# Von Heijne and SVM (Support Vector Machine) models for the detection of signal peptides in UniProt protein sequences

Teresa Gianni, Martina Marotta, Bianca Mastroddi, Amedeo Antoci

Department of Pharmacy and Biotechnology - FaBiT, University of Bologna, Bologna, Italy

## Abstract

**Signal peptides (SPs)** are short N-terminal sequences that direct proteins to the secretory pathway. Accurate identification of SPs is essential for understanding protein localization and function. In this work, we developed and compared two computational approaches for SP prediction: a statistical model based on **Von Heijne's algorithm** and a **Support Vector Machine (SVM)** trained on physicochemical sequence features. Protein sequences with and without SPs were retrieved from UniProt, filtered, and clustered to remove redundancy. Exploratory analyses characterized sequence length, amino acid composition, and taxonomic distribution. Both models were evaluated through cross-validation and independent benchmarking. The Von Heijne model provided interpretable sequence-level insights, while the SVM achieved higher predictive performance through feature-based learning. Together, these complementary methods highlight how integrating classical sequence models with modern machine learning can enhance signal peptide prediction.
***Supplementary materials*** on GitHub

## 1 Introduction

Signal peptides (SPs)[1] are short N-terminal amino acid sequences that mediate the **transport of proteins** through the secretory pathway. They exhibit a characteristic tripartite organization — a **positively charged N-region**, a **hydrophobic core**, and a **polar C-region** containing the **cleavage site** that defines the mature protein.

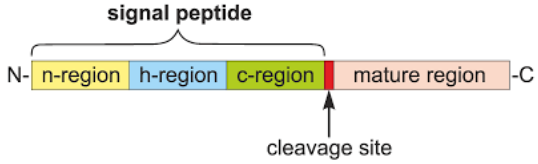


**Figure 1:** *General structure of a signal peptide (ref:[2]).*

Accurate identification of SPs is essential for understanding protein localization, membrane targeting, and secretion mechanisms in both prokaryotic and eukaryotic systems. In large-scale protein annotation pipelines, computational **SP prediction**[3] has become a fundamental complement to experimental methods, which remain labor-intensive and costly. Early computational approaches exploited statistical regularities within experimentally validated signal sequences. Among these, the classical work of Gunnar von Heijne (1983)[4][5] introduced a **position-specific weight matrix (PSWM)** that models residue frequencies along the SP sequence to infer its presence and cleavage site. Although conceptually simple, this model provided valuable biological insight and became a reference for later prediction tools. However, its purely statistical formulation limits its ability to capture complex dependencies and to generalize across diverse protein families.

The accurate recognition and identification of signal peptides is not merely an exercise in protein annotation, but is of fundamental importance for understanding a wide range of biological processes and for developing biomedical interventions. Correct SP identification is crucial for predicting a protein's subcellular localization, providing essential insights into its function. A dysfunction or mutation in the SP can lead to incorrect protein sorting, which has been implicated in numerous human diseases, including neurological disorders, metabolic conditions, and various forms of cancer [6]. Furthermore, knowledge of SPs is essential for biotechnology and drug production, allowing for the manipulation of cellular secretion pathways to optimize the yield of recombinant proteins of pharmaceutical interest. Advances in machine learning (ML)[7] have since enabled the integration of multiple physicochemical and compositional features to enhance protein sequence analysis. In particular, **Support Vector Machines (SVMs)**[8] have shown strong performance in classification problems due to their robustness and capacity to handle high-dimensional data. By encoding protein sequences through features such as hydrophobicity, charge, isoelectric point, flexibility, and amino acid composition, SVMs can learn complex discriminative patterns that distinguish SP-containing proteins from non-secretory ones.
In this project, we implemented and compared two complementary computational approaches for signal peptide prediction: a statistical model inspired by the von Heijne algorithm and an SVM-based machine learning classifier. Both methods were trained and evaluated on curated protein datasets to assess their predictive performance, generalization, and interpretability.
A final benchmarking phase was conducted to evaluate

both models on independent data, providing a comparative view of their strengths and limitations. This analysis demonstrates how integrating classical biological modeling with modern machine learning can improve the accuracy and interpretability of signal peptide prediction.

