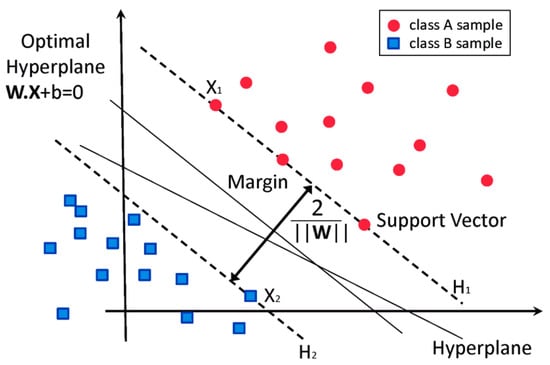
I/ Khái niệm về SVM:

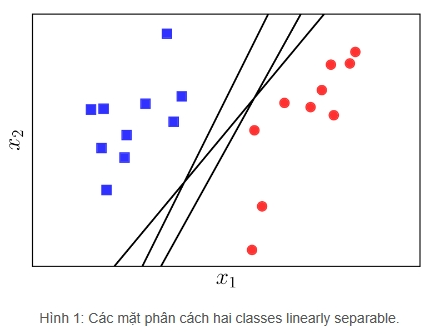
1. **Support Vector Machines (SVM)**:
   * SVM là một thuật toán học có giám sát được sử dụng cho phân loại và hồi quy.
   * Ý tưởng cơ bản của SVM là tìm một siêu phẳng (hyperplane) trong không gian nhiều chiều sao cho nó tách hai lớp dữ liệu (ví dụ: hai lớp “đúng” và “sai”) một cách tốt nhất.
   * Siêu phẳng này được chọn sao cho khoảng cách từ các điểm dữ liệu gần nhất đến siêu phẳng là lớn nhất. Những điểm dữ liệu này được gọi là “vector hỗ trợ” (support vectors).
   * SVM là một thuật toán tuyến tính trong không gian có số chiều cao, nhưng nó có thể tạo ra các ranh giới phức tạp hơn bằng cách sử dụng các hàm kernel (ví dụ: kernel Gaussian) để ánh xạ dữ liệu vào không gian cao chiều.
   * SVM có khả năng tổng quát hóa tốt và ít bị overfitting.
2. **Tối ưu hóa lồi và hàm mục tiêu**:
   * SVM tìm siêu phẳng tốt nhất bằng cách giải quyết một bài toán tối ưu lồi.
   * Bài toán tối ưu lồi này liên quan đến việc tối thiểu hóa một hàm bậc hai (thường là hàm mất mát) dưới ràng buộc tuyến tính.
   * Ràng buộc tuyến tính đảm bảo rằng các điểm dữ liệu nằm đúng phía của siêu phẳng.
   * Kết quả của bài toán tối ưu là siêu phẳng tốt nhất và các vector hỗ trợ.
3. **Ứng dụng của SVM**:
   * SVM được sử dụng rộng rãi trong phân loại văn bản, nhận dạng hình ảnh, phát hiện tín hiệu, và nhiều lĩnh vực khác.
   * Đặc biệt, SVM thường hoạt động tốt khi số lượng chiều dữ liệu lớn và khi dữ liệu không tách hoàn toàn tuyến tính.



**Hình 3.** Support vector machine

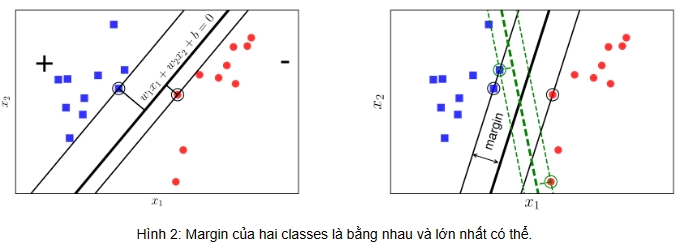
Như được thể hiện trong Hình 3, SVM dự đoán nhãn bằng cách tạo ra ranh giới quyết định, chẳng hạn như một siêu phẳng, giữa hai lớp được chỉ định với ít nhất một nhãn, các điểm dữ liệu và vector hỗ trợ được xử lý bởi siêu phẳng, nó tận dụng khoảng cách giữa các điểm dữ liệu để phân loại từng lớp một cách độc lập. (As shown in Figure 3, SVM predicts labels by creating decision boundaries, such as a hyperplane, between two designated classes with at least one label, data points, and support vectors processed by the hyperplane. It leverages the distance between data points to classify each class independently.)

