CHALMERS
UNIVERSITY OF TECHNOLOGY | UNIVERSITY OF GOTHENBURG

# Benchmarking Machine Learning Methods for Peptide Activity Predictions

Master's thesis in Computer science and engineering

Boel Knutson and Lida Meskini Moudi

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2022

Master's thesis 2022

# Benchmarking Machine Learning Methods for Peptide Activity Predictions

Boel Knutson and Lida Meskini Moudi

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2022

Benchmarking Machine Learning Methods for Peptide Activity Predictions
Boel Knutson and Lida Meskini Moudi

Supervisors: Ola Engkvist, Department of Computer Science and Engineering
Gökçe Geylan, AstraZeneca
Examiner: Alexander Schliep, Department of Computer Science and Engineering

Master's Thesis 2022
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in $\LaTeX$
Gothenburg, Sweden 2022

Benchmarking Machine Learning Methods for Peptide Activity Predictions

Boel Knutson and Lida Meskini Moudi
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

# Abstract

One of the main challenges in the drug discovery process is to find a suitable compound for further analysis. The compound must affect the target relevant for the specific disease, while at the same time have desired properties to make it a safe and efficient drug candidate. The task of finding and optimizing these compounds is a long and expensive process. Therefore, using machine learning algorithms to predict the properties of compounds can speed up the process and reduce the cost. To use the algorithms, the information about the compounds must be translated into a numerical representation. The choice of representation and algorithm is of greatest importance since the predictions must be reliable to avoid late-stage failures in the drug discovery process.

The objective of this thesis was to investigate if a molecular representation together with a machine learning model could be found to accurately predict the potency of peptides. This was done through a benchmarking study where different sequence-based descriptors and predictive models were combined to see if one combination worked well for various types of peptides. The descriptors were Z-scales, pseudo amino acid composition, and one-hot representation, and were combined with two different machine learning models, namely support vector classifier and random forests classifier. The results show that one-hot representation outperforms Z-scales and pseudo amino acid composition, however, the predictive model depends on the characteristics of peptides.

Keywords: Drug discovery, peptide, classification, molecular representation, Z-scales, pseudo amino acid composition, one-hot representation, random forests, support vector machines.

# Acknowledgements

We would like to express our thanks to our supportive and helpful supervisors Gökçe Geylan and Ola Engkvist who have supported us through this Master's thesis. Not only have they provided us with invaluable feedback but also shared their knowledge and expertise. We would also like to thank AstraZeneca for giving us this opportunity. Lastly, we want to mention our friends and family who have kept our motivation and spirits high during this time.

Boel Knutson & Lida Meskini Moudi, Gothenburg, June 2022

# Contents

# List of Figures

# List of Tables

# 1

# Introduction

Being able to predict the potency of peptides with the help of machine learning can improve the drug discovery process to become both faster and cheaper. The discovery and development of a new drug is a long and expensive process, where the idea to the finished product can take approximately 12-15 years and cost around USD$1 billion [2]. The discovery process begins by finding a suitable target, for example, an enzyme or gene, that is relevant to a disease. This is followed by finding and optimizing a compound. The compound must have properties suitable for a drug while affecting the target of interest in a useful pharmacological context [3]. The compound can for example be a small molecule, biological therapeutic, or peptide, and common properties to evaluate are the absorption, distribution, metabolism, excretion, and toxicity (ADMET) of the compound [4]. After this step, the chosen compound goes into pre-clinical and clinical development to evaluate its safety, efficacy, and dosing [4]. Lastly, the compound must be reviewed and approved by certain agencies before it finally can enter the market. However, the majority of potential drugs fail before entering these last steps of the process [3]. From a time and cost perspective, it is better that these failures happen as early as possible, and that is how predicting potency by using machine learning can improve the drug discovery process.

Drugs can be categorized based on the type of active substance they rely on. For instance, 75% of the global pharmaceutical market is accounted for by small molecule drugs while only 5% is accounted for by peptide drugs [5]. A peptide is a sequence of amino acids that can be found naturally in the human body or be produced synthetically [6]. Reasons why peptide drugs only account for around 5% of the market include that peptides have low oral bioavailability, low plasma stability, and short circulation time [5]. However, they have several desirable drug features such as low toxicity and high selectivity and they have been used against several serious diseases such as diabetes, cancer, and HIV. Moreover, in recent years there has been a lot of progress within the field of peptide drug development. In particular, in the last 20 years, the number of peptide drugs entering clinical development has increased considerably [5]. With this recent progress, it is desirable to explore and improve the drug discovery process for peptide drugs further.

Traditionally, properties of compounds are analyzed using *in vivo* and *in vitro* tests, which refers to experiments that are performed in living organisms or outside of organisms, such as in a test tube, respectively [7]. While these tests can provide accurate results, they are expensive and time-consuming. However, with the development of artificial intelligence and machine learning, a lot of work can be automated by *in silico* (computer-aided) modeling [8]. With *in silico* methods, it is possible to use machine learning models to predict compounds' properties faster, cheaper, and more sustainably than using *in vivo* and *in vitro* tests.

To use machine learning to predict the potency of peptides, information of the peptide must be translated into a numerical representation readable by machines [9]. In this thesis, different sequence-based encodings are explored. Sequence-based encodings are calculated from the primary structure of the peptide, i.e. from the sequence of amino acids [10].

Specifically, in this thesis, a benchmarking study is carried out to predict the potency of peptides by combining different machine learning models with various representations of peptides as input. Models used for this task are support vector machines (SVM), random forests, and encodings include one-hot representation, pseudo amino acid composition (PseAAC), and Z-scales. As the area of peptide drugs is a relatively new and expanding field, there is a need to explore which implementations work well with peptide data.

## 1.1 Problem formulation

This project aims to find the best combination of machine learning model and molecular representation in order to accurately predict the potency of peptides. Specifically, using different datasets containing peptide sequences and binary classes which represent high or low potency, the objective is to encode the sequences using sequence-based descriptors and use these representations in different machine learning models. The assay of which the dependent variable has been measured, as well as the length of peptides, differs between the datasets. A comparison between model performances for each dataset is then conducted in order to find the best combination of peptide representation and machine learning model.

## 1.2 Limitations

Rather than considering all possible machine learning algorithms, this project explores a subset of algorithms, which are selected based on their displayed performance in the existing literature. The same holds true also for the selection of peptide representations. The project is also limited to public datasets which need to be clean, well-annotated, and sufficiently large. What is considered sufficiently large is determined by the learning algorithms, as these might be more or less prone to overfitting when exposed to datasets of different sizes.

## 1.3 Thesis outline

In this report, Chapter 2 will introduce theory relevant to understanding this thesis. First, molecular representations are explained and then three different descriptors, one-hot representation, pseudo amino acid composition, and Z-scales, are presented closer, followed by an explanation of "SHapley Additive exPlanations" (SHAP) for feature exploration. Thereafter, two predictive machine learning models, support vector machine and random forests, are described. Lastly, an overview of commonly used evaluation metrics is presented.

In Chapter 3, the methodology used in this project is described. The chapter begins with a presentation of the data used and then explains the steps of data processing. This is in order for readers to be able to reproduce the same experiment. Then the motivation and practical details of the generated descriptors are explained, before going into the details of the chosen machine learning models and the motivation behind these. Lastly, the process of evaluating the different combinations of descriptors and predictive models is presented.

Next, in Chapter 4, the results of the feature exploration and experiment will be presented, as well as the results of using the best combination for each dataset on the corresponding test set. This is followed by Chapter 5, where the results will be analyzed and discussed. This includes a comparison between results for different combinations, as well as reflections and interpretations of the results. The report ends with a conclusion in Chapter 6, where the thesis is summarized.

# 2

# Theory

In the following sections, the theory of relevant concepts is presented. First, the theory behind molecular representation and the three descriptors, one-hot representation, pseudo amino acid composition (PseAAC), and Z-scales are explained. This is followed by explanations of relevant machine learning models, namely support vector machines (SVM), and random forests. An explanation of hyperparameter tuning within these models is also presented. Lastly, the formulas of relevant evaluation metrics are explained.

## 2.1 Molecular representation

A peptide is a chain of molecules, known as amino acid residues, which are connected by peptide bonds. To make the peptides readable for machine learning models, the peptide sequences are converted into molecular representations by using descriptors. The descriptors turn the sequences into numerical representations or features. Two common types of descriptors are sequence- and structure-based [10]. The structure-based descriptors contain properties obtained from the structures of peptides, while the sequence-based is calculated from the sequence of the peptide. An entire amino acid sequence is a straightforward model of peptide sample representation [11]. The advantage of such a representation is that it represents the peptide with complete information, containing both constituent amino acids and their order. Equation 2.1 shows the sequential representation of a given peptide sequence $P$ with $L$ amino acid residues,

$$P = [R_1 R_2 R_3 .....R_L], \tag{2.1}$$

where $R_i$ represents the residue in $i$th position.

Amino Acid Composition (AAC) is another sequential representation of peptide sample that is not based on the order of amino acids of sequence. Equation 2.2 shows the AAC representation of the given peptide $P$,

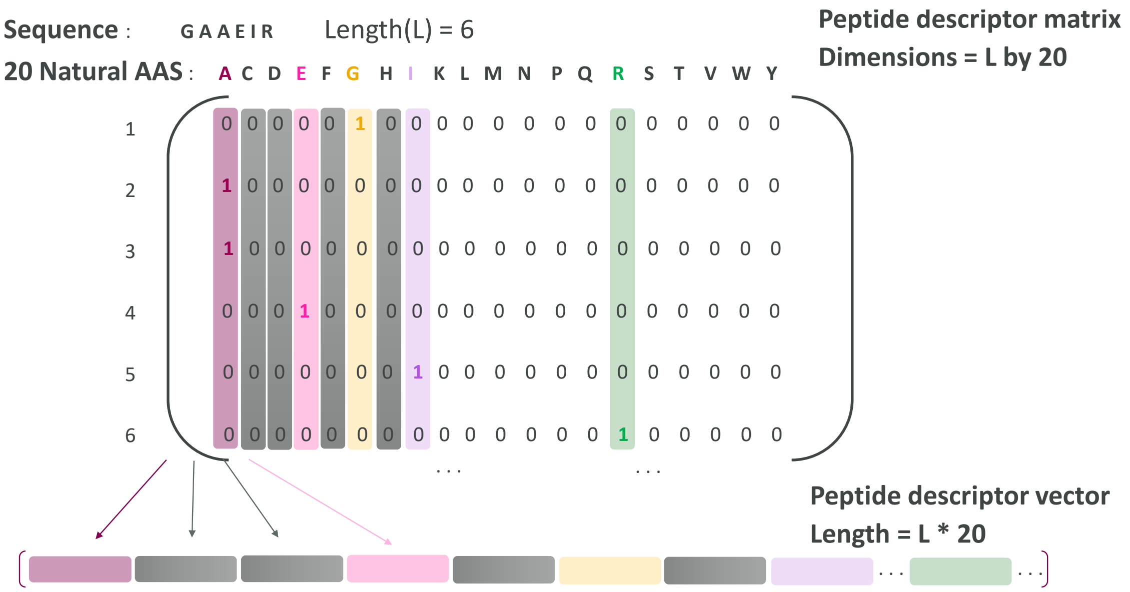
$$P = [f_1 f_2 f_3 .....f_{20}]^T, \tag{2.2}$$

where $f_u$ $(u = 1, 2, ..., 20)$ are the normalized occurrence frequencies of the 20 natural amino acids in $P$, and $T$ the transposing operator. The AAC representation is a popular and one of the simpler descriptors to compute, but its main shortcoming is the loss of the sequence-order information [11].

Three types of sequence-based descriptors are One-hot representation, Z-scales, and PseAAC are explained in the following sections.

### 2.1.1 One-hot representation

For one-hot representation, each peptide sequence is converted into a matrix $\in \mathbf{R}^{L\times 20}$, where $L$ is the length of the sequence. The columns represent the 20 natural amino acids, and the rows represent the position in the sequence. For each row, all amino acids are set to 0 except for the amino acid being encoded, which is set to 1 [12]. For instance, if the first position in a sequence is "G", the first row is a vector where the column corresponding to "G" is 1 and all other columns are 0, see Figure 2.1. To use the matrix in machine learning models, the matrix is flattened into a vector with length $L \times 20$ by concatenating the columns.



$$
\begin{array}{c|cccccccccccccccccccc}
 & A & C & D & E & F & G & H & I & K & L & M & N & P & Q & R & S & T & V & W & Y \\
\hline
1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
3 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
4 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0
\end{array}
$$

**Figure 2.1:** A schematic drawing to show one-hot matrix and one-hot vector representation.

### 2.1.2 Z-scales

Z-scales represent five z-values related to physicochemical properties of the amino acids in the peptide sequence [13]. Note that Z-scales and z-values are not related to z-scores in statistics. Hellberg et al. [14] described the variation in the peptide sequences by three principal properties, $Z_1$, $Z_2$, and $Z_3$, per amino acid in each position of the sequence. To derive these three principal properties they described 20 natural amino acids by 29 measures of various properties. The variables were scaled to unit variance and then they used principal components analysis (PCA) on the scaled data to capture the three components $Z_1$, $Z_2$, and $Z_3$. These three values of amino acid summarize all 29 physicochemical properties. Sandberg et al. [15] improved the original Z-scales introduced by Hellberg et al. [14], by publishing two additional values $Z_4$ and $Z_5$ to the three original z-values. They use 26 properties derived from 87 amino acids. The z-values mainly captures lipophilicity ($Z_1$), bulk ($Z_2$), polarity / charge ($Z_3$) and $Z_4$ and $Z_5$ are related to properties as electronegativity, heat of formation, electrophilicity, and hardness [15]. As an example, for $Z_1$ a negative value relates to a lipophilic amino acid, and a positive value relates to a polar, hydrophilic amino acid [15].

In the study by van Westen et al. [16], Z-scales performed well compared to other descriptors such as VHSE, T-scales, ST-scales, ProtFP, and BLOSUM. The descriptor also had a consistently good score for various datasets, such as dipeptides that have an inhibitory effect on the angiotensin-converting enzyme, and three proteochemometric datasets.

### 2.1.3 Pseudo amino acid composition

To overcome the disadvantage of the AAC model in providing information of sequence order, PseAAC was proposed [11]. Conventional AAC has 20 components to show the occurrence frequency of each 20 natural amino acids, however, PseAAC has more than these 20 factors. The first 20 components are the same as conventional AAC while the additional factors record some sequence-order information via various pseudo components. The additional factors can be a series of rank correlation factors along a protein chain or any combination of other factors as long as they are sequence order-correlated, as seen in Equation 2.3.

$$P = [p_1, p_2, p_3, .....p_{20}, p_{20+1}, p_{20+2}, ..., p_{20+\lambda}]^T, \tag{2.3}$$

where $\lambda$ is an integer factor, and assigning a different value to it leads to a different dimension of the PseAAC [1].

As mentioned, the first 20 components, $p_1, p_2, ..., p_{20}$ are associated with the conventional AAC of peptides. The other components $p_{20+1}, ..., p_{20+\lambda}$ are the correlation factors that reflect the 1st tier, 2nd tier, up to the $\lambda$-th tier sequence order based on correlation pattern, shown in Figure 2.2, and calculated by Equation 2.4, 2.5, and 2.6.

$$p_u = \begin{cases} \frac{f_u}{\sum_{n=1}^{20} f_i + w \sum_{k=1}^{\lambda} \tau_k} & 1 \leq u \leq 20 \\ \\ \frac{w\tau_{u-20}}{\sum_{n=1}^{20} f_i + w \sum_{k=1}^{\lambda} \tau_k} & 20 + 1 \leq u \leq 20 + \lambda \end{cases} \tag{2.4}$$

where $f_u$ $(u = 1, 2, ..., 20)$ is the normalized occurrence frequency of amino acid $u$, $f_i$ is the normalized occurrence frequency of the $i$th amino acid of the 20 natural amino acids in the protein sequence, $w$ is the weight factor, and $\tau_k$ the $k$th tier correlation factor, that reflects the sequence order correlation between all the $k$-th most contiguous residues, see Figure 2.2, and according to Equation 2.5, and 2.6.

$$\tau_k = \frac{1}{L-K}\sum_{i=1}^{L-K} J_{i,i+k}, \tag{2.5}$$

$$J_{i,i+k} = \frac{1}{\Gamma}\sum_{q=1}^{\Gamma}[\Phi_q(R_{i+k}) - \Phi_q(R_i)]^2, \tag{2.6}$$

where $\Phi_q(R_i)$ is the $q$-th function of the amino acid $R_i$, and $\Gamma$ the total number of the functions considered. For example, Chou [11] defined $\Phi_1(R_i)$, $\Phi_2(R_i)$ and $\Phi_3(R_i)$ as the hydrophobicity value, hydrophilicity value, and side chain mass of amino acid $R_i$, respectively. Therefore, the total number of functions in this example is $\Gamma = 3$.

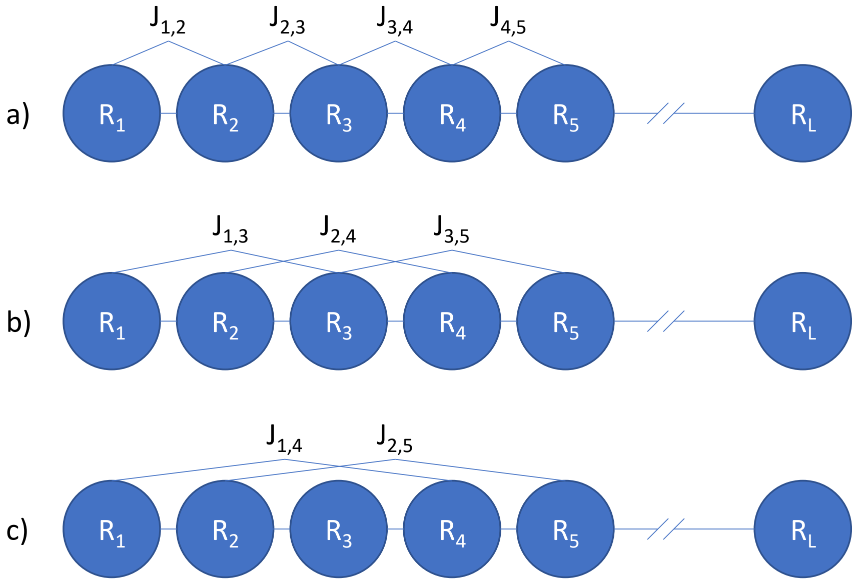


**Figure 2.2:** A schematic drawing to show (a) the 1st-tier, (b) the 2nd-tier, and (c) the 3rd-tier sequence-order-correlation pattern along a protein sequence, where $R_i$ represents the residue in $i$th position [1].

PseAAC has previously been used in various prediction tasks, such as predicting allergenic proteins [17], predicting DNA-binding proteins [18], and predicting enzyme subfamily classes [19]. In the study of predicting DNA-binding proteins made by Fang et al. [18], PseAAC with $w = 0.05$ and $\lambda = 20$, together with SVM outperformed three other descriptors, namely dipeptide composition, autocross-covariance transform, and native amino acid composition with an Matthews correlation coefficient (MCC) of 0.92, compared to scores between 0.45-0.66.

### 2.1.4 Feature exploration

Before training machine learning models with generated features, there are different ways of exploring and analyzing these to extract more information about the data. One method to use for this is SHAP. SHAP originates from the Shapley values introduced by Lloyd Shapley in 1953 for cooperative game theory [20, 21]. SHAP measures the importance of each feature for a specific prediction by the model [21]. SHAP does not only show feature importance but also the contribution of each feature to the prediction, i.e., if the feature impacts the prediction positively or negatively [22]. Higher SHAP values correspond to a higher probability for a positive prediction, and lower SHAP values correspond to a higher probability for a negative prediction. SHAP can be used to explain different models such as XGBoost and Scikit-learn tree models by using different SHAP explainers, such as TreeExplainer [22, 23].

## 2.2 Machine learning models

### 2.2.1 Support vector classifier

Support vector machine is a supervised learning algorithm that is used for both classification (SVC) and regression (SVR) tasks. The algorithm tries to find the optimal hyperplane based on the training data to classify new data points. In a two-dimensional space, this hyperplane is a simple line. SVM learns similarities between classes by finding the most similar samples between classes, which are called support vectors [24]. Based on these support vectors, the algorithm finds the optimal hyperplane by maximizing the margin between the hyperplane and all support vectors of each class. It is a global margin as it considers all classes. Linearly separable data in two dimensions is shown in Figure 2.3, where the margin (green line) is orthogonal to the hyperplane (red line) and it is equidistant to the support vectors.