# 2 Materials and methods

## 2.1 Data collection

The first step involved retrieving appropriate datasets for training and testing methods for the prediction of the presence of signal peptides in proteins. To this purpose, two distinct datasets were constructed to distinguish between positive (with signal peptide) and negative (without signal peptide) protein sequences. The data were retrieved from the UniProt database through an advanced search. The detailed search queries used to construct both datasets are available at this GitHub repository.
In order to select **positive entries**, the following filters were applied:

- non-fragmented sequences
- taxonomy ID restricted to *Eukaryota*
- sequence length $\geq 40$ amino acids
- reviewed entries (*SwissProt*)
- evidence at the protein level
- presence of a signal peptide feature

The advanced search did not allow filtering by signal peptide length (greater than 13 residues) or by the presence of a cleavage site. However, signal peptides shorter than 14 residues rarely reflect the typical length of functional signal peptides. For this reason, an additional filtering step was implemented in Python. Two custom functions were developed to automatically handle API pagination: one to extract the link to the next results page from the HTTP headers (`get_next_link`), and another to iteratively retrieve all data batches (`get_batch`). After applying the conditions that the signal peptide length must exceed 13 residues and the cleavage site must be defined, the final positive set contained `2,932 entries`.
To construct the **negative dataset**, the same selection criteria applied to the positive set were used, except for the presence of the signal peptide. In this case, sequences annotated with a signal peptide (any evidence) were explicitly filtered out. In addition, only proteins experimentally localized to compartments incompatible with signal peptides (cytoplasm, nucleus, mitochondria, plastid, peroxisome, and cell membrane) were included.
An additional condition was also implemented in the Python script to detect the presence of transmembrane (TM) helices within the first 90 amino acid residues. This information is important because TM segments share strong similarities with signal peptides; including this filter helps to distinguish between transmembrane regions and true signal peptides. A total of 1,384 proteins meeting this criterion were identified. The final negative set comprised `20,615 entries`.
The resulting datasets were exported in both `.tsv` and `.fasta` formats, grouped according to the taxonomic kingdom of each sequence (Fungi, Viridiplantae, Metazoa, or others).

## 2.2 Data preparation

In this step, the positive and negative datasets were split into training and test sets. Subsequently, the training set was further divided into five validation subsets for cross-validation purposes.
Before splitting the data, the FASTA files obtained during the data collection phase were used for the clustering step in order to remove redundancy and identify representative sequences. Redundancy reduction was performed using the MMseqs2 software, applying a minimum sequence identity threshold of 30% and an alignment coverage threshold of 40%. These parameters ensured that proteins with potentially shared structural or functional similarity were clustered together, thereby maintaining independence between the training and test sets. Furthermore, they ensured that a substantial portion of each sequence was considered while still allowing the detection of local similarities.
From the .tsv files generated during the clustering phase, the identifiers in the first column were extracted using a Bash command to produce two lists of unique sequence IDs corresponding to the positive and negative sets. These identifiers were then used to filter the original datasets, retaining only the representative sequences and generating the final non-redundant dataframes with the following columns: `['id', 'organism_name', 'kingdom', 'sequence_length', 'cleavage_site (for positives) / transmembrane (for negatives)']`. At this stage, the non-redundant positive and negative datasets were randomly split into 80% for training and 20% for testing, using a fixed random seed to ensure reproducibility. Subsequently, both training sets (positive and negative) were divided into five subsets to enable cross-validation. An additional column, `"sp_type"`, was added to label positive (1) and negative (0) protein sequences, in both the positive and negative training and test datasets, which were then concatenated while preserving the distinction between classes and between the training and test partitions.

## 2.3 Data analysis

To better understand the structure and characteristics of the training and test datasets, several exploratory analyses and visualization techniques were performed. The main goal was to assess data quality and ensure an appropriate representation of proteins within the datasets, while identifying potential biases or missing values. All plots and their corresponding descriptions are provided in the . The analyses and visualizations were carried out in Python using Pandas, Numpy, Seaborn and Matplotlib libraries. Figures were exported as PNG files.

### 2.3.1 Comparison between sequence length in positive and negative entries

Two histograms were generated to compare the distributions of sequence lengths between positive and negative entries. Sequence lengths were $\log_{10}$-transformed to improve visualization and to avoid considering extreme values. The histograms were normalized (parameter stat='probability') to enable comparison across datasets with different sample sizes. A kernel density estimation (kde=True) was applied to obtain a smooth representation of the distributions. Colors were assigned according to the sp-type variable

(positive vs. negative sequences) via the hue parameter.

#### 2.3.2 Density distribution of signal peptide lengths

Positive entries were selected from the training and test datasets by filtering sequences with `sp_type = 1`. The resulting dataframes were used to obtain the density distribution of signal peptide (SP) lengths. The variable cleavage-site was used as the x-axis, representing the end of each SP. Kernel density plots were generated to visualize and compare the distributions between the two datasets, which were displayed together in a single figure for direct comparison.