Ví dụ 1: Bắt đầu với bài toán trong Perceptron Learning Algorithm (PLA). Giả sử rằng có hai class khác nhau được mô tả bởi các điểm trong không gian nhiều chiều, hai classes này linearly separable, tức tồn tại một siêu phẳng phân chia chính xác hai classes đó. Hãy tìm một siêu mặt phẳng phân chia hai classes đó, tức tất cả các điểm thuộc một class nằm về cùng một phía của siêu mặt phẳng đó và ngược phía với toàn bộ các điểm thuộc class còn lại. Chúng ta đã biết rằng, thuật toán PLA có thể làm được việc này nhưng nó có thể cho chúng ta vô số nghiệm như Hình 1 dưới đây: (Example 1: Start with the problem in the Perceptron Learning Algorithm (PLA). Assume there are two different classes described by points in a multidimensional space, and these classes are linearly separable, meaning there is a hyperplane that accurately divides the two classes. Find a hyperplane that divides the two classes, meaning all points of one class are on the same side of the hyperplane and opposite to all points of the other class. We know that the PLA algorithm can do this, but it can give us an infinite number of solutions as shown in Figure 1 below:)



Vậy câu hỏi đặt ra ở đây là trong vô số các mặt phân chia đó, đâu là mặt phân chia tốt nhất theo một tiêu chuẩn nào đó? Trong ba đường thẳng minh họa trong Hình 1 phía trên, có hai đường thẳng khá lệch về phía class hình tròn đỏ. Điều này có thể khiến cho class màu đỏ không hài lòng lắm vì lãnh thổ xem ra bị lấn nhiều quá. Liệu có cách nào để tìm được đường phân chia mà cả hai classes đều cảm thấy công bằng và hạnh phúc nhất hay không? (So the question here is, among the infinite number of dividing surfaces, which is the best one according to some standard? In the three illustrated lines in Figure 1 above, two lines are quite skewed towards the red circle class. This may make the red class somewhat unhappy because their territory seems to be encroached upon too much. Is there a way to find a dividing line that both classes feel is the fairest and happiest?)

* Chúng ta cần tìm một tiêu chuẩn để đo sự hạnh phúc của mỗi class. Hãy xem Hình 2 dưới đây: (-> We need to find a criterion to measure the happiness of each class. Let’s look at Figure 2 below:)



Nếu ta định nghĩa mức độ hạnh phúc của một class tỉ lệ thuận với khoảng cách gần nhất từ một điểm của class đó tới đường phân chia, thì ở Hình 2 trái, class tròn đỏ sẽ không được hạnh phúc cho lắm vì đường phân chia gần nó hơn class vuông xanh rất nhiều. Chúng ta cần một đường phân chia sao cho khoảng cách từ điểm gần nhất của mỗi class (các điểm được khoanh tròn) tới đường phân chia là như nhau, như thế thì mới công bằng. Khoảng cách như nhau này được gọi là margin (lề). (If we define the happiness level of a class as being directly proportional to the nearest distance from a point of that class to the dividing line, then in Figure 2 on the left, the red circle class will not be very happy because the dividing line is much closer to it than the blue square class. We need a dividing line so that the distance from the nearest point of each class (the circled points) to the dividing line is the same, only then it is fair. This equal distance is called the margin.)

So, Đã có công bằng rồi thì chúng ta cần văn minh nữa. Công bằng mà cả hai đều kém hạnh phúc như nhau thì chưa phải là văn mình cho lắm. (In the pursuit of fairness, we also need to consider civility. If both parties are equally unhappy despite achieving fairness, it doesn’t quite embody civility.)