When the data is not linearly separable and it is not possible to find a linear hyperplane to separate the classes, SVM transforms the data into a new representation [25]. To generate the new representation, the space is transformed by mapping data points to higher dimensions using non-linear functions. This transformation is called kernel [25]. When adding additional dimensions to the data, it might be possible to make the data separable [24]. The kernel function is used to calculate the dot product of two vectors in a high-dimensional space while using their low-dimensional counterpart for the computations [26]. Three examples of kernel functions are Linear, Sigmoid, and Radial basis function (RBF).

In SVM, hinge loss is used as the loss function, as seen in equation 2.7 [27].

$$\text{Loss}(\mathbf{w},\ \mathbf{x}, y) = \max(0, 1 - y \cdot (\mathbf{w} \cdot \mathbf{x} + b)), \tag{2.7}$$

where **w** are the learned weights of the model, $b$ is the learned bias, and $y$ and **x** are

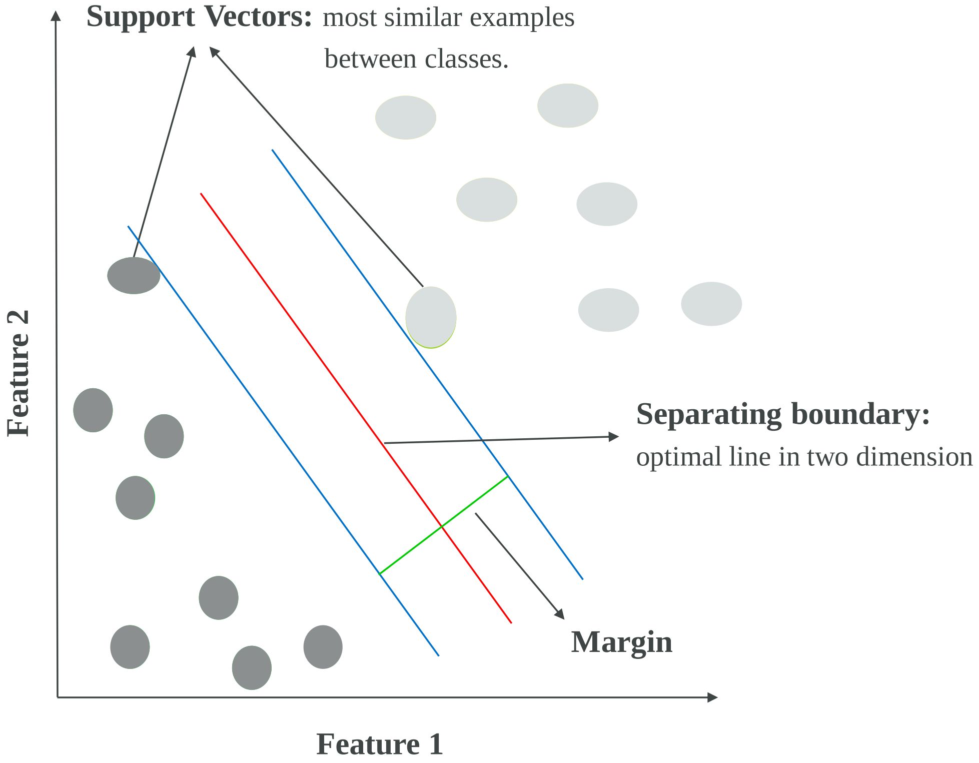

Support Vectors: most similar examples between classes.
Separating boundary: optimal line in two dimension
Margin
Feature 2
Feature 1

**Figure 2.3:** A schematic drawing to show binary SVM classifier in linear separable dataset.

the labels and features of the data, respectively.

Even though higher-dimensional features and kernel functions can reduce the number of data points being non-separable, there might still be scenarios when it is not possible. To overcome this issue, it is possible to use soft-margin SVM. This means that each data point has a slack parameter set to 0, and if a data point cannot be on the right side of the margin, it is moved in the right direction and the slack parameter increases. However, the bigger the total slack is, the more the model will be penalized [27].

In a study by Mohabatkar et al. [28], SVM together with PseAAC performed well for classifying gamma-aminobutyric acid receptor proteins, which had a length of 450–627 amino acids. The combination resulted in a MCC of 0.88, with $\lambda = 1$ and weight factor = 0.05. SVM also worked well for classifying antibacterial peptides in a study made by Lata et al. [29]. In their study SVM together with AAC resulted in an MCC of 0.84.

### 2.2.2 Random forests

A random forest is an ensemble method, which refers to constructing a model by combining multiple instances of a base model [30]. In random forests, depending on the prediction task, these base models are either simple classification trees (RFC) or regression trees (RFR). When training a random forests model, each tree trains on a random sample of the dataset, and for each split of a node in the tree, only a random subset of all features is considered [30]. The final prediction consists of the aggregated output of all trees, which means for a regression task it is the average of the predictions, and for a classification task, it is the class predicted by the majority of trees.

In a study by Dybowski et al. [31], random forests were used for predicting Bevirimat resistance in HIV-1. Random forests together with a combination of sequence-based and structural descriptors achieved competitive results for their p2 peptide sequences. Furthermore, as shown by Bhadra et al. [32] random forests also work well for classifying antimicrobial peptides which resulted in an MCC of 0.9. Significant results have also been achieved for protein sequences. For classifying protein-protein interactions from primary protein sequences, random forests achieved better than state-of-the-art predictors in a study made by You et al. [33].

### 2.2.3 Hyperparameter tuning

Models consist of two kinds of parameters: general parameters and hyperparameters. Parameters are determined by the training of the model, while hyperparameters refer to the structure of the model such as the number of trees or depth of trees in the random forests model. The process of finding the optimal model architecture is known as hyperparameter tuning. Tuning hyperparameters contain several steps; defining a model, defining all possible values for hyperparameters, defining a method for "searching" the hyperparameter space for the optimum values, and defining an evaluative metric to evaluate the model. Two methods for "searching" are grid search and random search. Grid search is the basic method of tuning hyperparameters and creates a model for all possible combinations of the provided value set [34]. This method selects the model which produces the best result by considering the defined evaluation metric. Random search is different from grid search in the way of providing possible hyperparameter values. In grid search, a discrete set of values is provided but in random search, a statistical distribution is given and the values are sampled randomly from it [34].

As mentioned, different models have different hyperparameters. For example, hyperparameters to tune in SVC include kernel, regularization parameter (defined "C" in `sklearn.svm.SVC` [35]), and gamma. Choosing the right kernel in SVC is important as the wrong transformation leads to poor results. The regularization parameter defines how much the model avoids misclassification [24]. A higher value means less misclassification which leads to a smaller margin. A lower value means a wider margin but more misclassified samples. However, increasing the value of the parameter does not mean the model is more precise as it increases the risk of overfitting [24].

Gamma is also an important hyperparameter, which modulates at which distance a single sample has influence [35]. Models with high gamma only consider samples close to the margin as support vectors, and models with lower gamma allow samples further away as support vectors. Common hyperparameters in random forest are "max-depth", "max-leaf-nodes", "min-samples-leaf", "n-estimators", "bootstrap" and "max-features". "max-depth" defines the longest path between root and node and limits the depth of each tree in random forests [36]. "max-leaf-nodes" specifies the maximum number of terminal nodes. If the number of terminal nodes is reached after splitting a node, the tree will terminate and not grow further [36]. "min-samples-leaf" sets a condition on the minimum number of samples presented in the leaf node after splitting a node. If the number of samples in the leaf node is equal to or bigger than the specified minimum number, that node is considered a leaf or terminal node. However, if the number of samples is less than specified, the parent of that node is considered a leaf node. "n-estimators" define the number of trees to use in the forest before making the final prediction of the aggregated output of all trees, and is set to an integer value above 0 [35]. "bootstrap" specifies if bootstrap samples of the dataset should be used for the decision trees, or to use the whole dataset [35]. The parameter can either be set to "true" or "false", where "true" means to use samples, and "false" to use the whole dataset. "max-features" specifies the maximum number of features to consider when splitting nodes in each tree. It can either be set to an integer or one of {`sqrt, log2`}, where `sqrt` is the square root of the number of features in the dataset and `log2` the logarithm of the number of features to the base 2 [35].

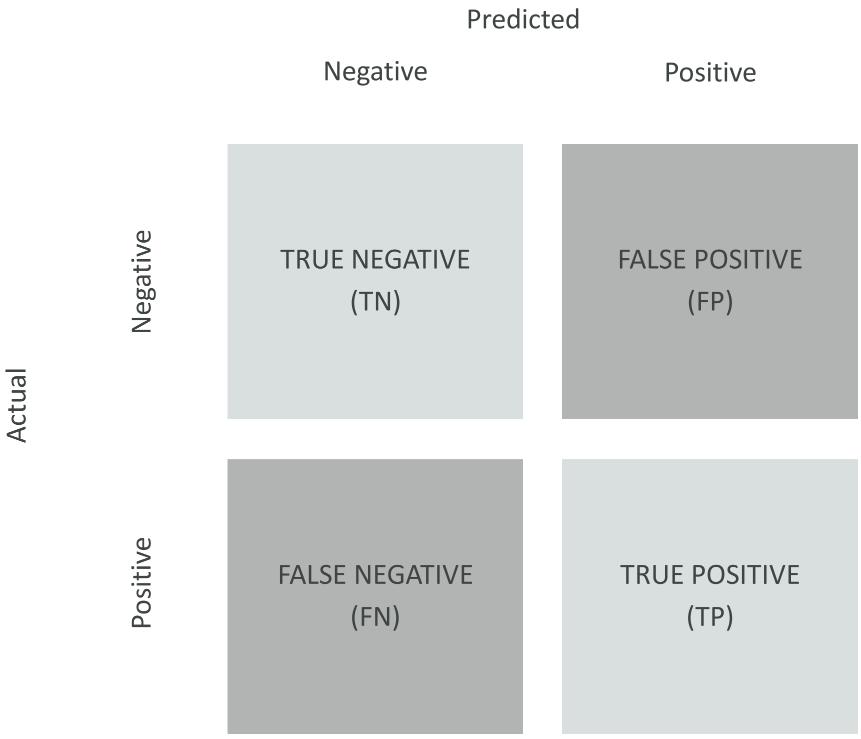
## 2.3 Evaluation metrics

Evaluation metrics are used to explain the performance of predictive models. Generally, they are generated by first feeding an unused test set to the trained model and then comparing the predicted outcome against the actual value. In classification tasks, the predicted class is compared to the actual class, and the outcomes can be represented in four different labels: True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN). These labels can be represented in a table called confusion matrix which is shown in Figure 2.4.

TP and TN mean that the model predicts the actual correct positive or negative class, respectively. FP means the model predicted a positive class for the sample, but the actual class was negative, so the prediction was false/wrong. A similar situation is given by FN, where the model predicts a negative class for the sample but the actual class of that sample is positive.

Different metrics can be used for evaluating classification models. The choice of evaluation metrics can be a challenging task, based on the given problem and the need of stakeholders. However, there are standard metrics that are widely used for evaluating classification models, such as accuracy, precision, recall, $F_1$, and MCC.

Accuracy is a standard metric that is commonly used for most problems. It shows the proportion of the number of correct predictions to the total number of predictions, as



| | | Predicted | |
|---|---|---|---|
| | | Negative | Positive |
| Actual | Negative | TRUE NEGATIVE<br>(TN) | FALSE POSITIVE<br>(FP) |
| | Positive | FALSE NEGATIVE<br>(FN) | TRUE POSITIVE<br>(TP) |

**Figure 2.4:** Confusion matrix.

seen in Equation 2.8. For this metric, a score of 1 represents a perfect performance, and 0 represents that the model performs poorly. Accuracy is defined as

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}. \tag{2.8}$$

Precision, recall, and $F_1$ are three suitable metrics for imbalanced datasets when the positive class is the minority class. Precision shows how many of the samples predicted as positive (TP+FP) are correctly predicted, see Equation 2.9. A maximum score of 1 is achieved when all positive predictions are correctly classified. A score of 0 means that all positive predictions are wrong. Precision is formulated as

$$Precision = \frac{TP}{TP + FP}. \tag{2.9}$$

Recall shows how many of the samples belonging to the actual positive class (TP+FN) that are correctly classified, see Equation 2.10. For recall, a score of 1 means that all samples with actual positive class are predicted correctly. A score of 0 means that the model couldn't assign any of the actual positive samples to the positive class. Recall is defined as

$$Recall = \frac{TP}{TP + FN}. \tag{2.10}$$

$F_1$-score is the average of precision and recall which is formulated as the harmonic mean as in Equation 2.11. It gives equal weight to both equations and the score ranges in $[0, 1]$. A score of 1 means a perfect prediction and is reached when $FN = FP = 0$, and the lowest score of 0 is reached when none of the positive samples has been classified correctly [37]. The formula for calculating $F_1$ is

$$F_1 = \frac{2 \times Precision \times Recall}{Precision + Recall}. \tag{2.11}$$

MCC is an evaluation metric for binary classification. A high MCC is only possible when the model predicts the majority of data samples for both classes correctly [37]. A model with perfect fit results in a score of 1, a score of 0 means that the predictions are random, and -1 indicate that the model performs poorly [37]. The formula can be seen in Equation 2.12. The main difference between MCC and $F_1$, is that $F_1$ is independent from TN [37]. MCC is formulated as

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP) \times (TP + FN) \times (TN + FP) \times (TN + FN)}}. \tag{2.12}$$

# 3
# Methods

In this chapter, the methods used to predict the potency of peptides are presented. First, a detailed introduction to the datasets is given. Since publicly available datasets are used, descriptions of how the experiments were conducted and how the data were obtained will be presented. Then, the implementation of the peptide representations is presented, followed by a detailed explanation of the machine learning models carried out in this project. Lastly, the evaluation process of the models will be presented.

## 3.1 Data

In this project, the modeling has been conducted on three different publicly available datasets. Each dataset has peptides with different characteristics, which is of interest to see how well the different descriptors work on different types of peptides. The datasets have been pre-processed so that all of them have a similar format of [peptide sequence, class]. How each dataset has been processed can be found in the following sections. After pre-processing, all datasets were split so that 80% were kept for training and validating the models, and 20% was kept as test data, which the final model was evaluated on in the final steps of this thesis. The splitting was done by using `train_test_split` from scikit-learn, with `random_state=100` [35].

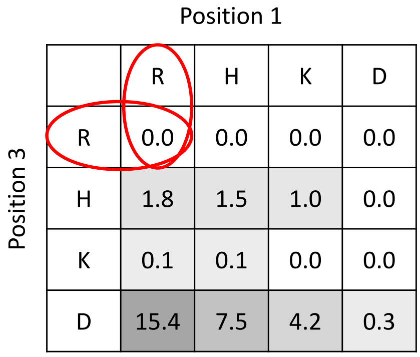
### 3.1.1 Dataset A

Dataset A is provided by van Rosmalen et al. [38]. This data contains the wild-type sequence "CQFDLSTRRLKC". Wild-type sequence refers to the sequence in its natural form, without any mutation. It is a cyclic peptide with ten amino acids, which are cyclized by a disulfide bridge. The measurement of the data is affinity through robust enrichment ratio, which can be regarded as the true enrichment ratio's lower 95% confidence limit, assuming the sampling is unbiased. True enrichment ratio can be explained as a measurement that describes to which degree a variant contributes to a selected property [38]. In the study by van Rosmalen et al. [38], the enrichment ratio indicates a positive or negative effect of the property binding affinity. It is calculated by measuring the frequency of a variant in the selected population and dividing it by the frequency of the variant in the unselected population. The robust enrichment ratio is used to minimize the effect of sampling errors [38].

From the paper, five different matrices are provided, each visualizing amino acid substitutions at two positions in the wild-type sequence, together with the corresponding robust enrichment score. In each matrix, the first position is shown along the horizontal axis and the second position along the vertical axis. What this means is that at these two positions in the wild-type sequence, the original two amino acids are substituted with the other 20 natural amino acids.

In Figure 3.1, an example from a matrix is shown. The example only shows the first five natural amino acids along the vertical and horizontal axis. The horizontal axis corresponds to position 1, and the vertical axis to position 3. This means that in the wild-type sequence "CQFDLSTRRLKC", both position 1 (Q), and position 3 (D) are substituted with the five natural amino acids. The variants are saved from the matrix as "Q1R, D3R" for the first row and column in the matrix, together with the corresponding robust enrichment score (0.0). The resulting variant therefore becomes "CRFRLSTRRLKC" with a robust enrichment ratio of 0.0.

A dataset is created where the positions in the wild-type sequence are substituted with the variant amino acids. The other matrices are similar to Figure 3.1 but have substitutions at positions 1 and 5, 1 and 6, 3 and 5, and 5 and 6. From the matrices and the wild-type sequence, a total of 2000 sequences were obtained. However, a total of 32 sequences did not have an available score, since these combinations were counted less than 10 times in the unselected library and were therefore removed. When combining the matrices into one dataset, each matrix was first normalized such that the score of the wild-type sequence equaled one, i.e., all scores in each matrix were divided by the score of the wild-type sequence in their matrix. Then 77 duplicate sequences, with a total of 193 appearances, were removed, and a dataset with 1775 sequences remained. These sequences had a score $s \in [0, 49.5]$. A score of 1 (the score of the wild-type sequence) was used as the threshold when adding class labels to the sequences. Every score equal to this threshold or less was classified as the negative class and everything above was classified as the positive class. The distribution between classes for the training and validation set can be seen in Table 3.1.



Position 1

Position 3

| | R | H | K | D |
|---|---|---|---|---|
| R | 0.0 | 0.0 | 0.0 | 0.0 |
| H | 1.8 | 1.5 | 1.0 | 0.0 |
| K | 0.1 | 0.1 | 0.0 | 0.0 |
| D | 15.4 | 7.5 | 4.2 | 0.3 |

**Figure 3.1:** Matrix of robust enrichment ratio of double amino acid substitutions at positions 1 and 3, for the first five natural amino acids.

According to Wang et al. [39], cyclic peptides have shown superior behavior in biological activities to linear peptides and have a large potential as therapeutic agents. It is therefore of interest to analyze how a descriptor and model combination for cyclic peptides might differ from the combination of linear peptides.

| Dataset | Set | Nr. of negatives | Nr. of positives | Total |
|---|---|---|---|---|
| A | Training | 1 117 (87%) | 161 (13%) | 1 278 |
| | Validation | 115 (81%) | 27 (19%) | 142 |
| B | Training | 11 272 (87%) | 1 673 (13%) | 12 945 |
| | Validation | 1 261 (88%) | 178 (12%) | 1 439 |
| C | Training | 140 (30%) | 320 (70%) | 460 |
| | Validation | 15 (29%) | 37 (71%) | 52 |

**Table 3.1:** Class sizes within each dataset.

### 3.1.2 Dataset B

Dataset B consists of a wild-type sequence together with a dataset of variants and corresponding scores [12, 40]. The measurement of the dataset is $\beta$-glucosidic linkages through scores calculated by Enrich2 tool [41]. One up to five variants are provided per measured score, in the format "A104E" where the first letter represents the amino acid in the wild-type sequence (A) at a position (104) and the substituting amino acid (E), similar to the variants in dataset A. From these, and including the wild-type sequence, a total of 26 653 sequences with a corresponding score $s \in [-4.72, 3.79]$ were obtained. 916 of these sequences contained a '*'-symbol, which were removed, resulting in a dataset of 25 737 data points. When using the Enrich2 tool, a score of 0 corresponds to the score of the wild-type sequence. Therefore, when adding class labels to the sequences, sequences with a score equal or less than 0 were classified as the negative class and everything above was classified as the positive class.

The sequences are representations of longer protein sequences, consisting of 501 amino acids, compared to the peptides in dataset A, which has a length of 12 amino acids. Using two such different molecules is relevant to see how descriptors and models differ for long and short sequences. To be able to compare the results between dataset A and dataset B, the protein dataset was downsampled so that the distribution between the two classes was the same as for dataset A. This was done by removing 7 756 sequences with positive class by using the function `resample` from scikit-learn module `utils` [35]. When implemented, `random_state` is set to 42, and since a sample from the class should not appear more than once, the parameter `replace` is set to "false". The sample chosen from `resample` is then combined together with the negative class. The final dataset consists of 17 981 sequences, including the test set, and the distribution of the training and validation set can be seen in Table 3.1.

