Course Project: Malicious URL Detection

Objective: Develop machine learning models to identify malicious URLs. This project will involve data preprocessing, model training using **3-4 different methods (each student is required to train at least one model)**, and a comparative analysis of their performance.

Dataset: You may choose **one or multiple datasets** listed below

1. Kaggle Malicious URLs Dataset
   https://www.kaggle.com/datasets/sid321axn/malicious-urls-dataset
2. Phishing URL Dataset
   https://data.mendeley.com/datasets/vfszbj9b36/1
3. Kaggle Malicious and Benign URLs
   https://www.kaggle.com/datasets/siddharthkumar25/malicious-and-benign-urls

Programming Language: Python3

Project Requirements

1. **Data Preprocessing**
   Address at least two issues of the dataset, including *but not limited to*:
   - Handling Missing Values: Identify and appropriately address any missing or inconsistent data entries.
   - Remove Duplicate Records: Identify and appropriately remove duplicate data entries.
   - Feature Extraction/Encoding (very important, read carefully): Extract structured and lexical features only, no content fetching. Examples (not an exhaustive list!!!):
     - Feature-agnostic processing
       - Lowercasing: Lowercase protocol and domain name
       - Fragment dropping: Drop `#fragment`
       - Default port stripping: `http://a.com:80/x` → `http://a.com/x`
     - Example Features:
       - Lengths: total URL length, host length, path length, query length.
       - Counts: number of subdomains, number of digits, number of "special" symbols (e.g., `-`, `_`, `@`, `?` , `%`, `=`, …)
       - Ratios: digit-to-length, symbol-to-length, uppercase-to-length.
       - Entropy: Shannon entropy
       - Here's a great resource to look into and see what the structured and lexical features you may want to extract: https://cyberlab.usask.ca/papers/Mamun2016_Chapter_DetectingMaliciousURLsUsingLex.pdf
   - Categorical Encoding: Convert categorical features (e.g., protocol type – http or https) into numerical representations.
   - Feature Scaling: Normalize or standardize numerical features

- Handling Imbalanced Data: Apply techniques such as oversampling, undersampling, or synthetic data generation to address class imbalance.

Note1: It is recommended to address as many of these issues as possible to ensure a clean dataset for your models.

Note2: Here's a blog you can read to better understand URL structures - https://medium.com/@joseph.pyram/9-parts-of-a-url-that-you-should-know-89fea8e11713

2. **Model Development**

2.1 Model Selection: Each member in the group should implement at least one model. Depending on your group size, you should have at least 3 – 4 different models.

Examples (not an exhaustive list!!!) including *but not limited to*:

- Supervised Learning:
  - Logistic Regression
  - Random Forests
  - Support Vector Machines (SVM)
  - KNN
  - Neural Networks (e.g., Multi-Layer Perceptron)
  - …
- Unsupervised Learning:
  - One-Class Support Vector Machines
  - K-Means Clustering
  - Isolation Forests
  - …

2.2 Hyperparameter Tuning: Optimize model parameters using cross-validation or other appropriate methods to enhance performance.

3. **Model Evaluation:**

3.1 Performance Metrics: Assess models using metrics such as Accuracy (overall and per-class), Precision, Recall, F1-Score, Runtime Analysis, etc. Explain your choice of metrics.

3.2 Comparative Analysis: Compare the performance of the selected models, discussing their strengths and weaknesses in the context of malicious URL detection.

4. **Individual Contributions:** Clearly list each team member's contributions at the end of the report. This should include specific tasks undertaken. Each member should engage in both coding and report writing. Collaboration is essential; discuss each model's pros and cons and reach a consensus on the final models to implement.