Chúng ta xét tiếp Hình 2 bên phải khi khoảng cách từ đường phân chia tới các điểm gần nhất của mỗi class là như nhau. Xét hai cách phân chia bởi đường nét liền màu đen và đường nét đứt màu lục, đường nào sẽ làm cho cả hai class hạnh phúc hơn? Rõ ràng đó phải là đường nét liền màu đen vì nó tạo ra một margin rộng hơn. -> **Việc margin rộng hơn sẽ mang lại hiệu ứng phân lớp tốt hơn vì sự phân chia giữa hai classes là rạch ròi hơn. Điều này khá quan trọng vì nó giúp Support Vector Machine mang lại kết quả phân loại tốt hơn so với Neural Network với 1 layer, tức là Perceptron Learning Algorithm.** (Now, let’s examine the right side of Figure 2, where the distance from the dividing line to the nearest points of each class is equal. Consider two dividing lines: the solid black line and the dashed green line. Which line would make both classes happier? Clearly, it’s the solid black line because it creates a wider margin. **-> The broader margin leads to better classification performance because the separation between the two classes is more distinct. This is crucial because it allows Support Vector Machines (SVMs) to achieve better classification results compared to a Neural Network with only one layer, namely the Perceptron Learning Algorithm.**)

* *Bài toán tối ưu trong Support Vector Machine (SVM) chính là bài toán đi tìm đường phân chia sao cho margin là lớn nhất. Đây cũng là lý do vì sao SVM còn được gọi là Maximum Margin Classifier. (=> The optimization problem in SVM aims to find the dividing line that maximizes the margin. This is why SVM is also known as the Maximum Margin Classifier.)*
* Ở đây chúng ta có đề cập đến thuật toán PLA thì mình xin được nhắc lại khái niệm cho mọi người nhớ là: (Let’s revisit the concepts related to the Perceptron Learning Algorithm (PLA):)

+ Perceptron(\*) là một trong những kiến trúc mạng nơ-ron nhân tạo đơn giản nhất vào những năm 1950. (The Perceptron is one of the simplest artificial neural network architectures, dating back to the 1950s.)

+ Một mạng nơ-ron feedforward(\*\*) đơn lớp, bao gồm một lớp đầu vào kết nối đầy đủ với một lớp đầu ra và nó có thể học các mẫu tách biệt tuyến tính. (It consists of a single-layer feedforward neural network, comprising an input layer fully connected to an output layer. The Perceptron can learn linearly separable patterns.)

+ Perceptron sử dụng threshold logic units – TLU để tính toán đầu ra và các trọng số của Perceptron được điều chỉnh trong quá trình huấn luyện để học các giá trị tối ưu. (The Perceptron uses threshold logic units (TLUs) to compute its output, and the weights of the Perceptron are adjusted during training to learn optimal values.)

+ Các thành phần cơ bản của Perceptron:

* Đặc trưng đầu vào: Perceptron nhận nhiều đặc trưng đầu vào, mỗi đặc trưng đại diện cho một thuộc tính của dữ liệu đầu vào.
* Trọng số: Mỗi đặc trưng được liên kết với một trọng số, quyết định tầm quan trọng của đặc trưng đó đối với đầu ra của Perceptron.
* Hàm tổng: Perceptron tính tổng có trọng số của các đặc trưng đầu vào.
* Hàm kích hoạt: Tổng trọng số được đưa qua hàm kích hoạt. Perceptron sử dụng hàm bước Heaviside(được ký hiệu bằng H hoặc θ) để đưa ra đầu ra (thường là 0 hoặc 1).
* Đầu ra: Kết quả cuối cùng của Perceptron, thường đại diện cho một lớp dự đoán (ví dụ: 0 cho đối số âm hoặc 1 cho đối số dương).

(+ Basic components of the Perceptron:

* Input features: The Perceptron receives multiple input features, with each feature representing an attribute of the input data.
* Weights: Each feature is associated with a weight that determines its importance for the Perceptron’s output.
* Sum function: The Perceptron computes the weighted sum of input features.
* Activation function: The total weighted sum is passed through an activation function. The Perceptron uses the Heaviside step function (denoted as H or θ) to produce the output (usually 0 or 1).
* Output: The final result of the Perceptron, typically representing a predicted class (e.g., 0 for negative input or 1 for positive input).)