In the study by Gelman et al. [12], the dataset was used to evaluate sequence-function mapping, i.e., predict the functions of unseen sequences, by using supervised learning methods. Specifically, by using four different supervised neural network architectures, they showed that the dataset could be used to design new sequence variants. The models resulted in a Pearson's correlation coefficient above 0.5 for all four architectures, where a score of 1 represents a perfect linear relationship between true and predicted scores, -1 a negative linear relationship, and 0 represents no linear relationship.

### 3.1.3 Dataset C

Dataset C was downloaded from the Immune Epitope Database (IEDB) [42]. This database includes over one million peptides which are possible to filter and search among. Possible filters include what type of peptides, sequence length, the natural source where the peptide can be found, what type of experiment is performed and the outcome, the type of host mounting the immune response, and what type of disease affecting the host. For this project, only linear peptides found in humans with a length of 9 amino acids are considered. Of these, only peptides that are experimentally tested for binding to major histocompatibility complex (MHC) ligand 'HLA-A*02:01', with both positive and negative outcomes are chosen. Further, only humans as hosts with any disease state are selected. After the dataset had been downloaded from the database, only the peptides with the assay unit nanomolar (nM) concentration were kept to only keep those peptides that have the same end-point measurement. After removing duplicate sequences, the dataset consisted of 641 peptides including the test set, of which 441 had a positive outcome and 200 had a negative outcome. The distribution of the training and validation set can be seen in Table 3.1.

Dataset C differentiates from the other two datasets as the sequences don't originate from a wild-type sequence. This is of interest to see how sequences with very different compositions impact the results of the descriptor and model combinations, compared to the sequences in datasets A and B, which have a similar amino acid composition. Furthermore, since IEDB is a widely-used public database including more than one million peptides, it has been used to compile data for a wide range of studies using machine learning. Data from this database have for example been used for predicting T-cell epitopes in order to develop a vaccine against Zika Virus [43], predicting human leukocyte antigen (HLA) class II molecules from which results can be utilized when developing a vaccine or cancer treatment [44], and predicting B-cell epitopes which, according to Liu et al. [45], are important in vaccine design, clinical diagnosis, and antibody production.

## 3.2 Molecular representation

In order to use the sequences in predictive machine learning models, the information from each sequence must be translated into numerical representations. In this project, three different descriptors are generated for each dataset, PseAAC, Z-scales, and one-hot representation. Before using the generated representations in the machine learning models, the representation for each dataset is normalized using the function `robust scaler` in scikit-learn [35]. This function centers the data by removing the median and then scales it according to the quantile range. The step of normalization is to make the data more consistent as it ensures that all features have roughly the same scale, and no feature has a very large value compared to the other features [27].

### 3.2.1 Pseudo amino acid composition

As mentioned in section 2.1.3, PseAAC has previously been applied in various prediction tasks, which makes it an interesting candidate for this project as well. In this project, to generate PseAAC, the Python package `propy` is used to generate the PseAAC [46]. In PseAAC there are two factors for the user to select, $\lambda$ and the weight. $\lambda \in [0, N]$ is a factor reflecting the rank of correlation, where $N$ is the length of the sequence, and 0 corresponds to the components for conventional AAC. The sequence-order effect incorporated in the representation increases by choosing a greater value for $\lambda$ [11]. This parameter was tuned by trying different values for $\lambda$ on a subsample of each dataset together with each model. For each dataset, a maximum of seven different values were evaluated, which were evenly distributed in the range $\in [0, N]$. The $\lambda$s with the highest $F_1$-score for each dataset and model were then selected for the modeling. The results for each dataset can be seen in Figure 3.2 and the optimal values for each combination can be found in Table 3.2. The weight factor is a value in range $\in [0.05, 0.7]$, and is used to put weight to the additional components ($20+1$ to $20+\lambda$) with respect to the conventional AAC [47]. In this thesis, the weight factor is set to the default value of 0.05, which is the same value as Chou [11] proposed when introducing PseAAC.

| Dataset | Model | Optimal $\lambda$ |
|---|---|---|
| Dataset A | SVC | 9 |
| | Random forests | 5 |
| Dataset B | SVC | 1 |
| | Random forests | 1 |
| Dataset C | SVC | 5 |
| | Random forests | 1 |

**Table 3.2:** Optimal $\lambda$ for each dataset and model.

### 3.2.2 Z-scales

In the study by van Westen et al. [16] mentioned in section 2.1.2, Z-scales performed well for various types of datasets. This makes it a relevant descriptor to explore in this thesis, to see if a similar conclusion can be drawn or if a particular predictive model might be more suitable for this descriptor. In this thesis, Z-scales are generated by using the `Peptides` package in Python [48]. So for each amino acid in the sequences, five z-values are calculated which results in a vector of 60 features for each sequence in dataset A, 2 505 features for the sequences in dataset B, and 45 features for the sequences in dataset C.

### 3.2.3 One-hot representation

One-hot representation doesn't capture any prior domain knowledge about amino acids [49], compared to PseAAC and Z-scales. The descriptor only contains information about the positions of the 20 natural amino acids. This makes it an interesting descriptor for this project in order to analyze if the manually curated features of PseAAC and Z-scales impact the predictions, or if the order of the amino acids in the sequence is enough. To generate a one-hot representation, each amino acid in the sequences is first encoded into integers. Then, for each position in the sequence, a list with a length of 20 is created with all zeros. At the position of the integer representing the amino acid, the zero is set to 1. These are inserted in a matrix $\in \mathbf{R}^{L\times 20}$ where $L$ is the length of the sequence. Then, this matrix is flattened into a vector, with length $L \times 20$. The number of features for dataset A is therefore 240, for dataset B 10 020, and dataset C 180.

### 3.2.4 Feature exploration

After generating the descriptors, an exploration of features was conducted. This was done for Z-scales and PseAAC. Since one-hot representation is a direct representation of the sequence, where the features don't include any prior domain knowledge about amino acids, this descriptor was not explored further before using it in the machine learning models. For each descriptor and dataset, first, the distribution of the features between the two classes is analyzed, and thereafter, the SHAP values for the features are presented. For the implementation of SHAP, only random forests are considered and therefore the "TreeExplainer" is chosen as the SHAP explainer. Summary plots of SHAP values give an overall picture of the features' contributions. The y-axis is the features in descending order, and the x-axis shows the calculated mean absolute value of SHAP values, where the red color shows the positive correlation with the target, and the blue the negative correlation [22]. The results of the feature exploration can be found in section 4.1.

## 3.3 Machine learning models

The numerical representations are then used for two different machine learning models: SVC and random forests. Each model is implemented with hyperparameter tuning, where `Gridsearch` from scikit-learn is used to find the optimal values of hyperparameters [35]. The hyperparameters which are tuned and the parameter values can be seen in Table 3.3. The procedure of defining the possible values for the hyperparameters was done by observing that no optimal values were the same as the border value. If an optimized model chose a value corresponding to a border, the borders were extended and the model was optimized once again. When fitting a model, Stratified K-Folds cross-validator with five folds is used, and the combination with the highest MCC is retained. MCC is used as the evaluation metric as it takes correct predictions for both classes into account. The optimal values of the hyperparameters for the model and descriptor combinations for each dataset can be seen in Table 3.4. The models are trained on 90% of the training data and the remaining 10 % is used as a validation set.

| Model | Hyperparameter | Values |
|---|---|---|
| SVC | Kernel | [linear, sigmoid, rbf] |
| | C | [0.0001, 0.1, 1, 10, 20, 50, 100] |
| | Gamma | [scale, auto] |
| Random forests | Bootstrap | [True, False] |
| | Max-depth | [10, 50, 100, 500, 1000] |
| | Max-features | [auto, sqrt] |
| | Min-samples-leaf | [1, 2, 10, 50] |
| | n-estimators | [1, 10, 100, 500, 1000, 1500] |

**Table 3.3:** Hyperparameters tuned for each model, and the input values.

### 3.3.1 Support vector machines

In the study by Mohabatkar et al. [28] mentioned in section 2.2.1, SVM together with PseAAC worked very well for long protein sequences, similar to the length of proteins in dataset B. Considering this, and that SVM also showed a good result classifying peptides in the study made by Lata et al. [29], it is a relevant model to explore further to see if it also performs well for other types of peptides and proteins.

In this thesis, SVM is implemented by using `SVC` from the scikit-learn module `svm` [35]. As all three datasets are imbalanced, when implementing SVC, the hyperparameter `class_weight` is set to "balanced". This is to assign different weights inversely proportional to the class frequencies to the data points [35].

### 3.3.2 Random forests

As mentioned in section 2.2.2, random forests have shown to be a suitable model in various studies for predicting different proteins and peptides. With these positive results, it is a good model to explore further in this thesis.

The implementation of random forest is done by using `RandomForestClassifier` from the `ensemble` module in scikit-learn [35]. When implementing random forests in this project, the value of the parameter `random_state` is set to 2 to control the reproducibility of the results.

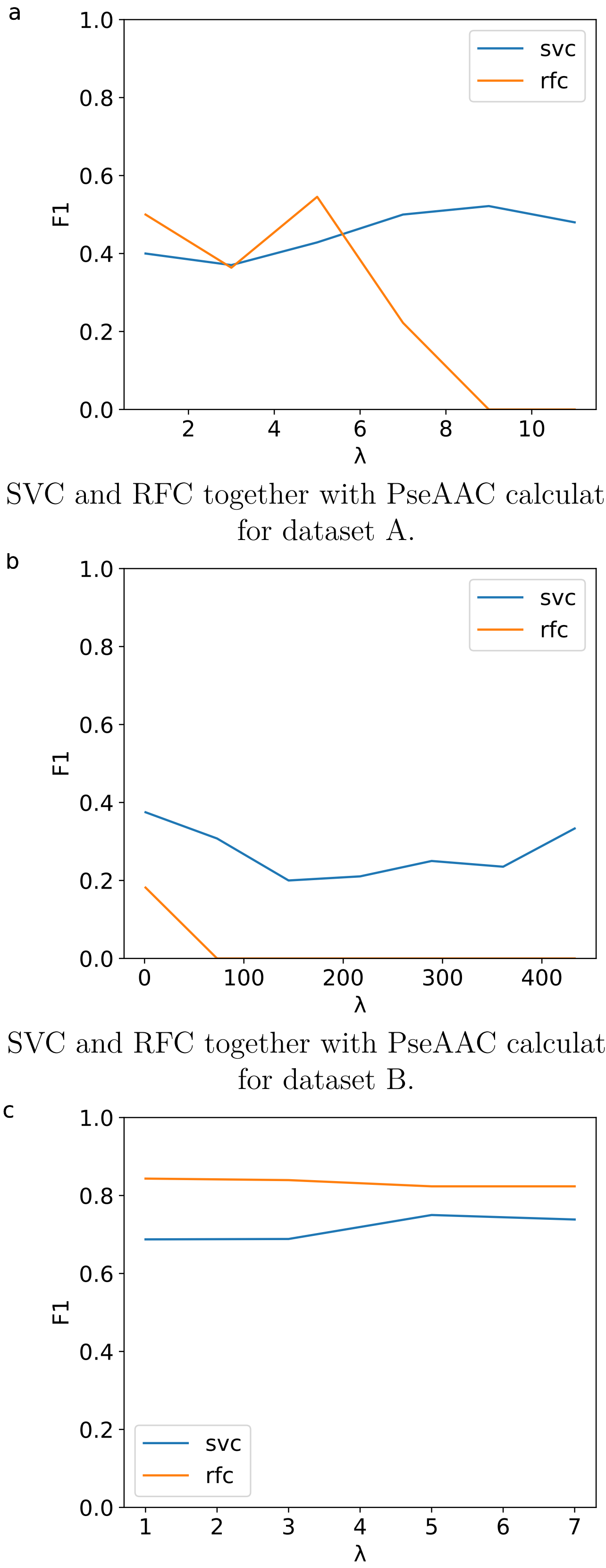
| Dataset | Model | Hyperparameter | Z-scales | PseAAC | One-hot |
|---|---|---|---|---|---|
| A | SVC | Kernel | rbf | linear | rbf |
| | | C | 20 | 100 | 10 |
| | | Gamma | scale | scale | scale |
| | Random forests | Bootstrap | False | False | False |
| | | Max-depth | 50 | 50 | 100 |
| | | Max-features | auto | auto | auto |
| | | Min-samples-leaf | 1 | 1 | 1 |
| | | n-estimators | 100 | 1000 | 100 |
| B | SVC | Kernel | linear | rbf | rbf |
| | | C | 10 | 0.1 | 0.1 |
| | | Gamma | scale | auto | scale |
| | Random forests | Bootstrap | True | True | True |
| | | Max-depth | 500 | 100 | 500 |
| | | Max-features | auto | auto | auto |
| | | Min-samples-leaf | 1 | 1 | 1 |
| | | n-estimators | 1 | 1 | 1 |
| C | SVC | Kernel | rbf | rbf | rbf |
| | | C | 1 | 10 | 10 |
| | | Gamma | scale | scale | auto |
| | Random forests | Bootstrap | True | True | False |
| | | Max-depth | 10 | 10 | 50 |
| | | Max-features | auto | auto | auto |
| | | Min-samples-leaf | 1 | 1 | 1 |
| | | n-estimators | 500 | 500 | 1000 |

**Table 3.4:** Optimal values for each hyperparameter for each model, dataset, and descriptor.

## 3.4 Evaluation

When evaluating the performance of the descriptor and model combinations, five different metrics are analyzed: accuracy, precision, recall, MCC, and $F_1$. Only looking at accuracy could be misleading when having an imbalanced dataset. As 87% of datasets A and B are of the negative class, the accuracy would result in 0.87 if the model only predicts the majority class. Therefore, it is important to consider other metrics as well. Since a negative peptide being classified as positive (FP) can become very costly in future steps of the drug discovery process, high precision is necessary. High precision makes it more likely that a sample predicted as positive is correct. Moreover, high recall is important to avoid missing potential drugable molecules. Since both precision and recall are of importance, $F_1$ is a suitable metric for this project, especially for datasets A and B which have the negative class as the majority class. High MCC can be interpreted as the model that can predict both classes correctly, which is suitable when both the positive and negative classes are important. This metric is therefore a suitable metric for binary classification and this project.

To calculate the metrics, `accuracy_score`, `precision_score`, `recall_score`, `matthews_corrcoef`, and `f1_score` are used from the `metrics`-module in scikit-learn [35].



**(a)** $F_1$ score of SVC and RFC together with PseAAC calculated with different $\lambda$ for dataset A.

**(b)** $F_1$ score of SVC and RFC together with PseAAC calculated with different $\lambda$ for dataset B.

**(c)** $F_1$ score of SVC and RFC together with PseAAC calculated with different $\lambda$ for dataset C.

**Figure 3.2:** Evaluating different $\lambda$ for (a) Dataset A, (b) Dataset B, and (c) Dataset C, to find the optimal value.

# 4

# Results

This chapter begins by first presenting the results from the exploration of the features generated by Z-scales and PseAAC. This is followed by a presentation of the results from the descriptor and model combinations which are evaluated by comparing the different evaluation metrics for each dataset. Lastly, the results of using the best combinations on each test set are presented and described.

## 4.1 Feature exploration

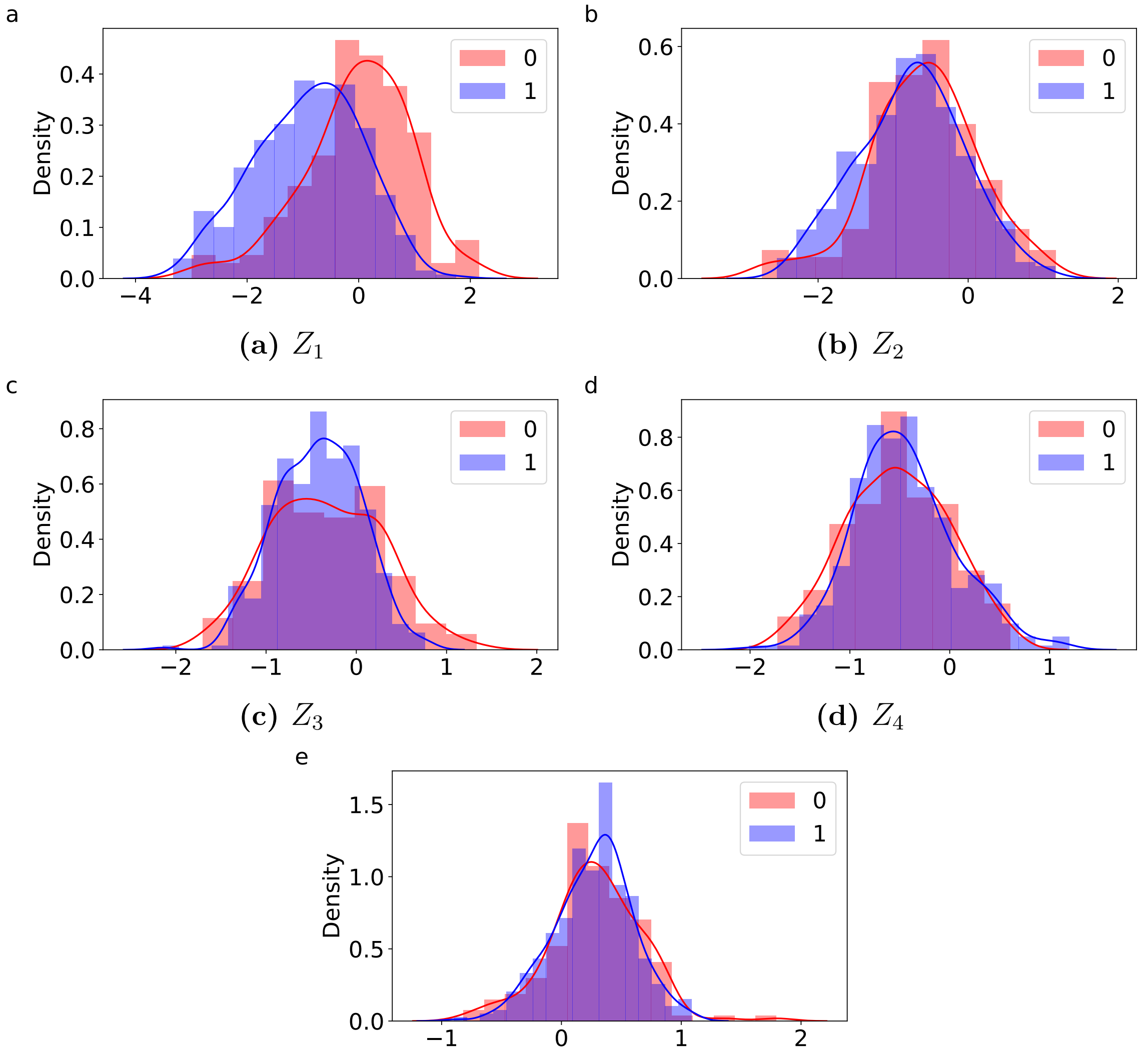
Most features showed a similar distribution for both the negative and the positive classes, which indicates that the features are not obvious identifiers when classifying new, unknown samples. Therefore, only the plots showing some indication of being differentiated and the corresponding summary plots of SHAP values are discussed, and the other plots can be found in Appendix A.

### 4.1.1 Z-scales

For Z-scales, the five average z-values were used to generate the feature exploration for each dataset. For dataset C, the distribution of the two classes for $Z_1$, corresponding to lipophilicity, are more separated, compared to the other z-values, as seen in Figure 4.1. This means that for the peptides in dataset C, the value of $Z_1$ can be an indicator of which class a sample belongs to. This is supported by the SHAP value in Figure 4.2. As seen in the figure, $Z_1$ is the feature that has the most impact on the model output, with a SHAP value above 0.3, compared to the other z-values which have a score around 0.1.