#### 2.3.3 Comparative aminoacidic composition of signal peptides

To compute the amino acid composition of signal peptides, sequences were retrieved from FASTA files in section 2.2 and merged with the corresponding cleavage site information contained in the dataframes of positive training and test entries. This resulted in two new dataframes with three columns: `['id', 'sequence', 'cleavage_site']`. Two Python functions were implemented:
(i) `cut_sp_sequence`, to extract only the SP portion of each sequence based on the cleavage site position (the position at which the SP ends);
(ii) `frequency_calculator`, to compute the relative frequency (expressed as percentages) of each amino acid belonging to the SP sequences of the training and test datasets. The resulting frequencies were compared against the average amino acid frequencies reported for the entire SwissProt database (available at https://web.expasy.org/docs/relnotes/relstat.html). Comparative histograms were generated to visualize the amino acid composition of SPs in the training, test sets and SwissProt reference.

#### 2.3.4 Taxonomic classification at kingdom and species level

To visualize the kingdom-level distribution of the training and test datasets, the number of entries belonging to each taxonomic category was counted for both datasets using the `value_counts()` function in Pandas, and the resulting proportions were plotted as pie charts.

#### 2.3.5 Sequence logos of signal peptide cleavage sites

For sequence logo generation, subsequences around the predicted cleavage site were extracted from positive entries in the training and test datasets. For each sequence, a 15 residue window (positions −13 to +2 relative to the cleavage site) was retrieved using the function `cut_sp_sequence_logo`. The resulting subsequences were exported in FASTA format for each dataset via `to_fasta`. Sequence logos were then generated using WebLogo based on these FASTA files.

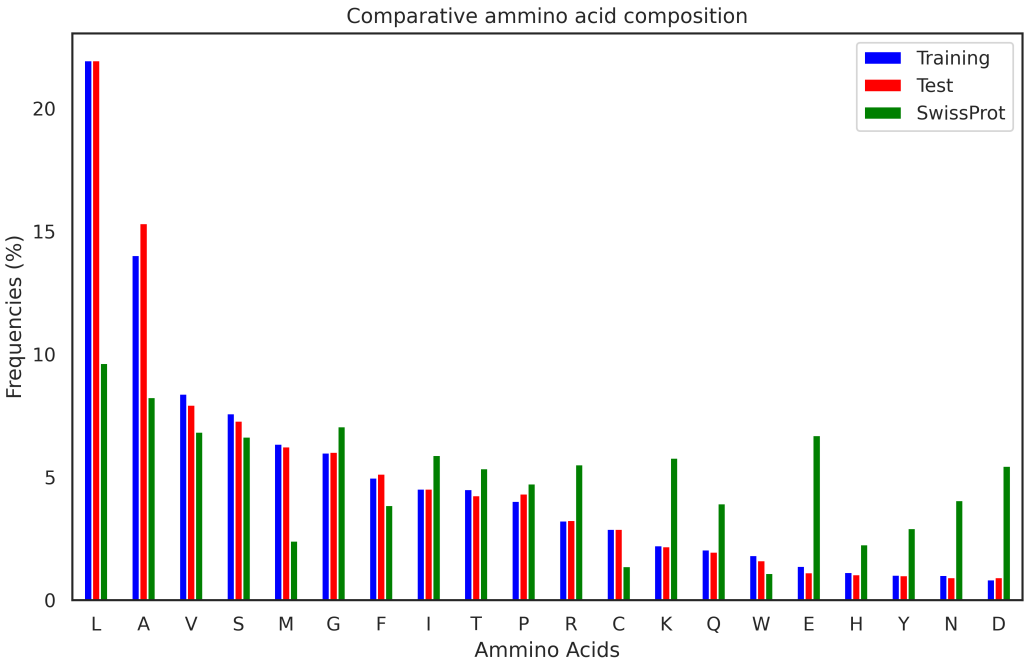


**Figure 2:** *The amino acid composition of signal peptides was compared among the training and test datasets and the general amino acid composition of all proteins in the SwissProt database, using only the positive protein sequences. The training and test sets displayed very similar amino acid distributions, consistent with the balanced nature of the datasets. Compared to the overall SwissProt distribution, signal peptides showed a higher frequency of hydrophobic residues,particularly leucine (L), which accounted for approximately 20% of residues compared to around 10% in the general SwissProt population. Other enriched residues included alanine (A), methionine (M), phenylalanine (F), and cysteine (C), which are characteristic of the hydrophobic core of signal peptides. Conversely, the relative frequency of acidic residues, such as glutamic acid (E) and aspartic acid (D), was markedly lower in the signal peptides.*

## 2.4 Von Heijne model

The first of the two methods used to train a model capable of distinguish between the presence or absence of a signal peptide in protein sequences is based on the Von Heijne algorithm. The system utilises a statistical model named **Position-Specific Weight Matrix (PSWM)**, which enables the training process to be conducted by modelling the amino acid distribution around recognised cleavage sites (obtained from section 2.1) and then comparing the observed patterns to the SwissProt distribution. In this project said model has been implemented in Python (the complete code is available on GitHub).