Project Proposal Guidelines (and Rubric)

Your project proposal should be structured as follows:

1. **Title Section:**

- Project title
- Team members' names

2. **Abstract (10%):**
    - A concise summary of the project objectives, methods
3. **Introduction (10%):**
    - Background information on Malicious URL detection, including URL structure, reasoning behind feature extraction, etc
    - Overview of your choice of datasets
    - Clear statement of project objectives
4. **(Optional) Related Work:**
    - Summary of existing research and methodologies in lexical malicious URL detection
    - Discussion on how your approach aligns with or diverges from previous studies
5. **Methodology (60%):**
    - Plans and justification for data preprocessing (20%)
    - Plans and justification for model selection (30%)
    - Plans for evaluation metrics (10%)
6. **Task division (20%):**
    - Provide a plan detailing each team member's responsibilities

Note: Plans and individual tasks can be adjusted as the project progresses. While the proposal will be graded solely on its content, please be aware that the final report will also be evaluated based on its clarity and quality (See "Final Report Rubric" section below).

Final Report Guidelines

Your final report should be structured as follows:

1. **Title Section:**
    - Project title
    - Team members' names
2. **Abstract:**
    - A concise summary of the project objectives, methods, results, and conclusions.
3. **Introduction:**
    - Background information on Malicious URL detection, including URL structure, reasoning behind feature extraction, etc
    - Overview of your choice of datasets
    - Clear statement of project objectives
4. **(Optional) Related Work:**
    - Summary of existing research and methodologies in lexical malicious URL detection.
    - Discussion on how your approach aligns with or diverges from previous studies.
5. **Methodology:**
    - Detailed description of data preprocessing steps.
    - Explanation of the selected models and justification for their choice.
    - Outline of the training process, including hyperparameter tuning.
6. **Results:**
    - Presentation of evaluation metrics for each model.

- Visualizations (e.g., confusion matrices, ROC curves) to illustrate performance.

7. **Discussion:**
   - Interpretation of results, highlighting key findings.
   - Comparative analysis of model performances.
   - Discussion of potential limitations and challenges encountered.
8. **Conclusion:**
   - Summary of the project's outcomes.
   - Suggestions for future work or improvements.
9. **Individual Contributions:**
   - Detailed account of each team member's specific contributions to the project.
10. **References:**
    - List of all sources and literature referenced throughout the report.

Final Report Rubric

Your project will be evaluated based on the following criteria:

1. **Introduction and Background (5%):**
   - Clarity and relevance of the introduction
   - Adequate background information provided
2. **Data Preprocessing (20%):**
   - Appropriateness and thoroughness of preprocessing methods (15%)
   - Justification for chosen techniques (5%)
3. **Model Implementation (30%):**
   - Correctness of model implementation (25%)
   - Justification for model selection (5%)
4. **Evaluation and Analysis (20%):**
   - Use of appropriate evaluation metrics (5%)
   - Depth of comparative analysis between models (15%)
5. **Report Quality (15%):**
   - Organization and clarity of the report (10%)
     E.g., Use sections, headings, captions for tables and figures
   - Proper use of figures, tables, and references (5%)
6. **Individual Contributions (10%):**
   - Individual contribution of each member

Submission Guidelines

- **Report Length:** The final report should not exceed 13 pages, excluding references and appendices.
- **Formatting:** Use a standard font (e.g., Times New Roman, 11/12 pt) with 1-inch margins. Consider using the Latex template provided (NeurIPS.zip).
- **File Format:**
  - Project Proposal: Submit the project proposal as a pdf document.
  - Project Final Report: you will turn in your project by uploading one single .zip file. The zip file should include the following:

        - a written final report (as a pdf file)
        - a zip file that contains all your code files (e.g., py or ipynb files) in a form that can be run to verify your results
        - a written README documentation file that explains instructions to run your code and the environment your code is being executed in for your report.
- **Deadline:**
    - Project Proposal: Oct 31, 2025 11:59pm
    - Project Final Report: Dec 12, 2025 11:59pm
    - No late submissions will be accepted.

Note: Each team member may submit the final project individually (must be the exact same file; if multiple versions are submitted, only one will be graded) or designate one member to submit. If choosing the latter, make sure the submission is complete by the deadline, as late submissions are not allowed.

Additional Notes

- Make sure that all code is well-documented and reproducible.
- Any external sources (i.e., the use of Generative AI) or libraries used must be properly cited.
- Adhere to academic integrity guidelines; plagiarism will not be tolerated.