### 4.1.2 PseAAC

For PseAAC, only the features corresponding to the pseudo components are used in the distribution plots, i.e., the value of $\lambda$ for the different combinations. This is because the first 20 components corresponding to the conventional amino acid composition are only a direct representation of the sequence. Similar to Z-scales, most features show no significant difference in the distribution for both classes. However, for dataset A and SVC where optimal $\lambda = 9$, pseudo components 27, 28, and 29 show some differences between the distribution, see Figure 4.3. It can be



**(a)** $Z_1$

**(b)** $Z_2$

**(c)** $Z_3$

**(d)** $Z_4$

**(e)** $Z_5$

**Figure 4.1:** Distributions of the average z-values for each class in dataset C.

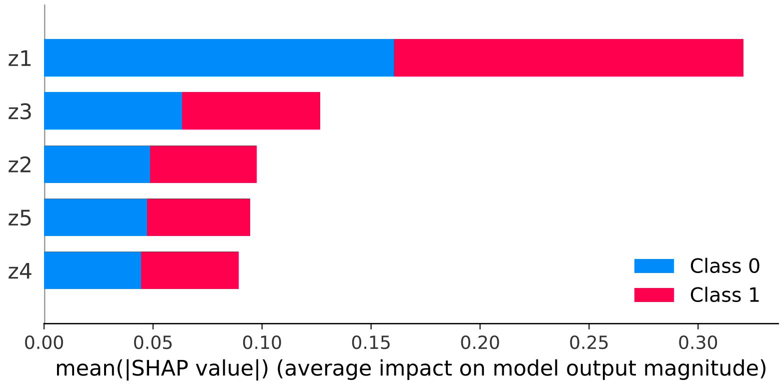
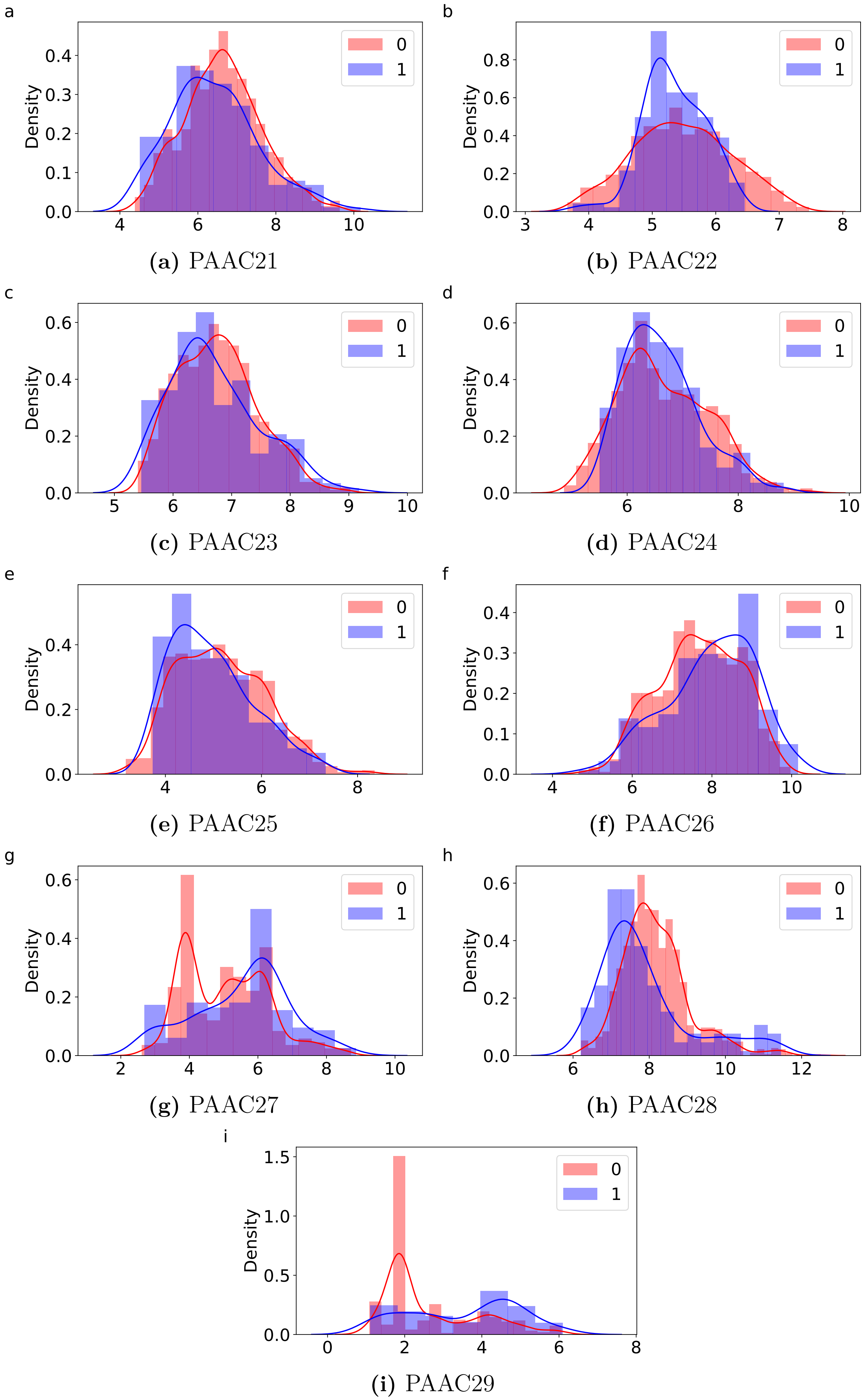


**Figure 4.2:** Summary plot for the average z-values in dataset C.

seen that a lower value for pseudo components 27 and 29, as well as a higher value for pseudo component 28, is more related to the negative class.



**(a)** PAAC21

**(b)** PAAC22

**(c)** PAAC23

**(d)** PAAC24

**(e)** PAAC25

**(f)** PAAC26

**(g)** PAAC27

**(h)** PAAC28

**(i)** PAAC29

**Figure 4.3:** Distributions of the pseudo components for each class in dataset A with optimum lambda for SVC.

## 4.2 Modeling

In this project, the comparison between the performance of each combination is made by considering five evaluation metrics: accuracy, recall, precision, $F_1$, and MCC. A comparison of all results are shown in Figure 4.4-4.8 and the most relevant confusion matrices are shown in Figure 4.9-4.13. An overview of all confusion matrices for each dataset, model and descriptor can be found in Appendix B.

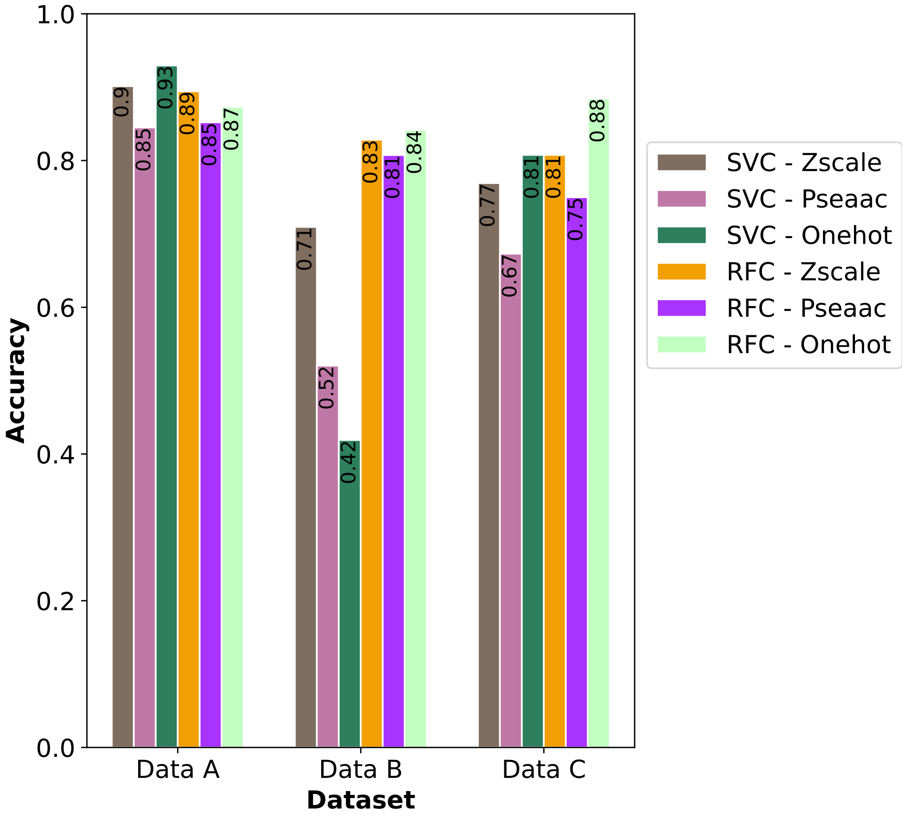


**Figure 4.4:** Accuracy of each combination of model and descriptor.

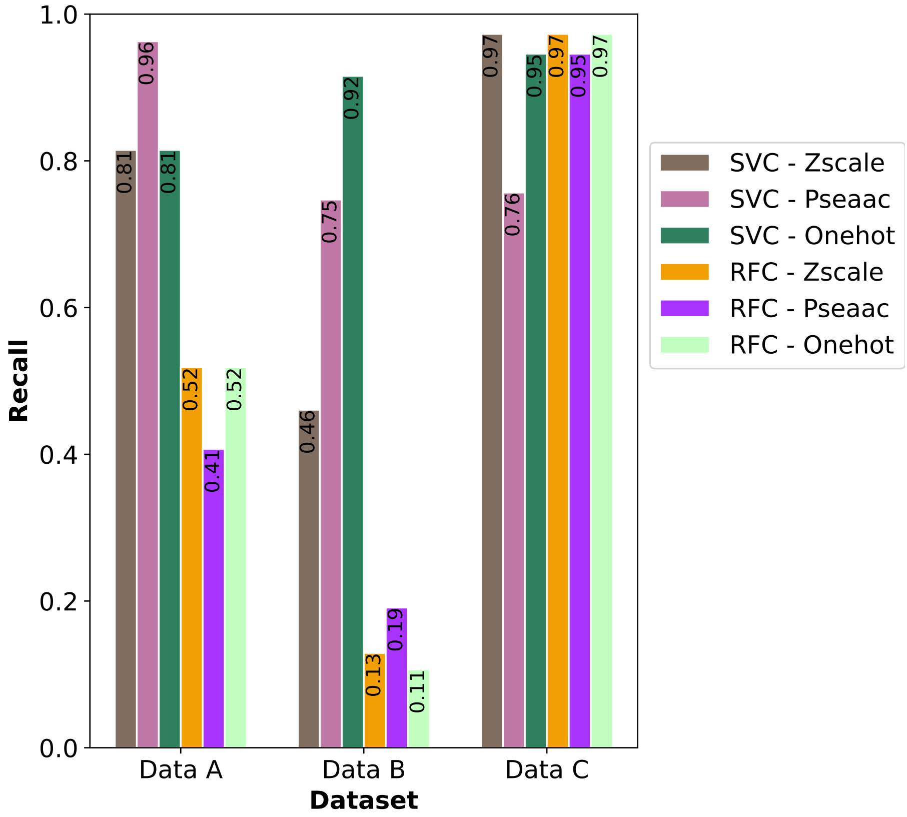


**Figure 4.5:** Recall of each combination of model and descriptor.

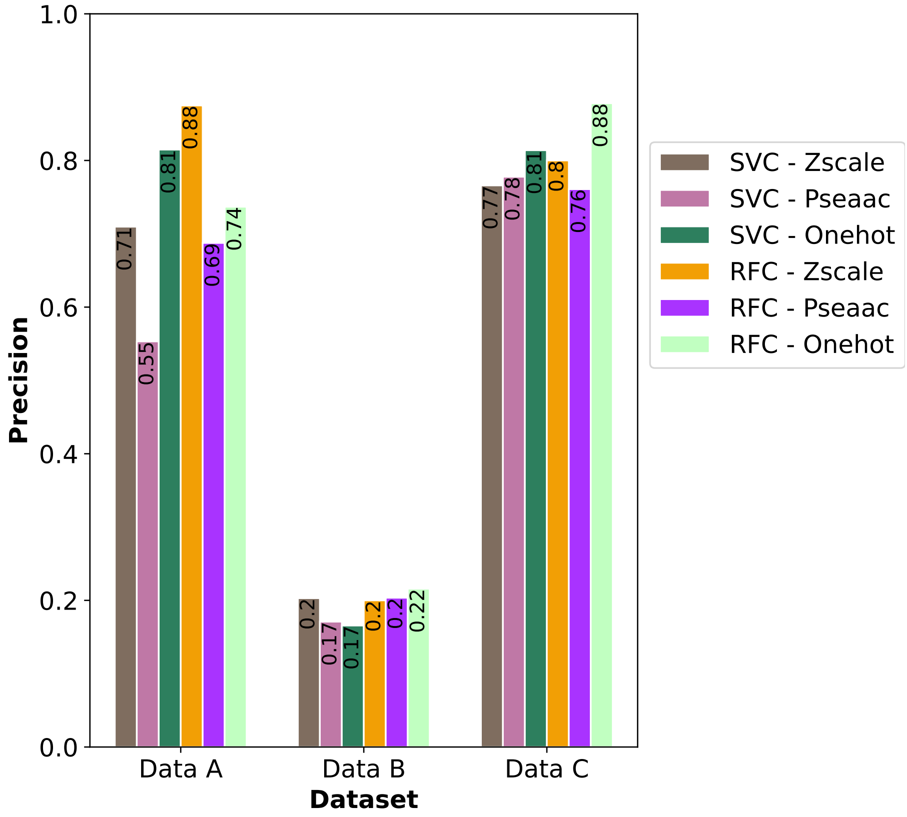


**Figure 4.6:** Precision of each different combination of model and descriptor.

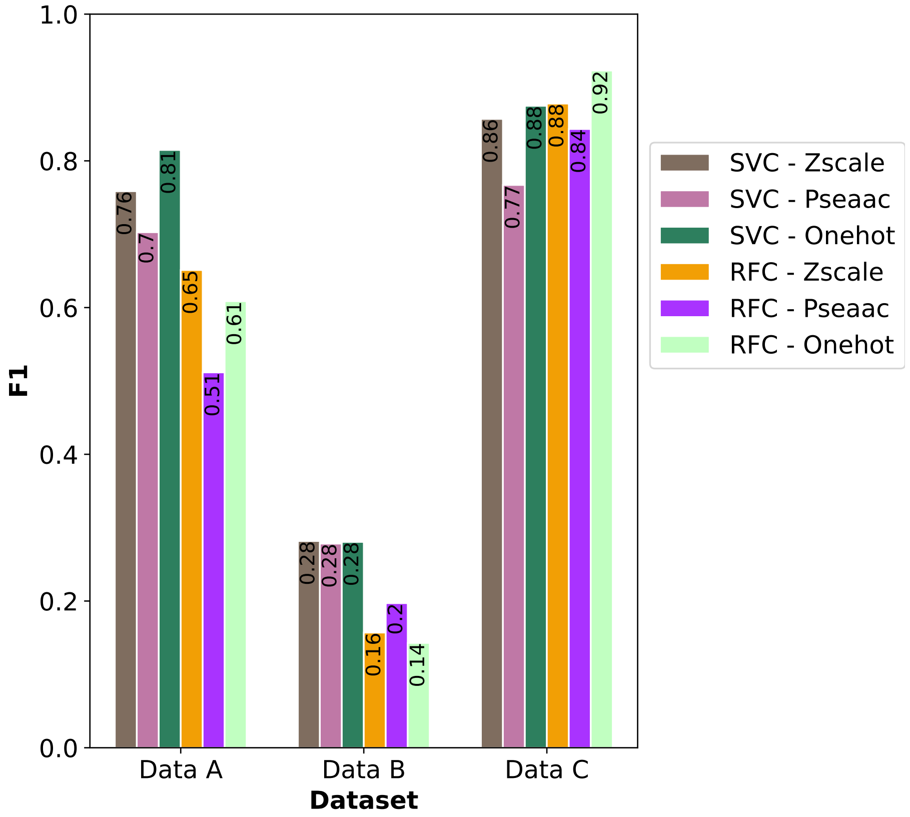


**Figure 4.7:** F1 of each combination of model and descriptor.

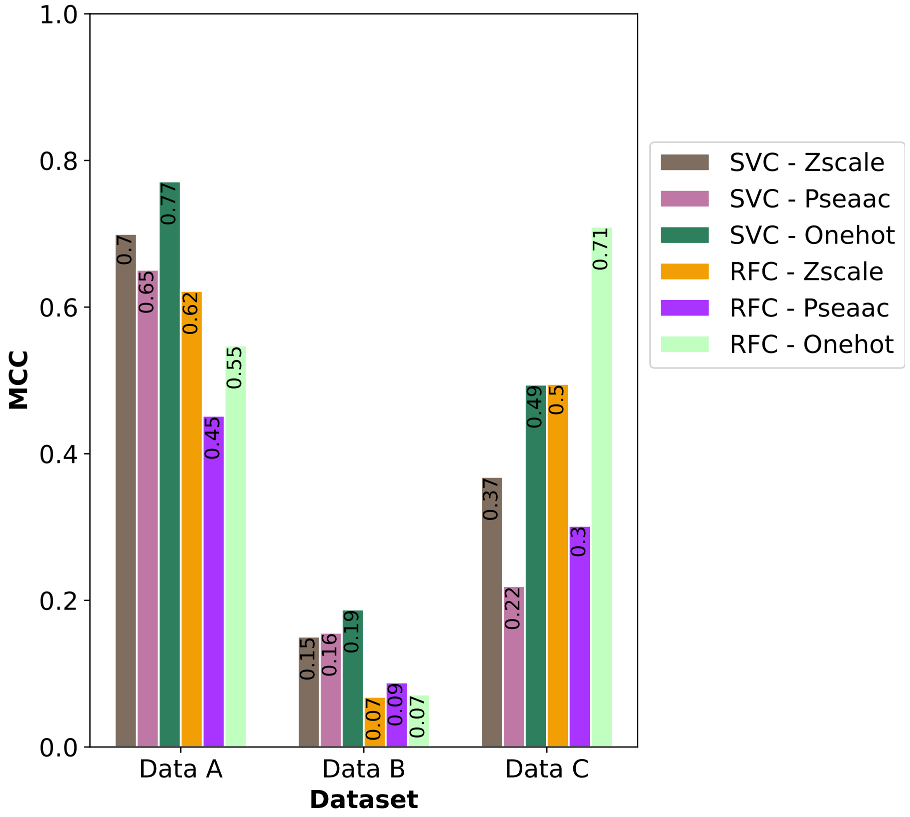


**Figure 4.8:** MCC of each combination of model and descriptor.

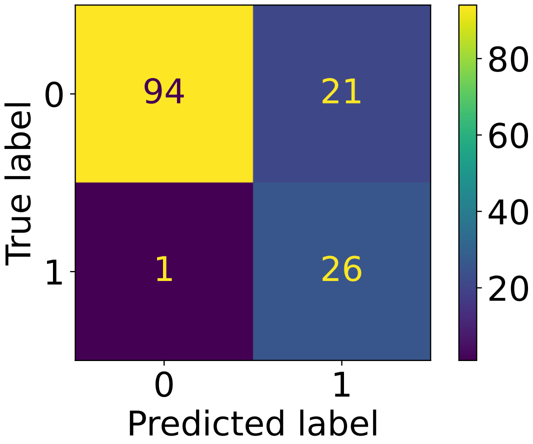


**Figure 4.9:** Confusion matrix for SVC and PseAAC for dataset A.

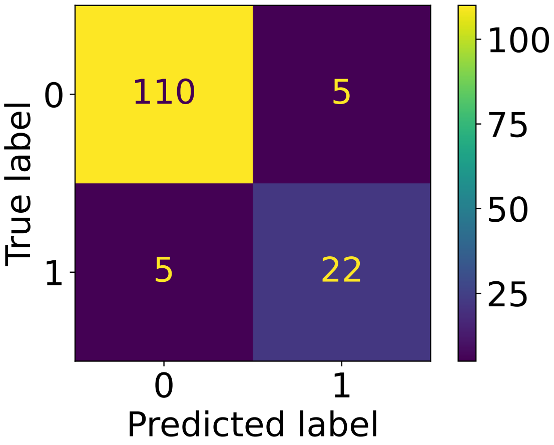


**Figure 4.10:** Confusion matrix for SVC and one-hot representation for dataset A.

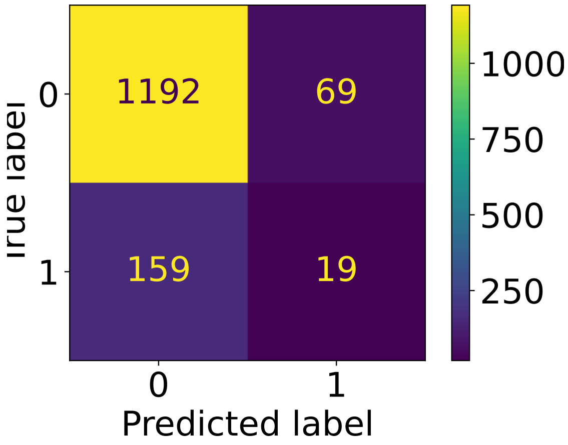


**Figure 4.11:** Confusion matrix for random forests and one-hot representation for dataset B.

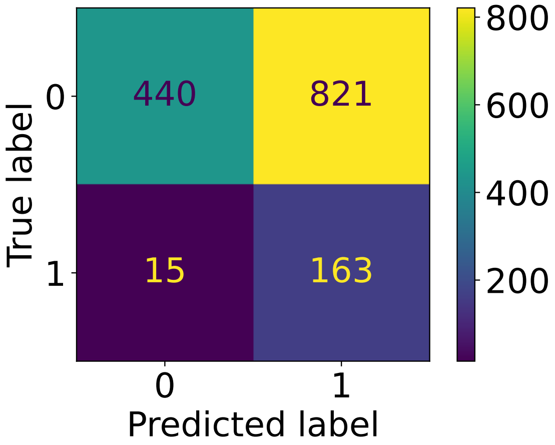


**Figure 4.12:** Confusion matrix for SVC and one-hot representation for dataset B.

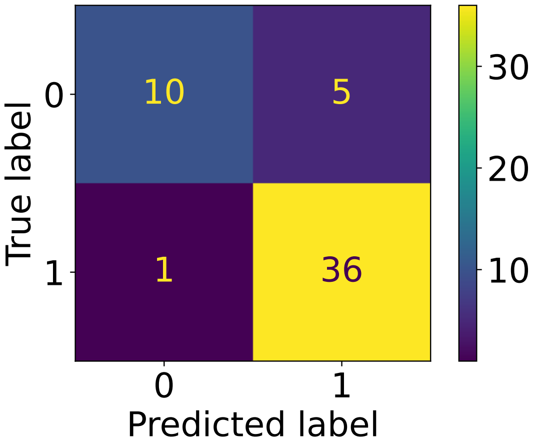


**Figure 4.13:** Confusion matrix for random forests and one-hot representation for dataset C.

### 4.2.1 Dataset A

For dataset A, each combination has an accuracy above 0.85, as seen in Figure 4.4, with SVC together with one-hot representation having the best score of 0.93. Since this dataset is very imbalanced with 87% and 81% of the training and the validation set, respectively, belonging to the negative class, the accuracy is not a suitable metric to evaluate the performance. The recall for dataset A in Figure 4.5 shows the best result for SVC together with PseAAC with a score of 0.96, followed by SVC together with Z-scales and one-hot representation, both with a score of 0.81. This means that these models are good at predicting the samples belonging to the positive class correctly. However, looking at the precision in Figure 4.6, SVC together with PseAAC has the lowest score of 0.55. This indicates that a lot of the model's positive predictions are false, which is supported by the confusion matrix in Figure 4.9, where almost half of the positive predictions are false. Low precision increases the risk of late-stage failure in drug discovery. Choosing the wrong candidate (FP) for further analysis leads to spending unnecessary costs and time. However, both SVC together with one-hot representation and random forests together with Z-scales show good precision scores above 0.8.