### 2.4.1 Construction of the PSWM matrix

Firstly, the matrix is implemented and trained using signal peptide sequences of 15 residues length corresponding to the window `[-13 +2]` with respect to the **cleavage site**. It undergoes subsequent testing on the validation set to determine the thresholds, and is ultimately tested on the test set with unseen data. This matrix provides a representation of amino acid frequencies at specific positions on a sequence motif of a given length ($L$). In particular, the columns of the matrix correspond to the positions of the motif, while the rows refer to the amino-acids (20 for proteins). The implementation consists of the below points:

1. **Initialization** of a matrix `(15x20)` with all values equal to 1, with 20 representing the possible amino acids and 15 indicating the position of the sequences.

2. **Count** of the amino-acids occurrences at each position ($C_k$).

3. Creation of a **Position-Specific Probability Matrix (PSPM)** where the observed probability for each residue is calculated by normalizing the count for the number of sequences in the training set and pseudocounts[1] with the formula:

$$M_{k,j} = \frac{1}{N+20}\left(1 + \sum_{i=1}^{N} I(s_{i,j} = k)\right) \qquad (1)$$

   Where:

   - $s_{i,j}$ is the observed residue of aligned sequence $i$ at position $j$;
   - $k$ is the residue corresponding to the k-th row in the matrix;
   - $I(s_{i,j} = k)$ is an indicator function (1 if the condition is met, 0 otherwise).

4. **Weight** the PSPM matrix by confronting the frequencies obtained with the SwissProt frequencies and transforming values using a logarithmic scale to obtain the final log-odds matrix.

$$W_{k,j} = log\frac{M_{k,j}}{b_k} \qquad (2)$$

   Where:

   - $W_{k,j}$ is the log-odds weight for residue $k$ at position $j$;
   - $b_k$ is the background frequency of residue type $k$ in the model.
   - $M_{k,j}$ is the observed frequency of residue $k$ at position $j$ in the alignment obtained by eq. (1)

   If $W_{k,j} > 0$ (when $M_{k,j} > b_k$) it signifies that the residue $k$ at position $j$ is more likely to be an important site than random (background distribution). When $W_{k,j} = 0$ (i.e. $M_{k,j} = b_k$), it can be deduced that the residue $k$ at position $j$ is more likely to be a random site then a functional one, since the probabilities are the same. Finally, if $W_{k,j} < 0$ (i.e. $M_{k,j} < b_k$) it is less probable that the residue $k$ will be found at position $j$ than expected.

### 2.4.2 Score the motif on new sequences

Given a PSWM matrix $W$, a new sequence $x$ of length $L$ can be evaluated by assigning to it a log-likelihood score of the occurence of a particular motif (in our case, the signal peptide) as follows:

$$score_{(X|W)} = \sum_{i=1}^{L} W_{x_i}, i \qquad (3)$$

### 2.4.3 Cross-validation

In order to evaluate the procedure for building a Von Heijne model, a cycle of cross-validation has been implemented. The data of the training set, which has been split into five validation subsets(2.2) are used as: **3 training** sets, **1 validation** set and **1 test** set. In particular the cross-validation is performed in 5-runs made by all the possible combinations of set types (obtained with the function `make_groups()`). Each run of cross-validation carries out the following operations:

1. The PSWM matrix is created using the signal peptide sequences from the current train set.
2. The Von Heijne scores eq. (3) are calculated by evaluating each 15-residue window in the first 90 amino acids of each sequence in the current validation set. This establishes an optimal score threshold for classifying sequences as having or not the signal peptide.
3. The same scores from point 2 are calculated, and the chosen threshold is then used to make predictions.
4. The performance of the model's ability to predict is evaluated using the following metrics: Matthews Correlation Coefficient (MCC), Accuracy (ACC), Recall (or Sensitivity) (SEN), Precision (PPV), Confusion Matrix (CONF) and F1 score. These are supplied by the Python Library `'sklearn.metrics'`.
5. The Precison-Recall Curve and the Confusion Matrix are plotted for a better visualization of the results.

### 2.4.4 Evaluation on the benchmarking set

In this section, the procedure adopted in the cross-validation step is repeated, with the validation sets (1, 2, 3 and 4) designated as the training set, validation set 5 as the validation set and the benchmark set (created in section 2.2 and never used until now) as the test set.

[1] Pseudocounts are added to avoid zero probabilities (if, in position $j$ of a multiple alignment, no sequence contains a certain amino acid $k$), which would make it impossible to later calculate weights using log odds. This is also the reason why the matrix has been initialized at 1 and not at 0.