+ Thuật toán học (Cập nhật trọng số):

* Trong quá trình huấn luyện, Perceptron học bằng cách điều chỉnh trọng số dựa trên thuật toán học.
* Trọng số được điều chỉnh để tìm ra siêu phẳng tốt nhất để phân tách các lớp dữ liệu.
* Các vector hỗ trợ (support vectors) chứa thông tin cần thiết để giải quyết vấn đề phân loại.

(+ Learning Algorithm (Weight Updates):

* During training, the Perceptron adjusts its weights based on a learning algorithm.
* The weights are updated to find the best hyperplane for separating the data classes.
* Support vectors contain essential information to address the classification problem.)

(\*): Perceptron là một loại nơ-ron nhân tạo đơn giản hoặc dạng cơ bản nhất của mạng nơ-ron. Nó được sử dụng cho các vấn đề phân loại nhị phân, có thể quyết định liệu một đầu vào được biểu diễn bằng một vector số liệu có thuộc về một lớp nào đó hay không.

(\*\*):

* **Đặc điểm của mạng nơ-ron truyền thẳng(FNN)**:
  + Mạng nơ-ron truyền thẳng không có chu trình hoặc vòng lặp.
  + Thông tin di chuyển chỉ theo một hướng từ các nút đầu vào, thông qua các nút ẩn (nếu có) và đến các nút đầu ra.
  + Mạng này thường bao gồm một lớp đầu vào, một hoặc nhiều lớp ẩn và một lớp đầu ra.
* **Ứng dụng của mạng nơ-ron truyền thẳng(FNN)**:
  + Mạng nơ-ron truyền thẳng được sử dụng rộng rãi trong phân loại, dự đoán và hồi quy.
  + Đặc biệt, nó thường hoạt động tốt khi số lượng chiều dữ liệu lớn và khi dữ liệu không tách hoàn toàn tuyến tính.

((\*): The Perceptron is a simple or basic form of an artificial neural network. It is used for binary classification problems, determining whether an input represented by a feature vector belongs to a specific class or not.

(\*\*):

* **Characteristics of Feedforward Neural Networks (FNNs):**
  + **FNNs do not have cycles or loops.**
  + **Information flows in only one direction, from input nodes through hidden nodes (if any) to output nodes.**
  + **An FNN typically consists of an input layer, one or more hidden layers, and an output layer.**
* **Applications of Feedforward Neural Networks (FNNs):**
  + **FNNs are widely used for classification, prediction, and regression tasks.**
  + **They perform well, especially when dealing with high-dimensional data and cases where data is not perfectly linearly separable.)**

II/ Ứng dụng của SVM trong việc phát hiện các trang web lừa đảo:

* + 1. Phân loại nhị phân:

+ SVM là một thuật toán phân loại nhị phân, giúp phân loại dữ liệu thành hai lớp khác nhau (ví dụ: lừa đảo và không lừa đảo).

+ Trong trường hợp phát hiện trang web lừa đảo, SVM có thể được sử dụng để xác định xem một trang web có khả năng là lừa đảo hay không dựa trên các đặc trưng(features) của nó.

* + 1. Học từ dữ liệu huấn luyện:

+ SVM học từ dữ liệu huấn luyện, trong đó các trang web đã được gán nhãn (lừa đảo hoặc không lừa đảo) và tìm ra siêu phẳng tốt nhất để phân tách hai lớp dữ liệu.

* + 1. Đặc trưng của trang web:

+ SVM sử dụng các đặc trưng(features) của trang web (ví dụ: đường dẫn, nội dung, thông báo) để xác định xem trang web có khả năng là lừa đảo hay không.

* + 1. Ứng dụng thực tế:

+ SVM đã được áp dụng rộng rãi trong việc phát hiện trang web lừa đảo, email spam, phân loại văn bản, và nhiều lĩnh vực khác.