$F_1$ takes both precision and recall into account and that is why SVC together with one-hot representation, which had the second-best score for both these metrics, has the highest $F_1$ with a score of 0.81, as seen in Figure 4.7. This combination is also the one showing the best MCC score with a value of 0.77, as seen in Figure 4.8. This indicates that the model is relatively good at predicting both classes correctly, and by looking at the confusion matrix in Figure 4.10, only 10 samples, or 7% of the validation set, are predicted wrongly.

From the results, it is seen that random forests overall show lower scores than SVC, except for accuracy and precision. Even if precision is of importance when evaluating potential peptides for drug discovery, the score of 0.88 for random forests together with Z-scales is not significantly higher than the score of 0.81 for SVC and one-hot representation. Therefore, when taking all the metrics into consideration, SVC together with one-hot representation is chosen as the best combination for dataset A.

### 4.2.2 Dataset B

For dataset B, the combination of random forests and one-hot representation has the highest accuracy score of 0.84, closely followed by random forests together with both Z-scales and PseAAC, as seen in Figure 4.4. Even if accuracy gives a good overview of the performance, since it shows the proportions of correct predictions, it is not a reliable score for the evaluation of imbalanced datasets. Since dataset B is extremely imbalanced, similar to dataset A, with 87% consisting of the negative class in both the training and validation sets, the accuracy can be affected by the majority class. As shown in the confusion matrix in Figure 4.11, random forests together with one-hot representation, which had the highest accuracy, have a large number of TNs compared to the other combinations.

Recall score in Figure 4.5 shows that the combination of SVC and one-hot representation has the highest score of 0.92, which means that this combination could predict 92% of the samples actually belonging to the positive class correctly. The high recall score for this combination is because of the low number of FNs (15) and high number of TPs (163) as seen in Figure 4.12. As also shown in Figure 4.5, the combinations of SVC have significant higher scores compared to the combinations of random forests. This means that SVC could predict more samples actually belonging to the positive class correctly. One reason for this could be that the tuned value for hyperparameter C (regularization) for this model and dataset is 0.1, as seen in Table 3.4. Lower C leads to a larger margin and allows more misclassification. It increases the ability of the model to classify most of the samples belonging to the minority class (higher recall), but it also increases the number of FP which leads to a lower precision score, as seen in Figure 4.6. The precision of SVC together with one-hot representation shows that only 17% of the samples predicted as positive are correct, even if the combination could capture a majority of the positive class, as shown by the high recall score.

The highest score of $F_1$ presented in Figure 4.7 is 0.28, which is the same for all SVC combinations. The highest MCC score is 0.19 for SVC together with one-hot representation which implies this combination has better performance in both negative and positive classes compared to the other combinations. Even if SVC together with one-hot representation has the highest MCC, it has also the highest number of FPs which leads to the lowest precision. This is unfavorable when exploring new molecules as peptides without the expected properties will move on for further analysis. However, all combinations have a very low score of $F_1$ and MCC, indicating that the explored combinations work poorly for this dataset. Although random forests together with one-hot representation has the lowest MCC, it also has the lowest rate of FPs. This means that if the model predicts a molecule as positive it is more likely to be correct and decreases the risk of failure in later stages of the process. Therefore, to evaluate the performance of the unseen test set, this combination is chosen.

### 4.2.3 Dataset C

When looking at the accuracy for dataset C in Figure 4.4, random forests together with one-hot representation show the best result of 0.88, followed by SVC together with one-hot representation and random forests with Z-scales, both with a score of 0.81. However, the dataset is imbalanced with around 70% of the dataset consisting of samples belonging to the positive class, which makes accuracy a non-suitable score. However, all combinations show a very high recall, as seen in Figure 4.5. All combinations except SVC together with PseAAC have a score above 0.95. Also for precision, all combinations show good results, with scores above 0.76, with random forests together with one-hot representation having the best score of 0.88. The high values for both recall and precision also result in high $F_1$ scores. As seen in Figure 4.7, all combinations have a score above 0.77, again with random forests together with one-hot representation having the best result with a score of 0.92. However, when looking at the MCC in Figure 4.8, no combination except random forests together with one-hot representation has a score above 0.5. The lower scores of MCC and the high scores of precision and recall indicate that the models have problems predicting the negative class correctly. However, as random forests together with one-hot representation has the best MCC with a score of 0.71, and it has shown the highest scores consistently for all evaluation metrics, this is chosen as the best combination for dataset C. As seen in the confusion matrix in Figure 4.13, the model predicts almost all samples belonging to the positive class correctly, as expected. However, around 10% of the samples are predicted as positives but are actually negatives, which is the reason behind a lower MCC.

## 4.3 Evaluation on test set

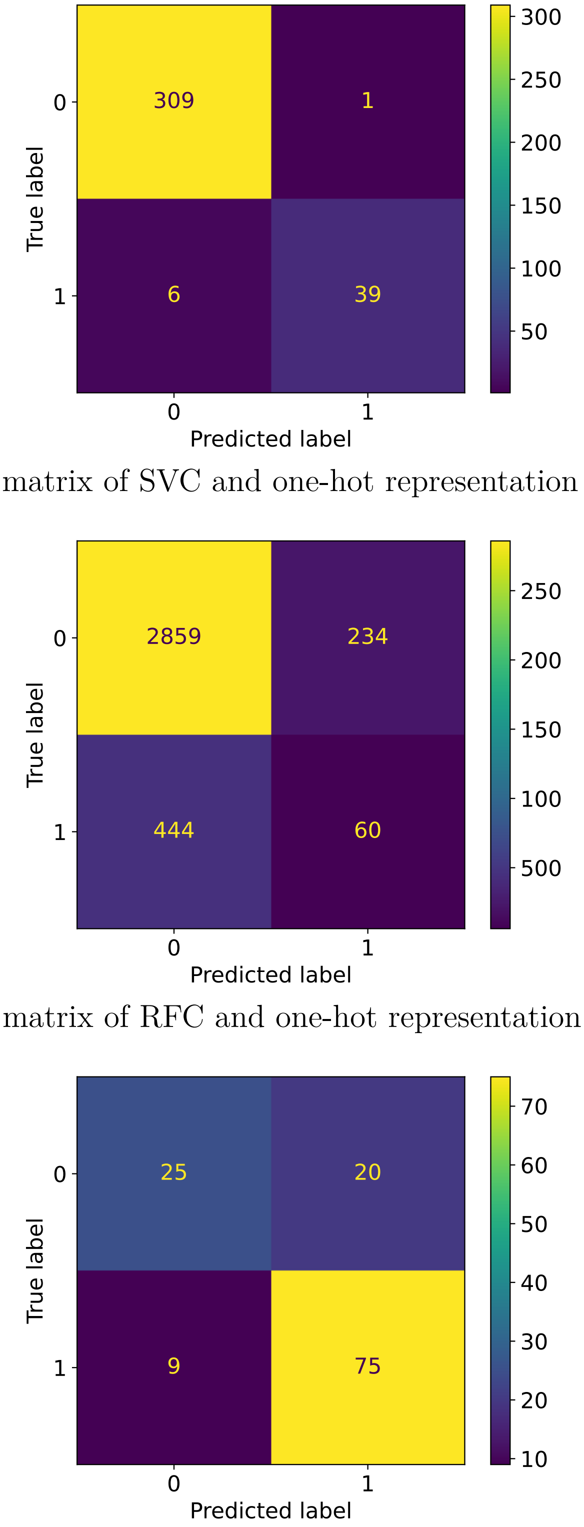
After the best combination had been chosen for each dataset, the combinations were used on the respective test set to evaluate the final models on unseen data. In Table 4.1, the final results are presented. For dataset A, $F_1$ and MCC scores of SVC and one-hot combination are better on the test set compared to the validation set. A reason behind this could be that the data in the test set is very similar to the one in the training set. The data in the test set is more similar to the training data, compared to the validation data. As seen in the confusion matrix in Figure 4.14a, this combination only predicted one FP, corresponding to 0.2% of the dataset, which implies that this combination is a good choice for classification of unseen data with similar characteristics to dataset A.

The result of the evaluation on test set B showed poor performance. This was expected as the results of the validation set of dataset B showed similar results. As seen in Figure 4.14b, the confusion matrix has a high number of FPs, which is not desirable in the drug discovery procedure.

For dataset C, $F_1$ and MCC scores of random forests together with one-hot representation on the test set are not as good as on the validation set. This indicates that there is a risk of overfitting on the training dataset when building this predictive model. Moreover, dataset C is smaller than the two other datasets, with only around a third of the number of samples in dataset A. A larger dataset where the model has more samples to train on could have a positive effect on the results. However, the model still performs well, which implies that the model and descriptor combination work on unseen data with similar characteristics as dataset C.

| Dataset | Model | Descriptor | Accuracy | $F_1$ | MCC | Recall | Precision |
|---|---|---|---|---|---|---|---|
| Dataset A | SVC | One-hot | 0.98 | 0.92 | 0.91 | 0.86 | 0.97 |
| Dataset B | RFC | One-hot | 0.81 | 0.15 | 0.05 | 0.12 | 0.20 |
| Dataset C | RFC | One-hot | 0.77 | 0.84 | 0.48 | 0.89 | 0.79 |

**Table 4.1:** Evaluation of best combination on test datasets



**(a)** Confusion matrix of SVC and one-hot representation on test set A

**(b)** Confusion matrix of RFC and one-hot representation on test set B

**(c)** Confusion matrix of random forests and one-hot representation on test set C

**Figure 4.14:** Confusion matrix for best descriptor-model combination for each test dataset.

# 5

# Discussion

In this chapter, the results of this thesis are discussed. First, the possible impact which the datasets, descriptors, and machine learning models have on the results are analyzed. Thereafter, the choice of research methodology is discussed and how these choices might have affected the results. Lastly, some suggestions for future research are pointed out, which could improve the performance and therefore also move a step closer to predicting the potency of peptides.

## 5.1 Discussion of results

Here the previously presented results will be discussed. This is done by first focusing on which aspects of the datasets could affect the results, followed by a discussion of the results on descriptor level, and lastly analyzing the performance of the machine learning models.

### 5.1.1 Datasets

Looking at the MCC for the best combinations for all datasets, the selected models for datasets A and C outperform the selected model on dataset B with scores of 0.77 and 0.71 respectively, compared to 0.19 for dataset B. Overall, the scores of $F_1$ and MCC show a much better performance for the models on datasets A and C than for dataset B. A possible explanation for this could be that the sequences in dataset B are significantly longer with a length of 501 amino acids, compared to datasets A and C which has a length of 12 and 9 amino acids, respectively. Longer sequences increase the complexity of modeling by enlarging the feature space for the descriptors, except for PseAAC. Representing longer sequences could be more complicated than representing shorter sequences. The representation for longer sequences should provide a suitable dimension of the feature space which also is computationally efficient and contains the most relevant information. Running the models and tuning hyperparameters of a large dataset containing long sequences like a dataset B is not time efficient and makes this process more challenging.

Moreover, dataset B is much larger than the other two datasets, with 14 384 samples in the train and validation sets, compared to 1 420 samples in dataset A and 512 sequences in dataset C. With larger datasets, the machine learning models have more samples to train on which should be an advantage, compared to smaller datasets.

However, dataset B probably doesn't have enough variation in the data for the size to have an impact. As only one up to five positions in the sequences have variations, which corresponds to a maximum of 1% of the sequence, the difference between sequences is probably not large enough for the model to learn anything tangible. The sequences in dataset A also originate from a wild-type sequence, however, the variety in the sequences is sufficient for the model to reach a better performance. The reason behind this is that the sequences are much shorter with a length of 12 amino acids, and therefore when one or two positions mutate, that corresponds to 8.3% or 16.7% of the sequence which leads to having a greater impact on the measured property. Furthermore, as mentioned in section 2.2.1 and 3.3.1, SVC together with PseAAC have shown good performance for long protein sequences similar to dataset B in the study made by Mohabatkar et al. [28]. In this study, the same parameters were used, i.e., weight factor = 0.05 and $\lambda = 1$. Therefore, a probable reason behind the poor results of dataset B is the type of sequences.

In dataset C the majority class is the positive class, compared to datasets A and B, which have the negative class as the majority class. The positive class is the more important class to predict correctly in this work, as those represent the peptides with high potency. However as dataset C has this as the majority class, it is easy for machine learning models to predict these correctly, and detecting the negative class will be harder. Therefore the recall and precision will automatically be higher for this dataset, compared to datasets A and B. However, this difference was handled by analyzing the performance of multiple metrics.

### 5.1.2 Descriptors

As seen in the results, one-hot representation worked best for all three datasets. One reason for this could be that, even if all descriptors in this study are sequence-based, one-hot representation is the one providing complete information about the order of amino acids in the sequences. Z-scales don't have this incorporated, and in PseAAC $\lambda$ defines the sequence-order effect, therefore it is not as explicit as one-hot representation. For example, for dataset B the optimal $\lambda$ was found to be one for both SVC and random forests, which means that only the pairwise relation with the immediate neighboring amino acid is considered, and limited information about the sequence order is incorporated. However, different $\lambda$s were investigated, and $\lambda = 1$ had the best result. For dataset A, the optimal $\lambda$ was found to be nine and five for SVC and random forests, respectively. Compared to dataset B, this means that PseAAC considers more correlation factors and reflects more information about the sequence order between the amino acids. However, when finding the optimal $\lambda$, the models were not tuned which could have affected the results. With optimized models, the optimal $\lambda$s could have been different which would affect the results.

Another advantage of one-hot representation is the size of the feature space. One-hot representation provides the most features, with a feature space of size $L \times 20$, where L is the length of sequences. Comparing this to Z-scales' $L \times 5$ and PseAAC's $20 + \lambda$, the feature space of one-hot representation is significantly greater. As the machine learning models are provided with more information, it is possible that the model can learn more complex patterns.

However, one-hot representation doesn't capture any prior domain knowledge about amino acids, therefore it was not expected that this descriptor would the best descriptor for all three datasets. One-hot representation only contains information about the number and the positions of the 20 natural amino acids in the peptide sequences. Therefore, the descriptor is a very simple representation and was not expected to get the best result. However, as seen in the feature exploration of Z-scales and PseAAC, the majority of features followed the same distribution for both classes and the SHAP values showed that most features had a relatively low impact on the model's predictions. This is a possible reason for their slightly worse performance since only a few features give an indication of which class a sample belongs to. Another reason behind PseAAC and Z-scales not having the best performance could be that the incorporated domain knowledge in these two descriptors is too general for specific tasks. Since three datasets with significantly different peptides were used in this work, with three different assays, the descriptors might be too general for these datasets. However, as the descriptors have shown promising results in for example the study by van Westen et al. [16], where Z-scales performed well for various tasks, it was expected that these would perform better than a simpler, direct representation.

### 5.1.3 Machine learning models

As seen in the results, SVC was chosen as the best model for dataset A and random forests for datasets B and C. The different characteristics of the datasets might be the reason behind these different results. For dataset A, where all sequences originate from the same wild-type sequence, with a length of 12 amino acids, random forests perform overall worse than SVC for all three descriptors. Similarly, for dataset B where all peptide sequences also originate from a wild-type sequence but are 501 amino acids long, random forests have the lowest MCC and $F_1$ score for all three descriptors, compared to SVC. However, as mentioned in the results, precision shows a better score for random forests. For dataset C, where the peptides are shorter, similar to dataset A, but much more differentiated, random forests have a similar or higher score than SVC for all descriptors and metrics.

SVC has performed well in a lot of previous research, and for datasets A and B the model shows the highest MCC for all three descriptors. However, looking at the precision, random forests perform better, which is of high importance in the drug discovery process. Establishing which model has superior performance might be a task-specific and data-dependent decision. Regardless, further research is needed.

## 5.2 Discussion of methods

The research methodology used in this thesis has had a large influence on the achieved results. Firstly, three public datasets with various formats were used, of which two were peptide datasets with shorter sequences and one protein dataset with longer sequences. The reason for using such various datasets was to see if the best combination of descriptor and machine learning model had similarities for various types of peptide sequences. In this thesis, the positive class in dataset B was chosen to be downsampled to have the same class ratios as dataset A. A more balanced dataset would probably have provided better results in this work. However, in reality, the positive class is usually the minority class with only a few samples. Therefore, downsampling the positive class better reflects the reality, and could make the results more applicable. Using public available datasets have both advantages and disadvantages. Advantages include for example that similar experiments don't have to be performed multiple times, the experiments can be conducted by experts, but the results can be used by multiple people, and in this work it was time-efficient. An example of a disadvantage is that the users can't be sure of the quality of the data, and there is no way of controlling this. Furthermore, finding a well-annotated peptide dataset large enough to train a model is difficult.

Secondly, in this thesis, three descriptors, namely Z-scales, PseAAC, and one-hot representation were implemented. These descriptors were chosen because of good performances in previous research. From these chosen descriptors, one-hot representation showed a better performance and could therefore be the focus of future research as a molecular representation when using peptides to develop new, potent drugs. The choice of descriptors certainly has an impact on the final results, and there might be other descriptors that would have performed better. However, as mentioned in the introduction, this work had to be limited to the descriptors that seemed most suitable, as the work otherwise risked becoming too large and complex.

Thirdly, a binary classification task was used in this work to predict the potency of peptides. However, the aim was first to proceed as a regression task by using the continuous values in datasets A and B as the dependent variable. However, the results of the descriptor and model combinations showed very poor performance, and it seemed as the datasets were not suitable for this kind of task. Therefore, the continuous values were converted to a positive and negative class, the models were changed into classification models, and the results improved considerably.