Therefore, a PSWM matrix is created using the training set's data. Subsequently, this matrix has been used to score the validation set's sequences and choose an optimal threshold (5.9). Finally, the scores were calculated also for the sequences in the benchmark set data. The presence or absence of a signal peptide was predicted using the established threshold. The resulted performances are:

## 2.5 SVM model

The second model that has been implemented and trained to distinguish between the presence or absence of a signal peptide is based on a Support Vector Machine (SVM), that in this project is built with the function `Support Vector Classifier (SVC)` of `scikit-learn`. The objective of this type of model is to separate the classes of the training set by dividing them with a hyperplane. In this sense, the **Support Vectors** are the points of the training set closest to the separating hyperplane, meaning that these points are pivotal in determining the position of the separating hyperplane. In situations where classes are non-linearly separable, the Support Vector Machine's objective is to identify the hyperplane that **maximises the margin** between the two classes, while assigning a penalty $C$ to the errors (i.e. the points that encroach upon the margin). The parameter $C$ thus controls the tightness of the margin. If $C$ is set high, the SVM will accept few errors, and there will be a risk of overfitting[2]. Conversely, if $C$ is small, the SVM will tolerate more errors, so it is more prone to misclassifications. It is therefore essential to choose this parameter carefully, finding a trade-off between these two situations. In cases of non-linearly separable classes, the most common strategy is to transform the data into a multidimensional space through a function $\phi(x)$. The calculation of the decision boundary will be executed as follows:

$$f(x) = w \cdot \phi(x) + b \tag{4}$$

where $\phi(x)$ is the representation of the point $x$ in the transformed space. Since computing explicitly the transformation can be really computational expensive, the SVM model utilises the so called **kernel function** that returns as output value the **scalar product** in the transformed space:

$$K(x_i, x_j) = \langle \Phi(x_i), \Phi(x_j) \rangle \tag{5}$$

In this particular case the SVC has been implemented using the Radial Basis Function (RBF) or Gaussian kernel function with the parameter `kernel='rbf'`:

$$K(x_i, x_j) = \exp\left(-\gamma ||x_i - x_j||^2\right) \tag{6}$$

Where:

- $x_i$ and $x_j$ are the two input vectors
- $|| \cdot ||$ is the Euclidean norm[3]
- $\gamma > 0$ is a parameter that controls the range of action of the Gaussian around each point (large $\gamma \rightarrow$ each point affects only nearby points, small $\gamma \rightarrow$ each point affects many others points).

### 2.5.1 Features extraction

Before implementing the model, feature extraction was performed to identify all the possible features that could be useful in learning how to distinguish the signal peptide sequences. Furthermore, it is important to ensure that data is organised in a format that can be efficiently fed into the SVM model. Based also on the analysis in section 2.3 the features chosen are:

- **Amino acid composition**: stored in a 20-dimensional array for each sequence of the database in which each value corresponds to the frequency of a particular amino acid in that sequence.
- **Hydrophobicity**: measures the overall hydrophobic character along the sequence (based on the Kyte-Doolittle scale and sliding window of 5 residues over the first 40 amino acids).
- **Net Charge**: equivalent to the overall electrostatic charge (KLEP840101 scale, window 2).
- **Hydrophilicity**: indicates exposed sequence regions that are hydrophilic (HOPT810101 scale, window 5).
- **Helix Propensity**: is the tendency to form $\alpha$-helices, reflecting secondary structure preference (CHAM830101 scale, window 5).
- **$\beta$-sheet propensity**: is the propension of an aminoacid to form a $\beta$-sheet in the secondary structure of a protein. It is calculated according to the Chou-Fasman propensity scale (CHOP780202, window 7).
- **Flexibility**: measures local backbone flexibility (BHAR880101 scale, window 7).
- **Isoelectric Point (pI)**: indicates the local charge balance (ZIMJ680102 scale, window 2).
- **Bulkiness**: measures steric volume of side chains, related to packing and solvent accessibility (ZIMJ680102 scale, window 7).
- **Von Heijne Score**: quantifies the structural resemblance of the sequence to the canonical N-, H-, and C-regions of known signal peptides. This score is derived from a PSWM trained on positive sequences, where the final feature value represents the maximum score obtained at any potential cleavage site.
- **Shannon Entropy**: measures the complexity and information content (randomness) of the amino acid distribution in the N-terminal region. It is calculated over the first 30 residues of the sequence to assess the low variability characteristic of the signal peptide's hydrophobic core.

The final feature set was created by summarizing each physicochemical property using a sliding window technique, resulting in key statistical descriptors like the mean, standard deviation, and extreme values (maximum/minimum), along with other relevant summary metrics.

[2]Overfitting occurs when a machine learning model learns not only the underlying patterns in the training data, but also the noise and random fluctuations. As a result, the model performs very well on the training set but poorly on new, unseen data.

[3]The Euclidean norm of a vector is the square root of the sum of the squares of its components. It represents the magnitude of the vector in Euclidean space.