* Các bước để phân loại các trang web là lừa đảo hoặc không lừa đảo dựa trên các đặc trưng của chúng:

+ Bước 1: Thu thập dữ liệu huấn luyện

* Chúng ta cần một tập dữ liệu gồm các trang web đã được gán nhãn là lừa đảo hoặc không lừa đảo.
* Mỗi trang web sẽ được biểu diễn bằng các đặc trưng như độ dài đường dẫn, số lượng liên kết, tỷ lệ chữ viết hoa, v.v.

+ Bước 2: Tiền xử lý dữ liệu

* Chúng ta sẽ chuẩn hóa và rút trích các đặc trưng từ URL của trang web.

+ Bước 3: Xây dựng mô hình SVM

* Chúng ta sẽ sử dụng SVM để tạo một mô hình phân loại.
* SVM tìm ra siêu phẳng tốt nhất để phân tách hai lớp dữ liệu (lừa đảo và không lừa đảo).

+ Bước 4: Huấn luyện mô hình

* Chúng ta sẽ huấn luyện mô hình SVM trên tập dữ liệu huấn luyện.

+ Bước 5: Dự đoán và đánh giá

* Chúng ta sẽ sử dụng mô hình đã huấn luyện để dự đoán xem một trang web mới có khả năng là lừa đảo hay không.
* Đánh giá hiệu suất của mô hình bằng các chỉ số như độ chính xác, độ nhạy, độ đặc hiệu, v.v.

(II/ **Applications of SVM in Detecting Fraudulent Websites**:

1. **Binary Classification**:
   * SVM is a binary classification algorithm that separates data into two different classes (e.g., fraudulent and non-fraudulent).
   * In the case of detecting fraudulent websites, SVM can be used to determine whether a website is likely fraudulent based on its features.
2. **Learning from Training Data**:
   * SVM learns from labeled training data, where websites are already labeled as fraudulent or non-fraudulent.
   * It finds the best hyperplane to separate the two classes of data.
3. **Website Features**:
   * SVM utilizes features of websites (e.g., URL, content, notifications) to determine whether a website is likely fraudulent.
4. **Real-World Applications**:
   * SVM has been widely applied in detecting fraudulent websites, email spam, text classification, and various other fields.

**Steps for Classifying Websites as Fraudulent or Non-Fraudulent Based on Their Features**:

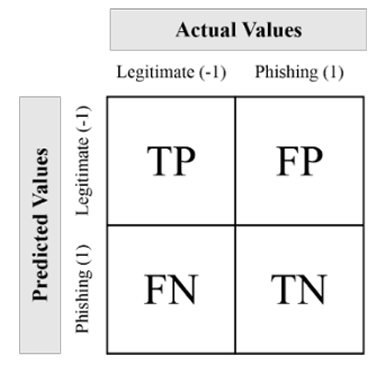
1. **Data Collection (Training Data)**:
   * Gather a dataset containing websites labeled as fraudulent or non-fraudulent.
   * Each website is represented by features such as URL length, number of links, capitalization ratio, etc.
2. **Data Preprocessing**:
   * Normalize and extract features from website URLs.
3. **Build an SVM Model**:
   * Use SVM to create a classification model.
   * SVM finds the best hyperplane to separate the data classes (fraudulent and non-fraudulent).
4. **Model Training**:
   * Train the SVM model on the labeled training dataset.
5. **Prediction and Evaluation**:
   * Use the trained model to predict whether a new website is likely fraudulent.
   * Evaluate the model’s performance using metrics such as accuracy, sensitivity, specificity, etc.

)

Ví dụ minh họa: Giả sử chúng ta có một tập dữ liệu về các trang web, trong đó mỗi trang web được mô tả bằng các đặc trưng(features) như độ dài đường dẫn, số lượng liên kết, tỷ lệ chữ viết hoa, v.v. SVM có thể học từ tập dữ liệu này để xây dựng một mô hình phân loại, dự đoán xem một trang web mới có khả năng là lừa đảo hay không dựa trên các đặc trưng tương tự.

(For illustrative example: Let’s consider a dataset of websites, where each website is described by features such as URL length, number of links, capitalization ratio, etc. SVM can learn from this dataset to build a classification model that predicts whether a new website is likely fraudulent based on similar features.)