Lastly, the choice of methods, for example normalizing features and tuning hyperparameters have also affected the results. When tuning the hyperparameters, MCC was used as the metric when evaluating the models. This can be evaluated as a convenient metric for dataset C, as that dataset has a positive majority class, and therefore looking at precision, recall, or $F_1$ would not be beneficial. However, as the results for dataset B show, the combinations with higher MCC had a lower precision, which is disadvantageous when trying to predict new peptides. Therefore, a possible improvement could be a more thorough investigation of the choice of the metric by considering task-specific model optimization.

## 5.3 Future work

The work done in this thesis can be developed further in future research. Three different parts can be of interest, namely additional descriptors, other machine learning models, and further improvements to existing work. Since only three descriptors have been implemented in this thesis, other existing descriptors, such as structure-based descriptors, can be used as well. Another interesting proposal would be to combine the descriptors used in this thesis in different ways to see if the results would improve or get worse when using these as input to the machine learning models.

Even if SVC and random forests have shown good performance in previous research, other models can be implemented as well. Particularly neural networks are of interest as many studies have shown that this type of model works well for predicting different aspects of peptides. It would therefore be of interest to investigate how well these would perform compared to the existing results. For example, since the positional information was seen to improve the model performance, a deep learning model such as CNN can allow the model to capture the long-range patterns in the data.

Additional parts which might improve the results and can be considered for future research are for example choosing another type of method when tuning hyperparameters. In this work Gridsearch was used, however, there are other options as well, such as random search. Using another type of tuning method would use other ways of trying potential values and therefore other optimal values could be found to improve the predictive results. Moreover, when generating PseAAC in this work, only $\lambda$ is tuned, however, the second parameter, weight factor, can be optimized as well. Finding the optimal value for this parameter could also have an improvement on the results for this descriptor. Lastly, implementing feature selection to the models could improve the results. This means exploring which features are the most important and have the biggest impact on the model, and only using these features when predicting. This can improve the results of the models as unnecessary features which might have a negative impact on the predictions would be removed. By exploring the most important features, additional information which could help explain the results would also be gathered. This information can then be used in other parts of the drug discovery process and improve the exploration of future peptides.

# 6

# Conclusion

Being able to predict the potency of peptides with the help of machine learning would be a major advantage in the discovery of new peptide drugs. In this thesis, we have explored different combinations of machine learning models and descriptors on different kinds of peptides and proteins found in three public datasets, to predict the potency of peptides. In particular, we implemented three sequence-based descriptors, namely Z-scales, PseAAC, and one-hot representation. To evaluate these descriptors, they were combined with two different machine learning models, SVC and random forests. The combinations were evaluated by five metrics, accuracy, recall, precision, $F_1$, and MCC.

Of the descriptors, one-hot representation was found to perform best on all three datasets. A reason behind this could be that the position has a significant impact in the explored datasets and can be even more prominent than the amino acid information. Another reason could be that one-hot representation has the largest feature space, providing the machine learning models with the most information. From SVC and random forests, no model outperform the other since it depends on the type of data. SVC performed better for dataset A, while random forests were chosen as the best model for datasets B and C. This thesis concludes that more research has to be conducted to investigate if a general descriptor and model combination can be found to work well for various kinds of peptides.

# Bibliography

[1] K.-C. Chou and H.-B. Shen, “Recent progress in protein subcellular location prediction.” *Analytical biochemistry*, vol. 370, no. 1, pp. 1–16, 2007.

[2] J. P. Hughes, S. Rees, S. B. Kalindjian, and K. L. Philpott, “Principles of early drug discovery,” *British journal of pharmacology*, vol. 162, no. 6, pp. 1239–1249, 2011.

[3] R. C. Mohs and N. H. Greig, “Drug discovery and development: Role of basic biological research,” *Alzheimer’s & Dementia: Translational Research & Clinical Interventions*, vol. 3, no. 4, pp. 651–657, 2017.

[4] A. B. Deore, J. R. Dhumane, R. Wagh, and R. Sonawane, “The stages of drug discovery and development process,” *Asian Journal of Pharmaceutical Research and Development*, vol. 7, no. 6, pp. 62–67, 2019.

[5] M. Muttenthaler, G. F. King, D. J. Adams, and P. F. Alewood, “Trends in peptide drug discovery,” *Nature Reviews Drug Discovery*, vol. 20, no. 4, pp. 309–325, 2021.

[6] (2019) Impact story: Developing the tools to evaluate complex drug products: Peptides. https://www.fda.gov/drugs/regulatory-science-action/impact-story-developing-tools-evaluate-complex-drug-products-peptides. Accessed on: 2022-02-01.

[7] European food safety authority: Glossary. https://www.efsa.europa.eu/en/glossary-taxonomy-terms/i. Accessed on: 2022-06-16.

[8] H. Van De Waterbeemd and E. Gifford, “Admet in silico modelling: towards prediction paradise?” *Nature reviews Drug discovery*, vol. 2, no. 3, pp. 192–204, 2003.

[9] S. Spänig and D. Heider, “Encodings and models for antimicrobial peptide classification for multi-resistant pathogens,” *BioData Mining*, vol. 12, no. 1, pp. 1–29, 2019.

[10] E. C. L. de Oliveira, K. Santana, L. Josino, L. e Lima, A. Henrique, and C. de Souza de Sales Júnior, “Predicting cell-penetrating peptides using ma-

chine learning algorithms and navigating in their chemical space," *Scientific reports*, vol. 11, no. 1, pp. 1–15, 2021.

[11] K.-C. Chou, "Prediction of protein cellular attributes using pseudo-amino acid composition," *Proteins: Structure, Function, and Bioinformatics*, vol. 43, no. 3, pp. 246–255, 2001.

[12] S. Gelman, S. A. Fahlberg, P. Heinzelman, P. A. Romero, and A. Gitter, "Neural networks to learn protein sequence–function relationships from deep mutational scanning data," *Proceedings of the National Academy of Sciences*, vol. 118, no. 48, 2021.

[13] G. J. van Westen, R. F. Swier, J. K. Wegner, A. P. IJzerman, H. W. van Vlijmen, and A. Bender, "Benchmarking of protein descriptor sets in proteochemometric modeling (part 1): comparative study of 13 amino acid descriptor sets," *Journal of cheminformatics*, vol. 5, no. 1, pp. 1–11, 2013.

[14] S. Hellberg, M. Sjoestroem, B. Skagerberg, and S. Wold, "Peptide quantitative structure-activity relationships, a multivariate approach," *Journal of medicinal chemistry*, vol. 30, no. 7, pp. 1126–1135, 1987.

[15] M. Sandberg, L. Eriksson, J. Jonsson, M. Sjöström, and S. Wold, "New chemical descriptors relevant for the design of biologically active peptides. a multivariate characterization of 87 amino acids," *Journal of medicinal chemistry*, vol. 41, no. 14, pp. 2481–2491, 1998.

[16] G. J. van Westen, R. F. Swier, I. Cortes-Ciriano, J. K. Wegner, J. P. Overington, A. P. IJzerman, H. W. van Vlijmen, and A. Bender, "Benchmarking of protein descriptor sets in proteochemometric modeling (part 2): modeling performance of 13 amino acid descriptor sets," *Journal of cheminformatics*, vol. 5, no. 1, pp. 1–20, 2013.

[17] H. Mohabatkar, M. Mohammad Beigi, K. Abdolahi, and S. Mohsenzadeh, "Prediction of allergenic proteins by means of the concept of chou's pseudo amino acid composition and a machine learning approach," *Medicinal Chemistry*, vol. 9, no. 1, pp. 133–137, 2013.

[18] Y. Fang, Y. Guo, Y. Feng, and M. Li, "Predicting dna-binding proteins: approached from chou's pseudo amino acid composition and other specific sequence features," *Amino acids*, vol. 34, no. 1, pp. 103–109, 2008.

[19] X.-B. Zhou, C. Chen, Z.-C. Li, and X.-Y. Zou, "Using chou's amphiphilic pseudo-amino acid composition and support vector machine for prediction of enzyme subfamily classes," *Journal of theoretical biology*, vol. 248, no. 3, pp. 546–551, 2007.

[20] L. S. Shapley, "A value for n-person games," in *Contributions to the Theory of Games II*, H. W. Kuhn and A. W. Tucker, Eds. Princeton: Princeton University Press, 1953, pp. 307–317.

[21] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," *Advances in neural information processing systems*, vol. 30, 2017.

[22] A. B. Parsa, A. Movahedi, H. Taghipour, S. Derrible, and A. K. Mohammadian, "Toward safer highways, application of xgboost and shap for real-time accident detection and feature analysis," *Accident Analysis & Prevention*, vol. 136, p. 105405, 2020.

[23] S. M. Lundberg, G. Erion, H. Chen, A. DeGrave, J. M. Prutkin, B. Nair, R. Katz, J. Himmelfarb, N. Bansal, and S.-I. Lee, "From local explanations to global understanding with explainable ai for trees," *Nature machine intelligence*, vol. 2, no. 1, pp. 56–67, 2020.

[24] S. Lessmann, R. Stahlbock, and S. F. Crone, "Optimizing hyperparameters of support vector machines by genetic algorithms." in *IC-AI*, 2005, pp. 74–82.

[25] N. Cristianini, J. Shawe-Taylor *et al.*, *An introduction to support vector machines and other kernel-based learning methods.* Cambridge university press, 2000.

[26] J. Shen, F. Cheng, Y. Xu, W. Li, and Y. Tang, "Estimation of adme properties with substructure pattern recognition," *Journal of chemical information and modeling*, vol. 50, no. 6, pp. 1034–1041, 2010.

[27] H. Daumé, *A course in machine learning.* Hal Daumé III, 2017.

[28] H. Mohabatkar, M. M. Beigi, and A. Esmaeili, "Prediction of gabaa receptor proteins using the concept of chou's pseudo-amino acid composition and support vector machine," *Journal of Theoretical Biology*, vol. 281, no. 1, pp. 18–23, 2011.

[29] S. Lata, N. K. Mishra, and G. P. Raghava, "Antibp2: improved version of antibacterial peptide prediction," *BMC bioinformatics*, vol. 11, no. 1, pp. 1–7, 2010.

[30] A. Lindholm, N. Wahlström, F. Lindsten, and T. B. Schön, *Machine Learning - A First Course for Engineers and Scientists.* Cambridge University Press, 2022. [Online]. Available: https://smlbook.org

[31] J. N. Dybowski, M. Riemenschneider, S. Hauke, M. Pyka, J. Verheyen, D. Hoffmann, and D. Heider, "Improved bevirimat resistance prediction by combination of structural and sequence-based classifiers," *BioData mining*, vol. 4, no. 1, pp. 1–13, 2011.

[32] P. Bhadra, J. Yan, J. Li, S. Fong, and S. W. Siu, "Ampep: Sequence-based prediction of antimicrobial peptides using distribution patterns of amino acid properties and random forest," *Scientific reports*, vol. 8, no. 1, pp. 1–10, 2018.

[33] Z.-H. You, K. C. Chan, and P. Hu, "Predicting protein-protein interactions from primary protein sequences using a novel multi-scale local feature representation scheme and the random forest," *PloS one*, vol. 10, no. 5, p. e0125811, 2015.

[34] M. Feurer and F. Hutter, "Hyperparameter optimization," in *Automated machine learning.* Springer, Cham, 2019, pp. 3–33.

[35] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[36] P. Probst, M. N. Wright, and A.-L. Boulesteix, "Hyperparameters and tuning strategies for random forest," *Wiley Interdisciplinary Reviews: data mining and knowledge discovery*, vol. 9, no. 3, p. e1301, 2019.

[37] D. Chicco and G. Jurman, "The advantages of the matthews correlation coefficient (mcc) over f1 score and accuracy in binary classification evaluation," *BMC genomics*, vol. 21, no. 1, pp. 1–13, 2020.

[38] M. van Rosmalen, B. M. Janssen, N. M. Hendrikse, A. J. van der Linden, P. A. Pieters, D. Wanders, T. F. de Greef, and M. Merkx, "Affinity maturation of a cyclic peptide handle for therapeutic antibodies using deep mutational scanning," *Journal of Biological Chemistry*, vol. 292, no. 4, pp. 1477–1489, 2017.

[39] X. S. Wang, P.-H. C. Chen, J. T. Hampton, J. M. Tharp, C. A. Reed, S. K. Das, D.-S. Wang, H. S. Hayatshahi, Y. Shen, J. Liu *et al.*, "A genetically encoded, phage-displayed cyclic-peptide library," *Angewandte Chemie*, vol. 131, no. 44, pp. 16 051–16 056, 2019.

[40] P. A. Romero, T. M. Tran, and A. R. Abate, "Dissecting enzyme function with microfluidic-based deep mutational scanning," *Proceedings of the National Academy of Sciences*, vol. 112, no. 23, pp. 7159–7164, 2015.

[41] A. F. Rubin, H. Gelman, N. Lucas, S. M. Bajjalieh, A. T. Papenfuss, T. P. Speed, and D. M. Fowler, "A statistical framework for analyzing deep mutational scanning data," *Genome biology*, vol. 18, no. 1, pp. 1–15, 2017.

[42] Free epitope database and prediction resource. http://www.iedb.org/. Accessed on: 2022-03-21.

[43] S. N. H. Bukhari, A. Jain, E. Haq, M. A. Khder, R. Neware, J. Bhola, and M. Lari Najafi, "Machine learning-based ensemble model for zika virus t-cell epitope prediction," *Journal of Healthcare Engineering*, vol. 2021, 2021.

[44] B. Chen, M. S. Khodadoust, N. Olsson, L. E. Wagar, E. Fast, C. L. Liu, Y. Muftuoglu, B. J. Sworder, M. Diehn, R. Levy *et al.*, "Predicting hla class ii antigen presentation through integrated deep learning," *Nature biotechnology*, vol. 37, no. 11, pp. 1332–1343, 2019.

[45] T. Liu, K. Shi, and W. Li, "Deep learning methods improve linear b-cell epitope prediction," *BioData mining*, vol. 13, no. 1, pp. 1–13, 2020.

[46] D.-S. Cao, Q.-S. Xu, and Y.-Z. Liang, “propy: a tool to generate various modes of chou’s pseaac,” *Bioinformatics*, vol. 29, no. 7, pp. 960–962, 2013.

[47] H.-B. Shen and K.-C. Chou, “Pseaac: a flexible web server for generating various kinds of protein pseudo amino acid composition,” *Analytical biochemistry*, vol. 373, no. 2, pp. 386–388, 2008.

[48] D. Osorio, P. Rondón-Villarreal, and R. Torres, “Peptides: a package for data mining of antimicrobial peptides,” *Small*, vol. 12, pp. 44–444, 2015.

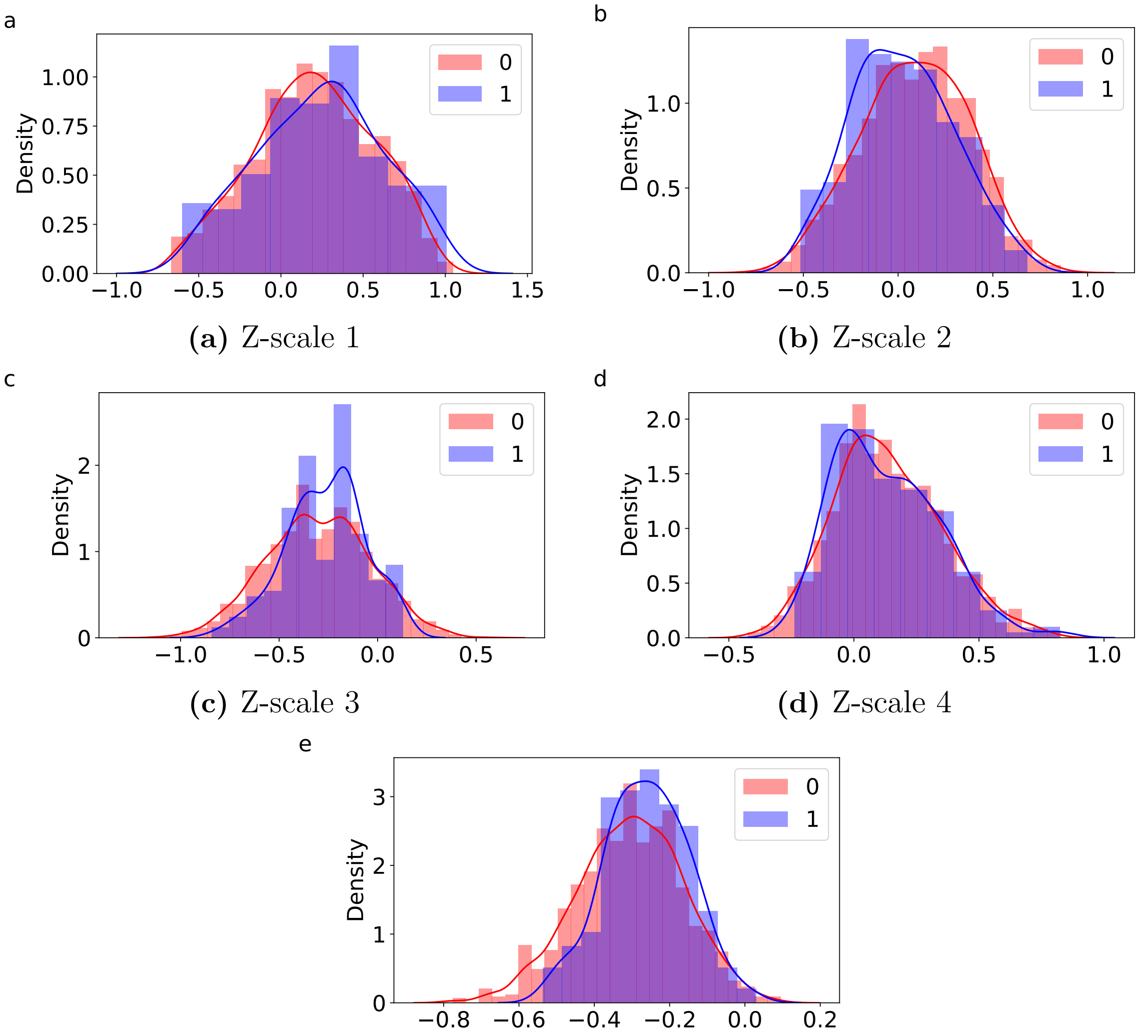
[49] H. ElAbd, Y. Bromberg, A. Hoarfrost, T. Lenz, A. Franke, and M. Wendorff, “Amino acid encoding for deep learning applications,” *BMC bioinformatics*, vol. 21, no. 1, pp. 1–14, 2020.