### 2.5.2 Features selection

To enhance the model's generalization capability and reduce complexity, a feature selection (FS) procedure was implemented prior to final model training. The main steps followed a structured workflow:

- **Pipeline Definition**: prior to training, a fixed pipeline was defined to handle data preparation and classification. This setup ensured that all features were properly scaled using a `StandardScaler()` before being passed to the Support Vector Machine, which was implemented with the non-linear RBF kernel.
- **Hyperparameter Search**: a grid search was performed to optimize the SVM hyperparameters $C$ and $\gamma$. The range explored included $C \in \{0.1, 1.0, 10.0\}$ and $\gamma \in \{\text{'scale'}, 0.01, 0.1, 1.0\}$.
- **Feature Importance Ranking**: a `Random Forest Classifier (RFC)` was trained on the training data, and the features were ranked based on their **Gini Importance**[4] to select the top 20 most important features.
- **Subset Optimization**: Different top $k$ subsets of features (e.g., $k = 1, 2, \ldots, 20$) were evaluated using the SVM to determine the optimal subset size, based on cross-validation accuracy. The most significant features that were ultimately selected included `the score_1, hydrophilicity_1, hydrophobicity_11, hydrophobicity_12, hydrophobicity_10, hydrophobicity_13, flexibility_1, hydrophobicity_9`.
- **Cross-validation**: as in the Von Heijne model implementation (section 2.4.3), the procedure described above has been repeated for 5 runs of cross-validation, dividing the validation sets of the training in the same way.
- **Final Model Training**: two final models were trained: **SVM with Feature Selection (FS)** using the optimal subset of features that were selected in all the 5 runs of cross-validation, and a **Baseline SVM** trained using all available features, both with their respective best-performing hyperparameters determined by a grid search on th validation set.

### 2.5.3 Evaluation on the Benchmarking Set

The final evaluation compared the performance of the two trained models on an independent benchmarking set. The model trained using all features slightly outperformed the SVM with Feature Selection (FS) model across all key metrics. Specifically, the all-feature model achieved a superior MCC of 0.90 (versus 0.83 for FS) and an F1 Score of 0.91 (versus 0.85 for FS), demonstrating better predictive quality. The Confusion Matrices confirmed this trend. The all-feature model recorded a higher number of True Positives (197 vs 181) and a lower number of False Negatives (22 vs 38), indicating better identification of signal peptide-containing sequences (the positive class).

While both models achieved excellent overall classification performance, with nearly identical ROC AUC values (AUC = 0.980 with FS, AUC = 0.982 without FS), the minor yet consistent advantage of the model using all available features suggests that, in this specific context, the dimensionality reduction introduced by feature selection may have discarded discriminative information valuable to the SVM.

## 2.6 False Positives and False Negatives analysis

A detailed post-validation analysis of incorrect predictions was conducted to investigate the limitations of both the von Heijne model and the Support Vector Machine (SVM) classifier, focusing on False Positives (FP) and False Negatives (FN). This analysis aimed to identify physicochemical and structural characteristics that differentiate correctly predicted signal peptides (True Positives, TP) from misclassified sequences, and they may therefore have led the models astray.

Analysis of the error distribution by **Kingdom** revealed that for the SVM, the majority of both False Positives ($\approx 59.3\%$) and False Negatives ($\approx 59.3\%$) originated from the Metazoa sequences. Furthermore, sequences corresponding to transmembrane proteins were identified as a specific challenge, exhibiting a false positive rate on transmembrane proteins of $\approx 10.3\%$, significantly higher than the overall dataset false positive rate of $\approx 1.35\%$.

The investigation into the physicochemical characteristics provided key insights into the nature of False Negatives (FNs). Analysis of the Von Heijne scores distribution showed that FNs predominantly clustered at lower, sub-threshold scores, indicating they represent sequences the von Heijne model struggled to identify . Similarly, the distribution of signal peptide sequence length demonstrated that FNs often correspond to shorter-than-average sequences, which deviates from the typical length distribution observed in True Positives (TP) .

Crucially, comparison of feature distributions highlighted significant biophysical differences:

- **Hydrophobicity**: the distribution plots showed that FNs consistently exhibited lower hydrophobicity compared to TPs across multiple feature indices (e.g., hydrophobicity 9-13) .
- **Flexibility and Hydrophilicity**: FNs also showed greater variability and generally higher flexibility compared to the TP group, confirming that sequences with atypical or non-canonical structural characteristics were the most challenging to classify.

[4] Gini importance quantifies how much each feature reduces Gini impurity in a decision tree or random forest, summed over all splits where the feature is used.