Ví dụ 2: Kết quả về hiệu suất của hệ thống phát hiện lừa đảo sử dụng thuật toán Support Vector Machine (SVM) được trình bày dưới đây. Trong nghiên cứu này, ma trận nhầm lẫn (confusion matrix) được sử dụng để hỗ trợ tính toán độ chính xác của hệ thống. Bảng ma trận nhầm lẫn (confusion matrix) cho mỗi thử nghiệm với phương pháp tối ưu hóa tham số có thể thấy trong hình sau:



Hình 2. Confusion matrix

* The confusion matrix consists of four parts, that is:

1. True Positive (TP): the number of correct predictions from positive data.

2. False Positive (FP): the number of false predictions from positive data.

3. False Negative (FN): the number of false predictions from negative data.

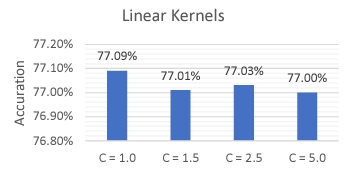
4. True Negative (TN): the number of correct predictions from negative data.

* **Để tính giá trị độ chính xác của hệ thống(system accuracy) thì chúng ta sử dụng công thức sau:**

𝐴𝑐𝑐𝑢𝑟𝑎𝑡𝑖𝑜𝑛 = (𝑇𝑃+𝑇𝑁) / (𝑇𝑃+𝐹𝑃+𝐹𝑁+𝑇𝑁) × 100%

Số lượng dữ liệu sử dụng là 11055 dữ liệu, được chia thành 10 phần bằng phương pháp chia 10 lớp. Trong cái nghiên cứu này, các thử nghiệm được thực hiện trên một thuật toán đó là SVM với tối ưu hóa tham số cho thuật toán.

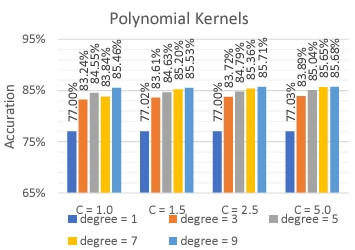
+ 2.1. Linear Kernels - Hàm kernel tuyến tính được sử dụng khi dữ liệu có thể phân biệt tuyến tính: Giá trị của tham số C là 1.0, 1.5, 2.5, và 5.0



Hình 2.1. Biểu đồ so sánh độ chính xác dựa trên tham số C.

+ 2.2. Polynomial Kernels - Hàm kernel đa thức được sử dụng để xử lý dữ liệu không phân biệt tuyến tính: gồm 2 tập tham số:

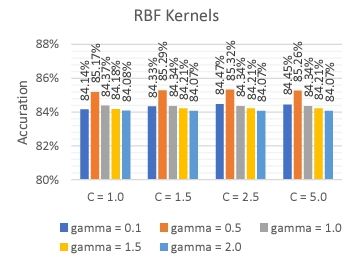
* Giá trị của tham số C là 1.0, 1.5, 2.5, 5.0
* Giá trị của tham số degree là 1, 3, 5, 7, 9



Hình 2.2. Biểu đồ so sánh độ chính xác dựa trên tham số degree và C.

+ 2.3. RBF Kernels - Hàm kernel RBF là một hàm phi tuyến tính được sử dụng phổ biến: gồm 2 tập tham số:

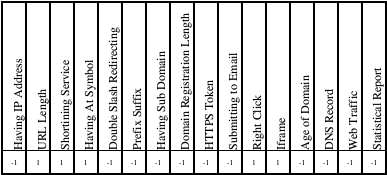
* Giá trị của tham số C là 1.0, 1.5, 2.5, 5.0
* Giá trị của tham số gamma là 0.1, 0.5, 1.0, 1.5, 2.0



Hình 2.3. Biểu đồ so sánh độ chính xác dựa trên tham số gamma và C.