# A

# Feature exploration

## A.1 Z-scales

### A.1.1 Dataset A



**(a)** Z-scale 1

**(b)** Z-scale 2

**(c)** Z-scale 3

**(d)** Z-scale 4

**(e)** Z-scale 5

**Figure A.1:** Distributions of the average z-values for each class in dataset A.

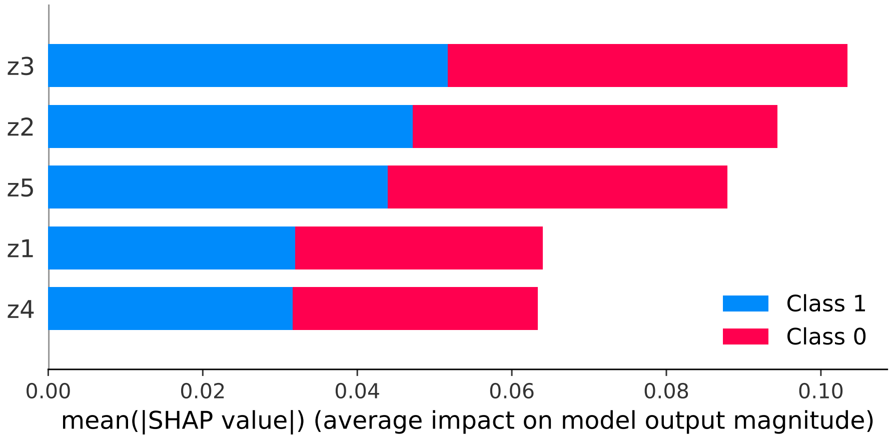
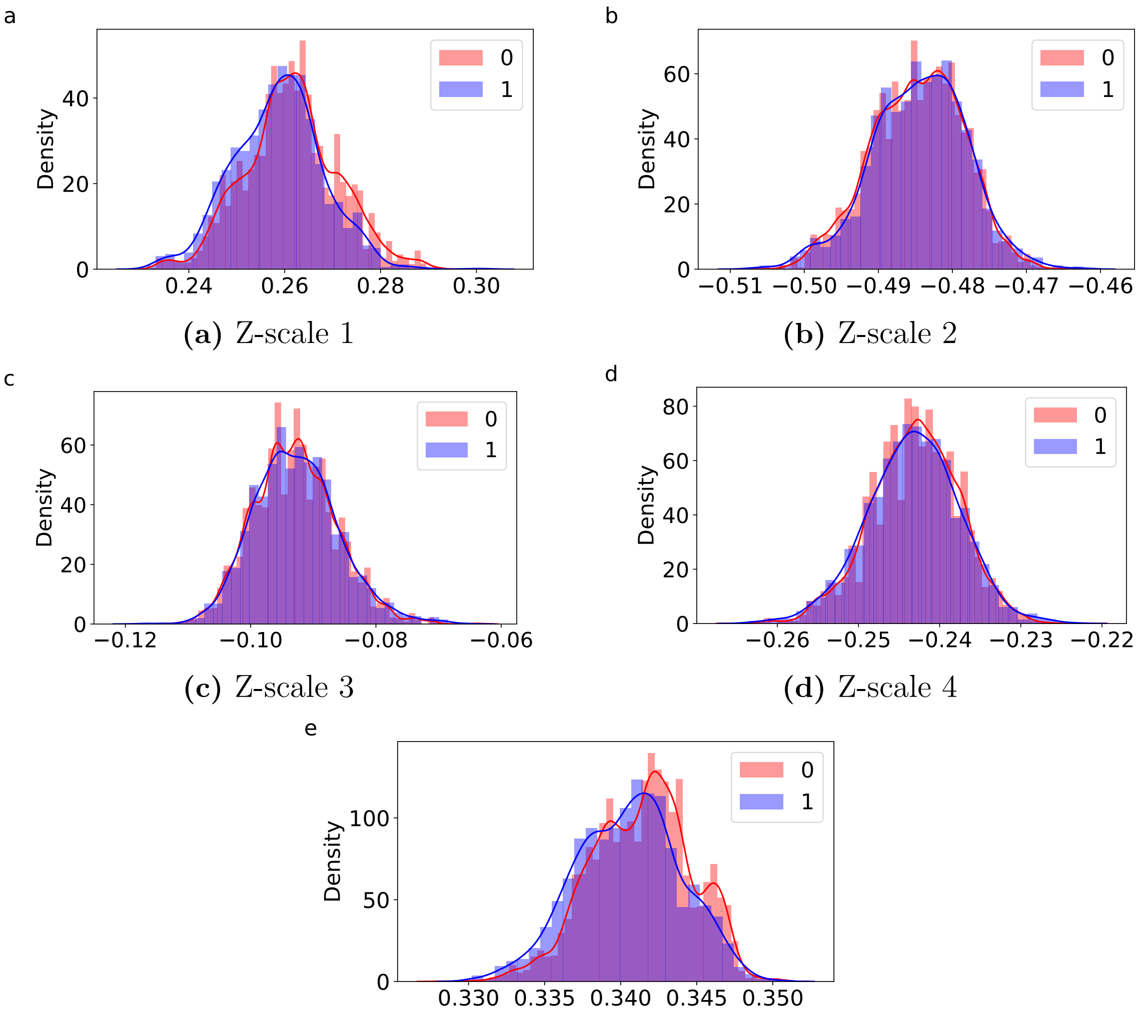


**Figure A.2:** Summary plot for the average z-values in dataset A.

### A.1.2 Dataset B



**(a)** Z-scale 1

**(b)** Z-scale 2

**(c)** Z-scale 3

**(d)** Z-scale 4

**(e)** Z-scale 5

**Figure A.3:** Distributions of the average z-values for each class in dataset B.

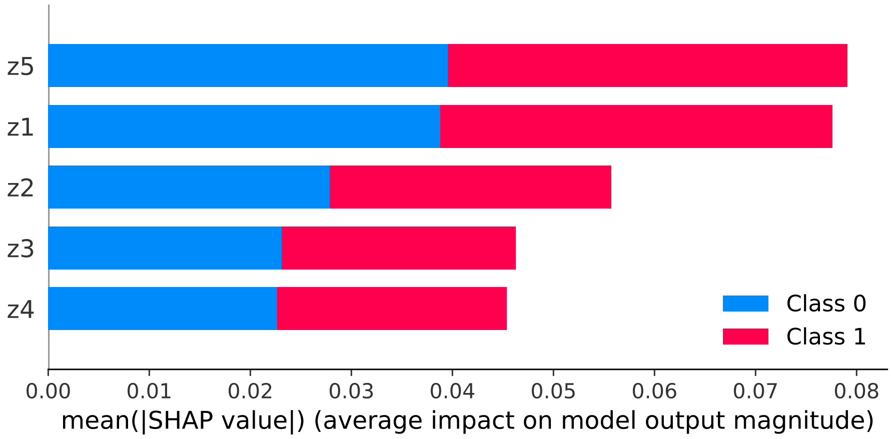
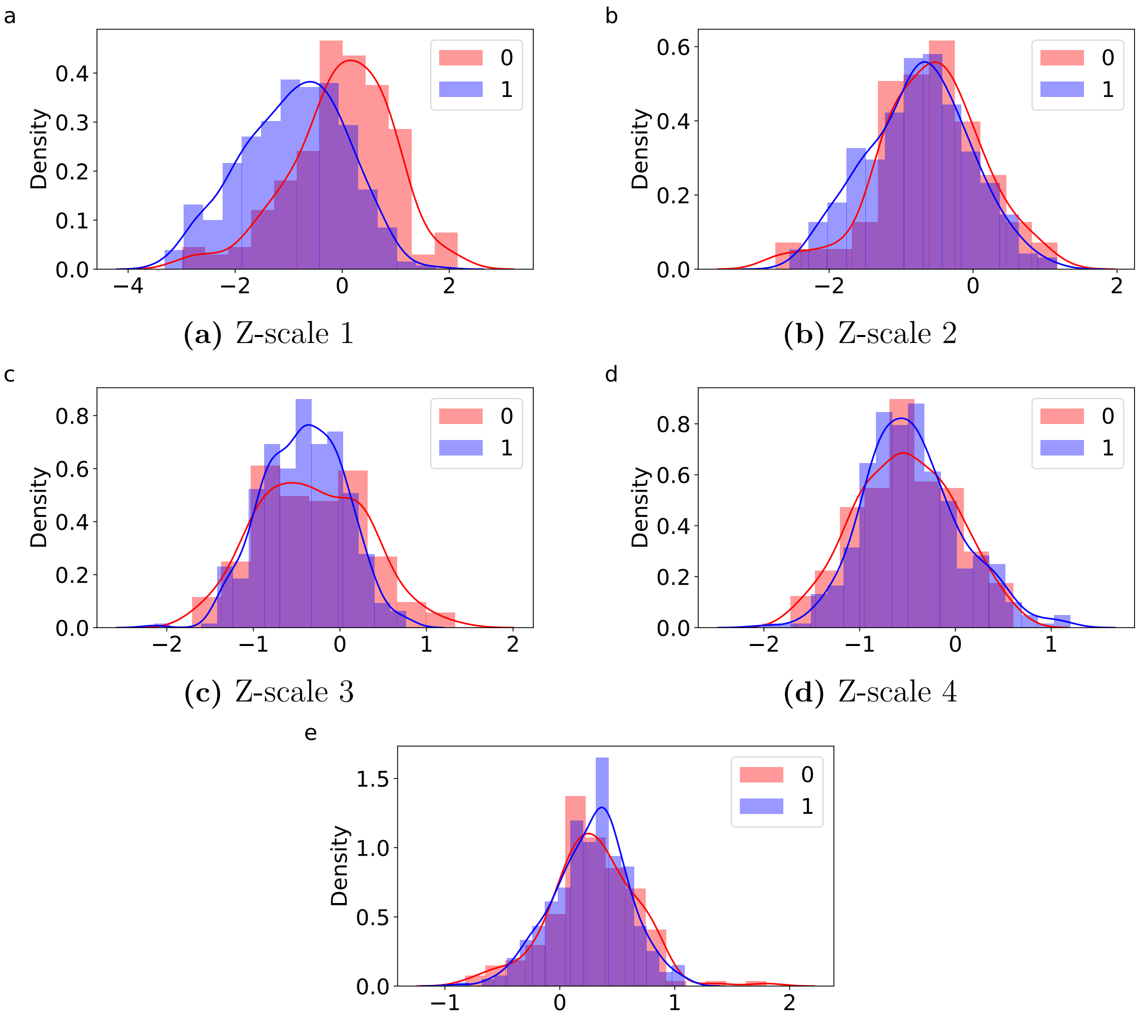


**Figure A.4:** Summary plot for the average z-values in dataset B.

### A.1.3 Dataset C



**(a)** Z-scale 1

**(b)** Z-scale 2

**(c)** Z-scale 3

**(d)** Z-scale 4

**(e)** Z-scale 5

**Figure A.5:** Distributions of the average z-values for each class in dataset C.

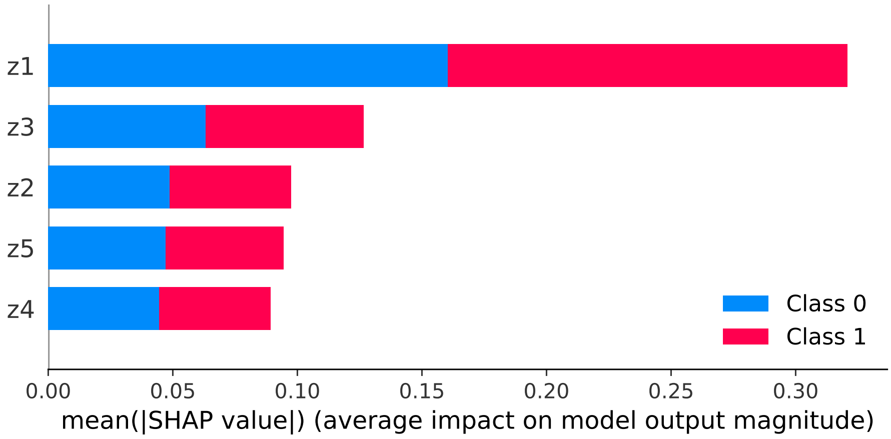
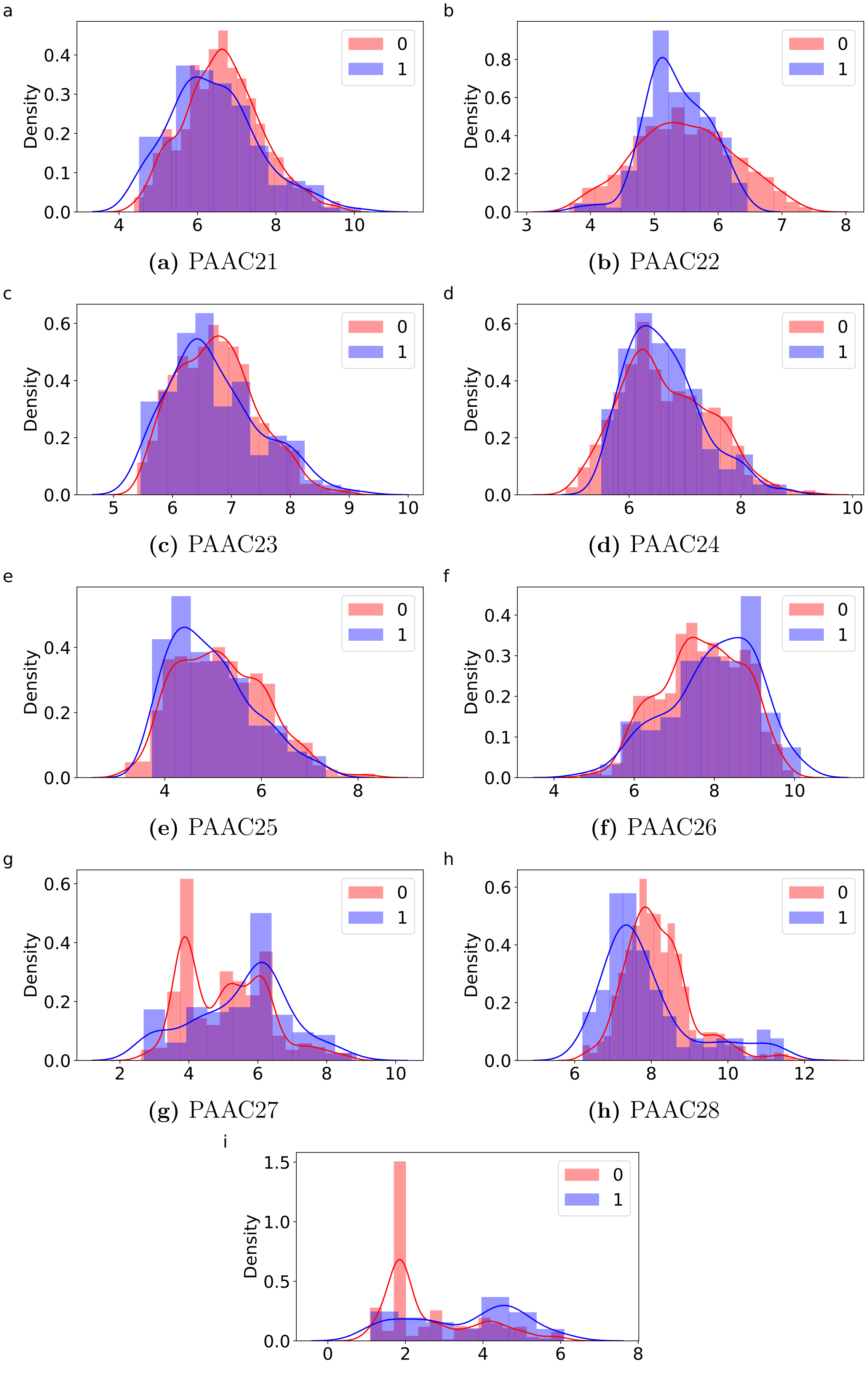


**Figure A.6:** Summary plot for the average z-values in dataset C.

## A.2 PseAAC

### A.2.1 Dataset A



**(a)** PAAC21

**(b)** PAAC22

**(c)** PAAC23

**(d)** PAAC24

**(e)** PAAC25

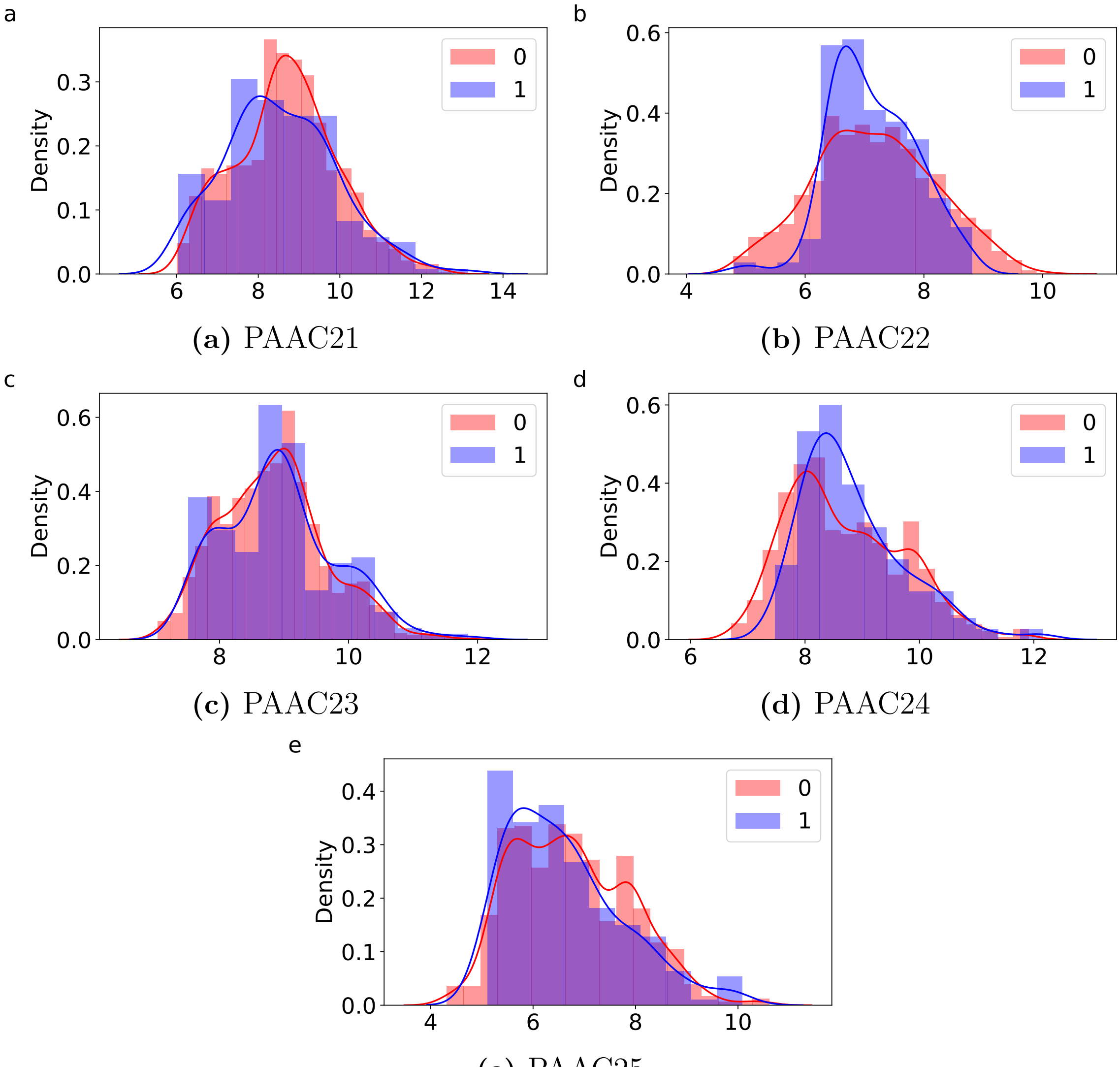
**(f)** PAAC26

**(g)** PAAC27

**(h)** PAAC28

**(i)** PAAC29

**Figure A.7:** Distributions of the pseudo components for each class in dataset A with optimum lambda for SVC.



**(a)** PAAC21

**(b)** PAAC22

**(c)** PAAC23

**(d)** PAAC24

**(e)** PAAC25

**Figure A.8:** Distributions of the pseudo components for each class in dataset A with optimum lambda for RFC.

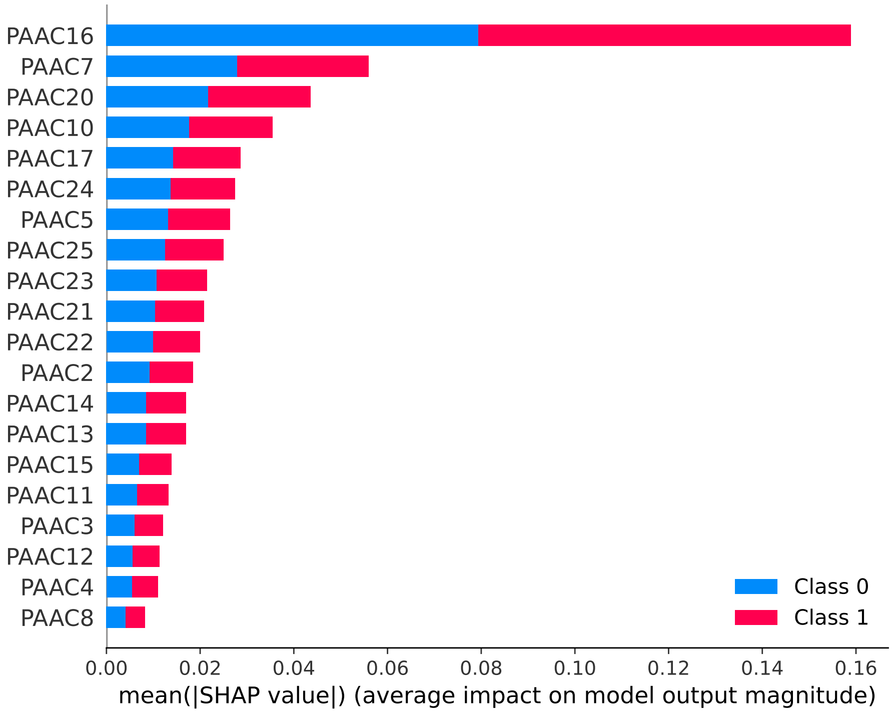
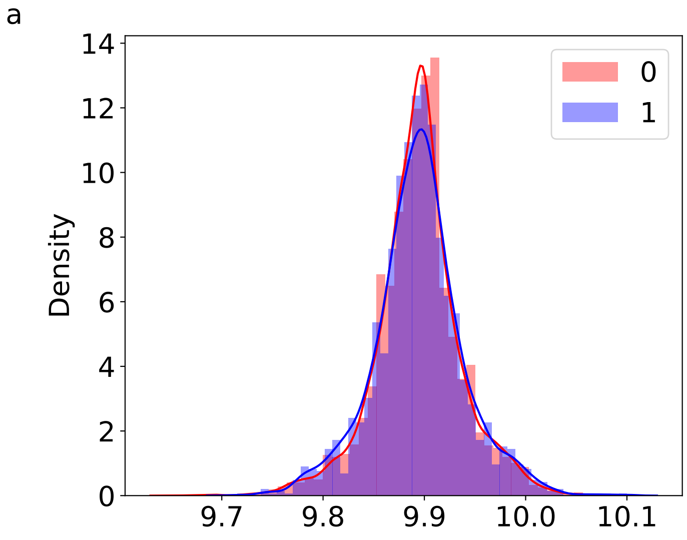