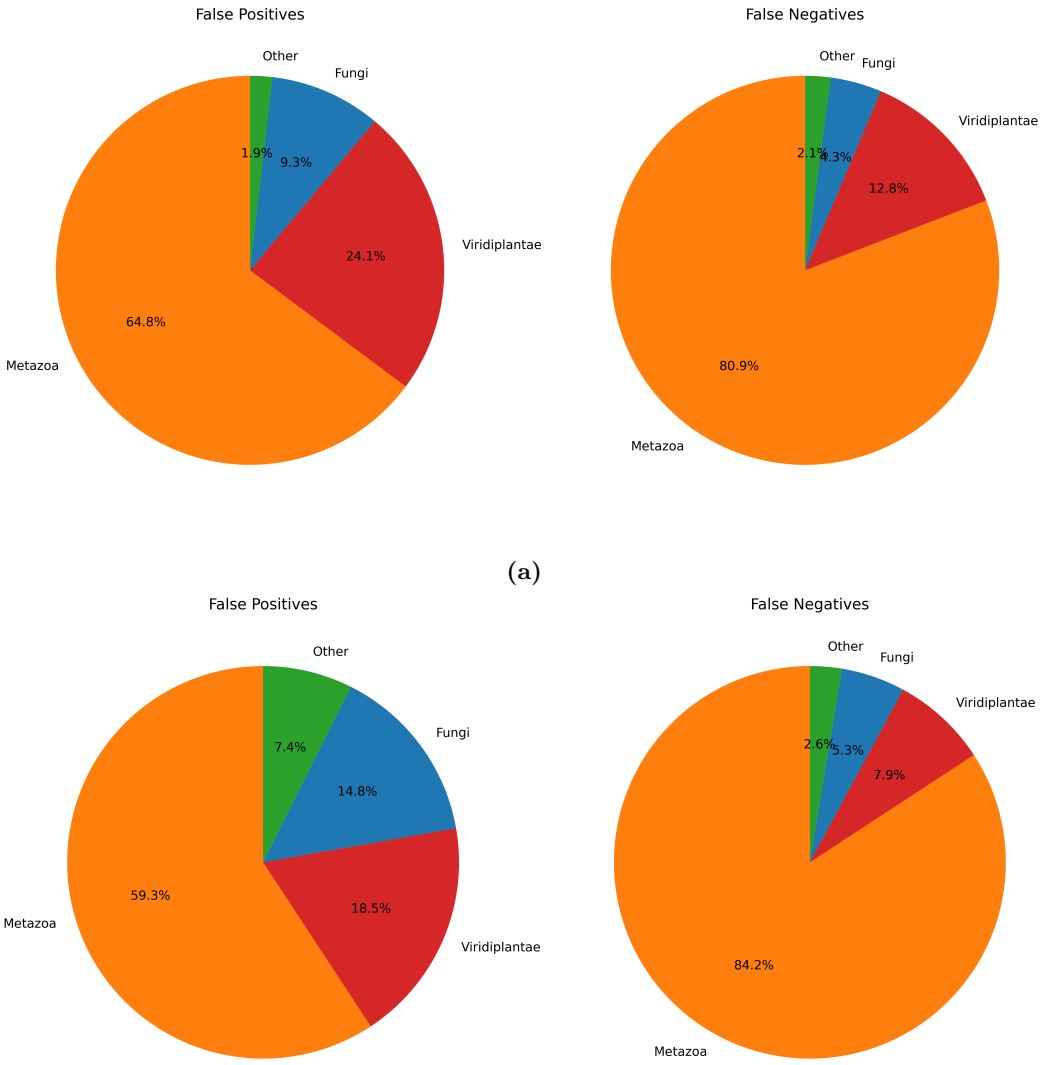


**Figure 3:** *The figures show the proportional distribution across kingdoms of samples misclassified by the* ***Von Heijne model (a)*** *and* ***SVM classifier (b)****. It has been observed that the initial datasets contained a higher number of examples belonging to the kingdom 'Fungi' compared to 'Plants'. In contrast, these pie charts reveal a greater prevalence of false positives among plants than among fungi. This may be due to the presence of* ***plastid signal peptides*** *in plants, which are similar to those investigated in this analysis.*

Overall, the von Heijne model was precise but conservative, missing short or weakly hydrophobic cases. The SVM model achieved better overall recall by successfully recovering many of these challenging sequences, though it incurred a slightly higher number of False Positives.

# 3 Results

The performance of the Von Heijne model, based on a Position-Specific Weight Matrix (PSWM), was evaluated using an optimized threshold of 5.9. The model yielded a high Accuracy (92.7%), though this can be attributed to the large number of True Negative sequences in the imbalanced dataset. Overall performance, however, was more modest, evidenced by a lower Precision (0.614) and Recall (0.785). The MCC score of 0.653 suggests a definite correlation between the model's prediction and the true classification, but highlights substantial room for improvement, particularly regarding the high number of False Positives.

