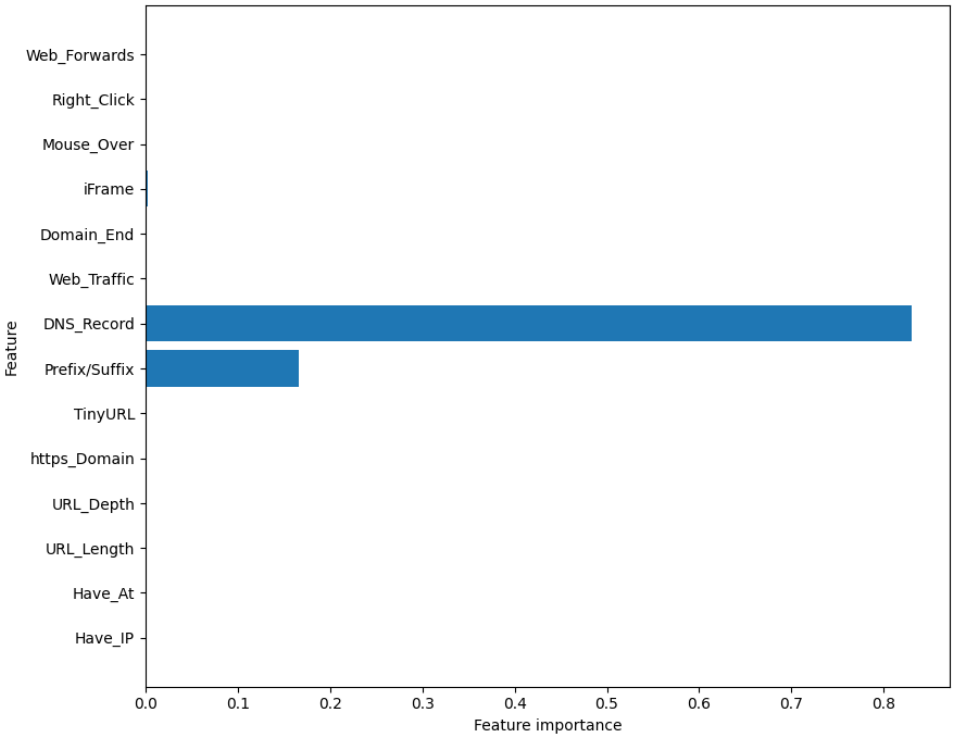
random_forest_feature_importance



data_distribution



decision_tree_feature_importance



Conclusion:

The script successfully implements and compares multiple machine learning models for phishing URL detection. Most models show excellent performance, with Random Forest and MLP achieving the highest accuracy. The Autoencoder's poor

performance suggests it might not be suitable for this specific classification task without further optimization.

The generated visualizations provide valuable insights into the dataset and model behaviour. The saved model can be used for future predictions on new, unseen data.