**Figure A.9:** Summary plot for the pseudo components in dataset A.

### A.2.2 Dataset B



**(a)** PAAC21

**Figure A.10:** Distributions of the pseudo components for each class in dataset B.

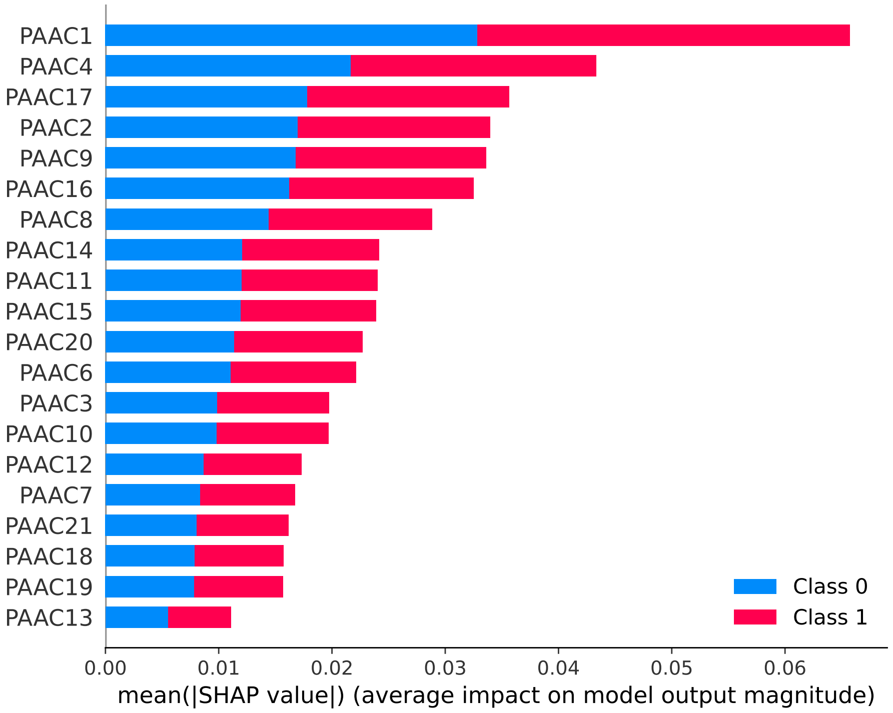
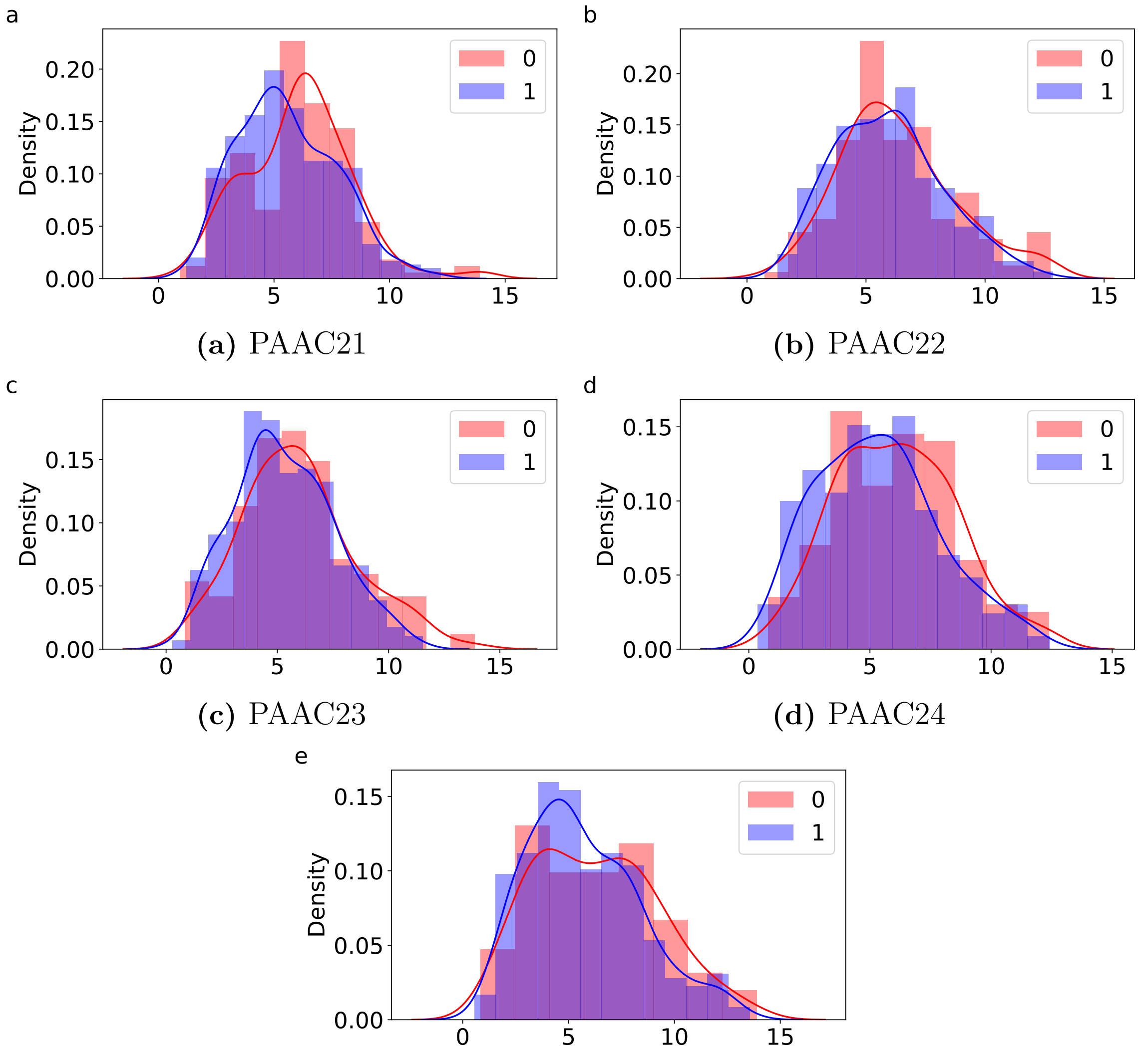


**Figure A.11:** Summary plot for the pseudo components in dataset B.

### A.2.3 Dataset C

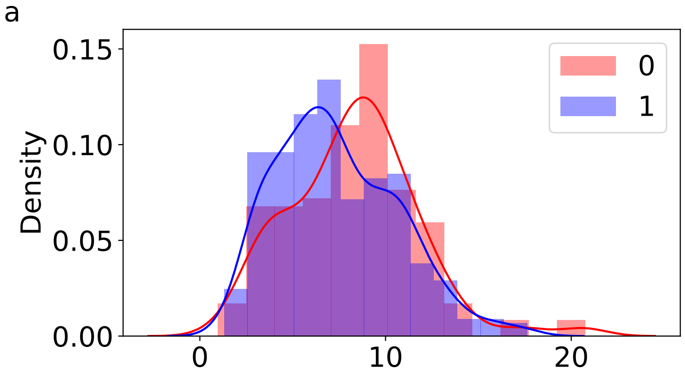


**(a)** PAAC21 **(b)** PAAC22

**(c)** PAAC23 **(d)** PAAC24

**(e)** PAAC25

**Figure A.12:** Distributions of the pseudo components for each class in dataset C with optimum lambda for SVC.



**(a)** PAAC21

**Figure A.13:** Distributions of the pseudo components for each class in dataset C with optimum lambda for RFC.

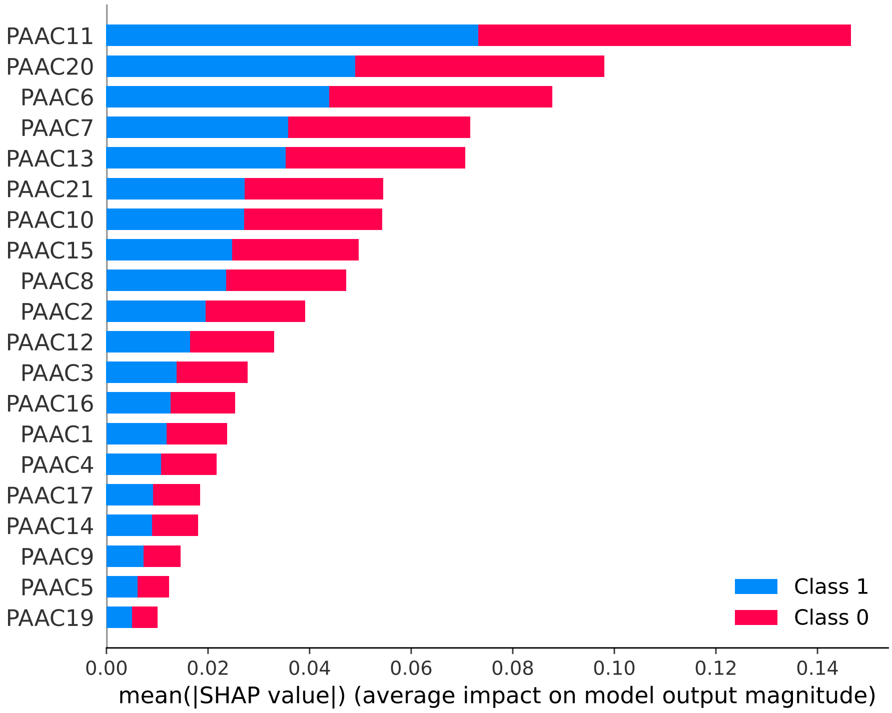
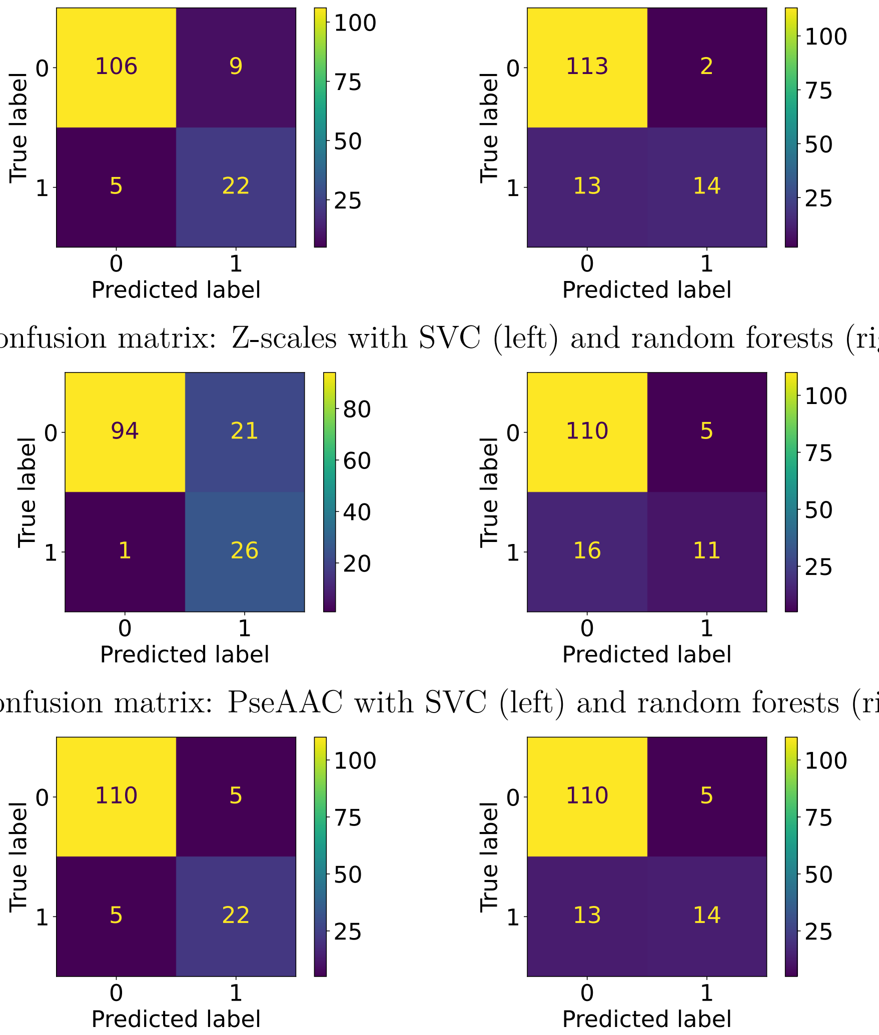


**Figure A.14:** Summary plot for the pseudo components in dataset C.

# B

# Confusion matrices

## B.1 Dataset A



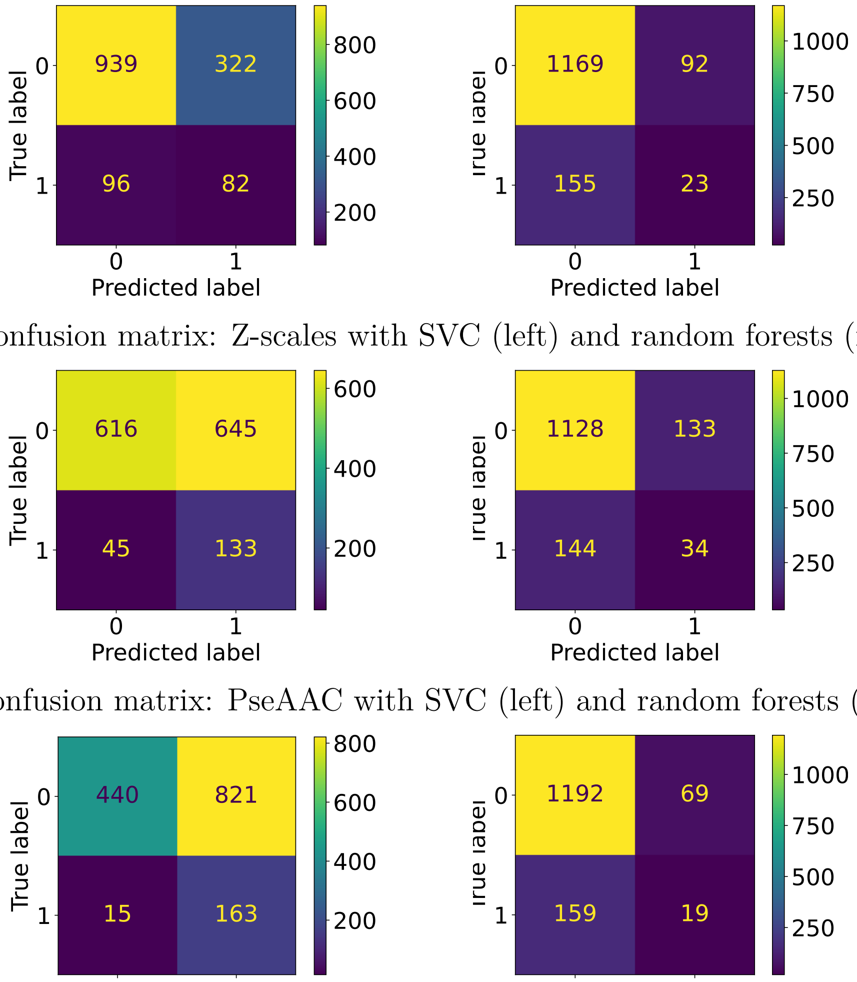
**(a)** Confusion matrix: Z-scales with SVC (left) and random forests (right)

**(b)** Confusion matrix: PseAAC with SVC (left) and random forests (right)

**(c)** Confusion matrix: one-hot representation with SVC (left) and random forests (right)

**Figure B.1:** Confusion matrix for each descriptor-model combination for dataset A.

## B.2 Dataset B



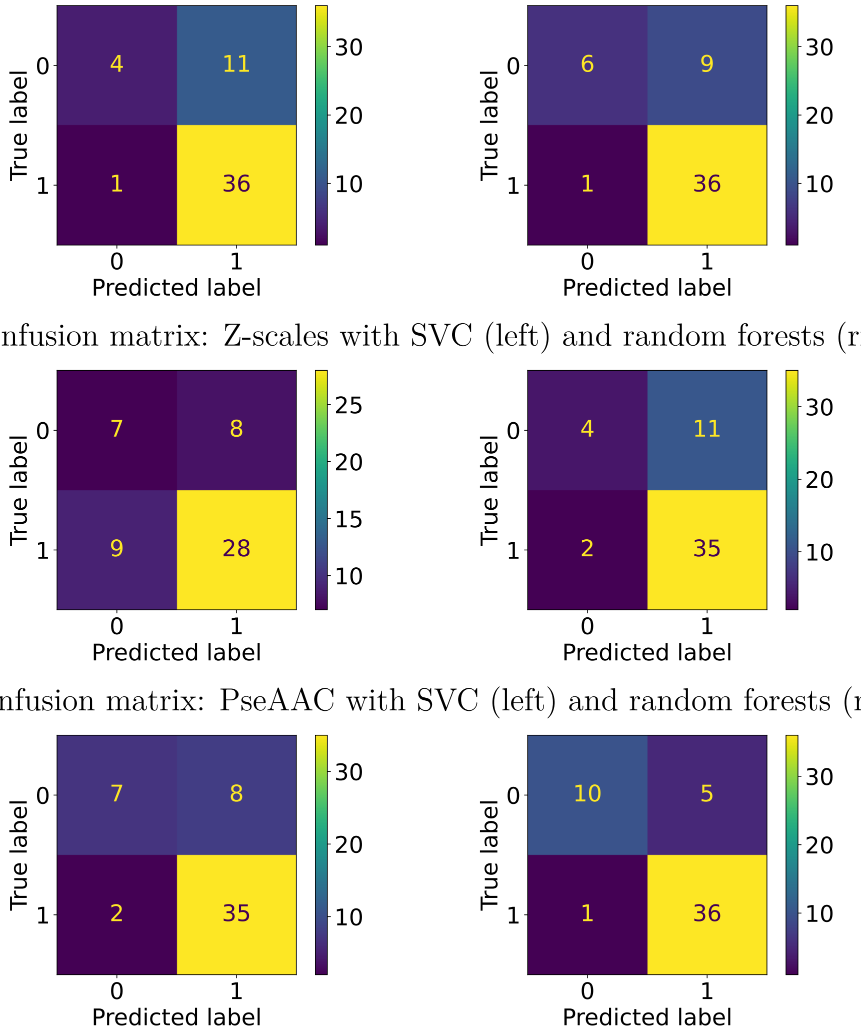
**(a)** Confusion matrix: Z-scales with SVC (left) and random forests (right)

**(b)** Confusion matrix: PseAAC with SVC (left) and random forests (right)

**(c)** Confusion matrix: one-hot representation with SVC (left) and random forests (right)

**Figure B.2:** Confusion matrix for each descriptor-model combination for dataset B.

## B.3 Dataset C



**(a)** Confusion matrix: Z-scales with SVC (left) and random forests (right)

**(b)** Confusion matrix: PseAAC with SVC (left) and random forests (right)

**(c)** Confusion matrix: one-hot representation with SVC (left) and random forests (right)

**Figure B.3:** Confusion matrix for each descriptor-model combination for dataset C.