The SVM model was based on a curated assortment of physicochemical sequence features that included the Von Heijne prediction score as a highly informative meta-feature and utilized the Radial Basis Function (RBF) kernel. The performance of this model was measured both with and without a Feature Selection (FS) step. Both SVMs achieved markedly high performance scores, though the SVM without FS clearly outperformed its counterpart in every metric. The model with all features achieved strong Precision (0.92) and Recall (0.89), and an overall Accuracy of 98.0%. The MCC of 0.90 indicates a superior predictive capability and a strong balance of Precision and Recall.

Comparing the two models, the SVM demonstrated a clear advantage over the standalone Von Heijne approach, with a 25-point increase in MCC (from 0.65 to 0.90) and a 22-point increase in F1 score (from 0.69 to 0.91). This superior performance highlights the benefit of using a hybrid feature-based machine-learning approach, which is capable of capturing a more complex scope of physicochemical

patterns while leveraging the strength of a well-established base line score ( Von heijne ) that cannot be identified solely by direct sequence data.

The False Positive analysis indicates that both models struggled with sequences belonging to the Metazoa kingdom ($\approx$ 59.3% of SVM errors) and proteins known to contain transmembrane (TM) helices in the N-terminal region. The False Negative analysis showed that although the SVM achieved a higher Recall (0.89) than the Von Heijne model (0.785), it may still struggle with atypical signal peptides that present below-average hydrophobicity or non-canonical cleavage motifs.

# 4 Conclusions

In this work, we successfully implemented and benchmarked two distinct strategies for Signal Peptide prediction: the Von Heijne statistical model and a Support Vector Machine classifier with a hybrid feature set. The Von Heijne model served as a valuable and interpretable baseline, providing sequence-level insight based on the amino-acid distribution around the cleavage site motif. While its structural simplicity and clear biological foundation are useful for understanding the conceptual framework of the problem, the model produced only a moderate predictive capability on its own.

The SVM classifier, on the other hand, leveraging a modern Machine-Learning approach and a diverse feature set that integrated the interpretable Von Heijne score with deeper physicochemical insight, achieved a significantly higher predictive performance. This demonstrates the ability of advanced ML approaches to integrate and simplify diverse data sources, and the importance of such ability when dealing with complex problems. Future work could focus on further expanding the feature set to include other protein sequence properties not investigated in this work, or on implementing other sophisticated ML architectures.

# References


[1] Hajar Owji et al. "A comprehensive review of signal peptides: Structure, roles, and applications". In: *European Journal of Cell Biology* 97.6 (2018), pp. 422–441. ISSN: 0171-9335. DOI: https://doi.org/10.1016/j.ejcb.2018.06.003. URL: https://www.sciencedirect.com/science/article/pii/S0171933518300189.

[2] Michał Burdukiewicz et al. "Prediction of Signal Peptides in Proteins from Malaria Parasites". In: *International Journal of Molecular Sciences* 19.12 (2018). ISSN: 1422-0067. DOI: 10.3390/ijms19123709. URL: https://www.mdpi.com/1422-0067/19/12/3709.

[3] Stefano Grasso et al. "Signal Peptide Efficiency: From High-Throughput Data to Prediction and Explanation". In: *ACS Synthetic Biology* 12.2 (2023), pp. 390–404. DOI: 10.1021/acssynbio.2c00328.

[4] Gunnar von Heijne. "Patterns of amino acids near signal-sequence cleavage sites". In: *European Journal of Biochemistry* 133.1 (1983), pp. 17–21. DOI: 10.1111/j.1432-1033.1983.tb07424.x.

[5] Gunnar von Heijne. "A new method for predicting signal sequence cleavage sites". In: *Nucleic Acids Research* 14.11 (1986), pp. 4683–4690. DOI: 10.1093/nar/14.11.4683.

[6] Sneider Alexander Gutierrez Guarnizo et al. "Pathogenic signal peptide variants in the human genome". In: *NAR Genomics and Bioinformatics* 5.4 (Oct. 2023), lqad093. ISSN: 2631-9268. DOI: 10.1093/nargab/lqad093. eprint: https://academic.oup.com/nargab/article-pdf/5/4/lqad093/52214079/lqad093.pdf. URL: https://doi.org/10.1093/nargab/lqad093.

[7] Jing Xu et al. "Comprehensive assessment of machine learning-based methods for predicting antimicrobial peptides". In: *Briefings in Bioinformatics* 22.5 (Mar. 2021), bbab083. ISSN: 1477-4054. DOI: 10.1093/bib/bbab083. eprint: https://academic.oup.com/bib/article-pdf/22/5/bbab083/40260025/bbab083.pdf. URL: https://doi.org/10.1093/bib/bbab083.

[8] D. Valkenborg et al. "Support vector machines". In: *American Journal of Orthodontics and Dentofacial Orthopedics* 164.5 (2023), pp. 754–757. DOI: 10.1016/j.ajodo.2023.08.003.