* Dựa trên các biểu đồ đã được trình bày, có thể thấy rằng độ chính xác tốt nhất là 85.71% được thu hoạch trong thuật toán SVM kernel đa thức (Polynomial Kernels) với giá trị bậc là 9 và giá trị C là 2.5.
* **Kết quả của việc triển khai một hệ thống phát hiện trang web lừa đảo sử dụng thuật toán máy vector hỗ trợ (SVM):** Trong quá trình triển khai, hệ thống đã có thể nhận đầu vào từ người dùng dưới dạng URL cần được phát hiện. Sau khi URL được nhập bởi người dùng, hệ thống sẽ thực hiện quá trình trích xuất đặc trưng (a feature extraction process) đối với URL đó. Kết quả của quá trình trích xuất đặc trưng sẽ được lưu dưới dạng giá trị phân cách bằng dấu phẩy (CSV). Một ví dụ về kết quả trích xuất đặc trưng từ hệ thống có thể được thấy trong Bảng 2.

Bảng 2. Feature Extraction Results



* Dựa trên kết quả của việc trích xuất đặc trưng, việc phân loại sẽ được thực hiện bằng cách sử dụng mô hình SVM với dữ liệu huấn luyện từ bộ dữ liệu Kaggle. Và với mô hình này, hệ thống sẽ có thể phân loại các URL đã được nhập trước đó, như trong ví dụ ở Hình 2.4.



Hình 2.4. Kết quả triển khai chương trình phát hiện website lừa đảo

* **Đánh giá:**

*1. Tác động của giá trị C đối với kết quả hiệu suất của thuật toán:*

+ Tham số C là lượng phạt (penalty) được áp dụng cho lỗi phân loại (the classification error). Trong kernel tuyến tính (Linear Kernels), độ chính xác cao nhất được đạt ở giá trị C = 1.0. Gần giống với kernel tuyến tính (Linear Kernels), kết quả độ chính xác cao nhất từ kernel đa thức (Polynomial Kernels) và kernel phi tuyến tính (RBF) đạt ở giá trị C = 2.5 với độ chính xác lần lượt là 85.71% và 85.32%. **-> Từ kết quả thử nghiệm với tối ưu hóa tham số C trên ba kernels trên, có thể kết luận rằng tập dữ liệu trang web lừa đảo được sử dụng trong nghiên cứu này không tốt khi sử dụng giá trị C quá lớn.**

*2. Tác động của giá trị degree đối với kết quả hiệu suất của thuật toán:*

+ Tham số degree là một tham số chỉ có thể được tối ưu hóa trong kernel đa thức (Polynomial Kernels). Từ kết quả thử nghiệm, độ chính xác tốt nhất được đạt ở giá trị degree = 9 với giá trị C = 2.5, cũng là độ chính xác tốt nhất so với các thuật toán khác. **-> Bằng cách chú ý đến độ chính xác ở mỗi giá trị degree, có thể thấy rằng có một sự tăng đáng kể về độ chính xác. Độ chính xác thấp nhất (77.00%) có thể tăng lên đến 85.00%. Vì vậy, từ những kết quả này, có thể kết luận rằng việc tối ưu hóa tham số degree ở một giá trị cụ thể có thể tăng độ chính xác của hệ thống.**

*3. Tác động của giá trị gamma đối với kết quả hiệu suất của thuật toán:*

+ Tham số gamma là một tham số tối ưu trong thuật toán SVM kernel RBF sử dụng các giá trị 0.1, 0.5, 1.0, 1.5, 2.0. Từ kết quả thử nghiệm, độ chính xác tốt nhất được đạt ở giá trị gamma = 0.5 với giá trị C = 2.5. **-> Bằng cách chú ý đến độ chính xác của mỗi giá trị gamma và C, có thể kết luận rằng tham số gamma tối ưu là ở giá trị 0.5.**

* **Triển khai:**

+ Trong hệ thống triển khai, có một số điểm khác biệt so với hệ thống đánh giá, đó là trong hệ thống triển khai đã tiếp xúc với người dùng sẽ nhập URL để phát hiện các trang web lừa đảo. Và trong hệ thống đánh giá, có một quy trình đóng vai trò quan trọng, đó là trích xuất đặc trưng (feature extraction). Với việc trích xuất đặc trưng, hệ thống sẽ có khả năng trích xuất các đặc trưng cần thiết theo tập dữ liệu được sử dụng từ bộ dữ liệu Kaggle. Kết quả của quá trình trích xuất đặc trưng này sau này sẽ được sử dụng làm dữ liệu kiểm tra cũng như dữ liệu sẽ được phân loại bởi hệ thống sử dụng thuật toán Máy Vector Hỗ Trợ (SVM) học máy.

* **Kết luận:**

+ Từ kết quả của phân tích đã được thực hiện trong việc kiểm tra hệ thống phát hiện trang web lừa đảo bằng cách sử dụng các phương pháp học máy, có thể kết luận rằng:

1. Hệ thống phát hiện trang web lừa đảo được tạo ra bằng cách sử dụng thuật toán SVM để phát hiện các trang web lừa đảo. Trước khi URL được nhập vào hệ thống và xử lý bằng thuật toán SVM, quá trình trích xuất đặc trưng (feature extraction) sẽ được thực hiện trước tiên trên URL đã được nhập. Phương pháp được sử dụng để thực hiện trích xuất đặc trưng khác nhau cho từng đặc trưng cần được trích xuất. Các phương pháp sử dụng bao gồm trích xuất ẩn URL, trích xuất dữ liệu Whois, trích xuất dữ liệu DNS Record, trích xuất dữ liệu từ cơ sở dữ liệu Alexa và trích xuất dữ liệu trên các trang web PhishTank và StopBadware.

2. Kết quả của thử nghiệm sử dụng kiểm định chéo 10 lần với thuật toán SVM cùng tối ưu hóa tham số của thuật toán thì độ chính xác tốt nhất của mỗi thuật toán là SVM kernel tuyến tính 77.09% với giá trị tham số C là 1.0, kernel đa thức 85.71% với giá trị tham số bậc là 9 và C là 2.5, kernel RBF 85.32% với giá trị tham số gamma là 0.5 và C là 2.5. Bằng cách so sánh độ chính xác tốt nhất của 3 kernels, có thể kết luận rằng độ chính xác tốt nhất được đạt trong thuật toán SVM kernel đa thức (Polynomial Kernels) với giá trị tham số bậc là 9 và C là 2.5, tức là 85.71%.

III/ Tài liệu tham khảo:

[1] Natan, Oskar dkk. 2019. “Grid SVM: Aplikasi Machine Learning dalam Pengolahan Data Akuakultur”. Jurnal Rekayasa Elektrika Volume 15 (hlm 7-17). Surabaya: Politeknik Elektronika Negeri Surabaya.

[2] Karima, Inna Sabily. 2014. Optimasi Parameter Pada Support Vector Machine untuk Klasifikasi Fragmen Metagenome Menggunakan Algoritme Genetika. Bogor: Institut Pertanian Bogor.

[3] Halim Z. 2017. “Prediksi Website Pemancing Informasi Penting Phising Menggunakan Support Vector Machine (SVM)”. Information System for Educators and Professionals. 2 (1): 71 – 82

[4] Diani, Rima dkk. 2017. “Analisis Pengaruh Kernel Support Vector Machine (SVM) pada Klasifikasi Data Microarray untuk Deteksi Kanker”. IndonesiaJournal on Computing Volume 2 (hlm. 109-118).

[5] Preethi, V dan Velmayil, G. 2016. “Automated Phishing Website Detection Using URL Features and Machine Learning Technique”. International Journal of Engineering and Technique Volume 5.

[6] Putra, Jan Wira Gotama. 2019. Pengenalan Konsep Pembelajaran Mesin dan Deep Learning.Tokyo: Tokyo Institute of Technology.

[7] Mohammad, Rami M dkk. “Phishing Websites Features”.

[8] Purwiantono, Febry Eka, dan Aris, Tjahyanto. 2017. “Model Klasifikasi untuk Deteksi Situs Phising di Indonesia”. Surabaya: Institut Teknologi Sepuluh Nopember.

[9] Nugroho, Anto Satriyo. 2007. “Pengantar Support Vector Machine”. Jurnal Teknologi.

[10] Erton, M Wiby. 2018. “Support Vector Machine”. Jombang: Universitas KH. A. Wahab Hasbullah.