# Specificity Prediction of Plant Acyl-ACP Thioesterases using Machine Learning

Deepro Banerjee\*, Soodabeh Ghaffari\*, Veda Sheersh Boorla\*, Michael Jindra†, Brian F. Pfleger†, Costas D. Maranas\*

\* Department of Chemical Engineering, Pennsylvania State University, University Park, Pennsylvania, USA

† Department of Chemical and Biological Engineering, University of Wisconsin-Madison, Madison, Wisconsin, USA

## Introduction

Thioesterases, which catalyze the hydrolysis of acyl-ACP (acyl carrier protein) during type II fatty acid synthesis, determine both fatty acid chain length and overall yields[1]–[3]. The goal of this project is to classify plant acyl-ACP Thioesterases (TE) based on their substrate specificity. Primary sequence information and machine learning have been widely used to classify proteins into known families/classes [4]–[7]. Here, we present a stacked ensemble framework using Support Vector Machines \cite as the base model to predict the substrate specificities of TE using their primary sequences as features. The heterogeneity among our base models in the ensemble will be governed by the different feature representation techniques used to encode the primary sequences of the enzymes. Ensemble methods are known to provide more robust and accurate predictions \cite compared to any individual model on small training samples which we will eventually illustrate through this study. Our ensemble framework achieves a mean cross validation accuracy of 0.777 across 10,000 simulations.

## Methods

### Dataset Compilation

The input dataset consists of 106 TE sequences and their corresponding substrate specificity. These characterized TE were obtained from multiple literature sources [3], [8]–[10]. These enzymes were divided into the following categories to formulate a classification problem:

1. TE with greater than 50% specificity above C12 and less than 10% specificity for C12 and below denoted by the label “1”
2. TE with greater than 50% specificity for C12 or below and less than 10% specificity above C12 denoted by the label “3”
3. TE with mixed specificity denoted by the label “2”

The length of TE sequences ranges from 300 to 412. Figure 1 shows the length of the sequences against the number of TE instances for each sequence length. The maximum count is of sequence length 399 representing the Cuphea viscosissima, henceforth referred as Cupv, class of enzymes. The dataset has a higher percentage of TE with more than 50% specificity at or below C12 and less than 10% specificity for above C12 substrates. Figure 2 highlights this trait of the dataset.Enzyme class 3 (the TE with more than 50% specificity at or below C12 and less than 10% specificity for substrates above C12) accounts for about 48% of the training data while Enzyme class 1 (TE with more than 50% specificity above C12 and less than 10% specificity for substrates below C12 accounts) accounts for about 35% of the training data. Only 17% of the training data is represented by Enzyme Class 2, (the TE of mixed specificity).

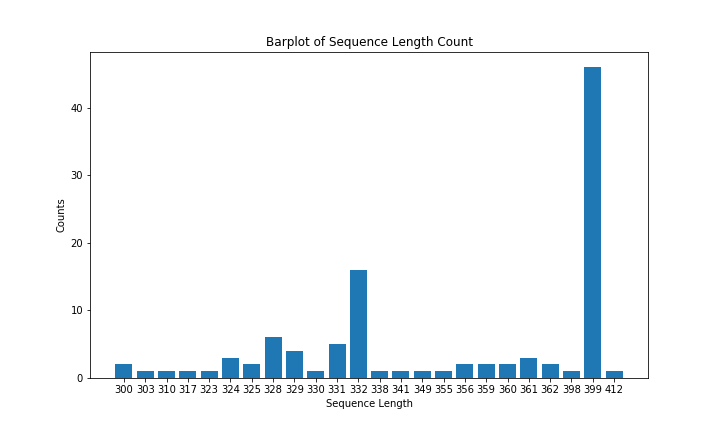


Figure : Sequence length vs count of enzymes with the same sequence length. The most frequent Thioesterase in the dataset is Cupv with 399 residues

A screenshot of a cell phone

Description automatically generated

Figure : Relative frequency distribution of TE arranged in descending order of their magnitude is shown here. Enzyme Class 3 accounts for about 48% of the training data.

### Feature Representation

We have used three different feature representation techniques to encode primary sequence information of the enzymes. The motive behind using different techniques is to extract distinct information from the sequences so that each model can produce relatively uncorrelated output predictions. The three different representations are 1) k-mer motif builder, 2) Grouped Amino Acid encoded motif builder 3) Positional feature builder. Among the three techniques, the first two can be encoded automatically using just the primary sequences of the enzymes while the last one requires expert level human intervention. Each of the representation techniques is described in the following subsections. The final step of all techniques involved creating a one hot encoded feature vector which is described in Appendix A.

#### k-mer motif builder

This representation is similar to the N-gram representation for language models [13]. Here, the entire training dataset (all enzyme sequences) is scanned to search for all possible sequence motifs of length k. A one hot encoded feature set is created with sequence motifs which are repeated in at least 2 instances of enzymes in the training data. This step is taken due to the following reasons:

* To avoid building a large set of features
* It is expected that enzymes of a specific substrate specificity will have common motifs and the main aim of the model is to learn these common motifs, not a motif specific to a particular enzyme.

Hypothetical Example:

Given 3 enzyme sequences , and where

and choosing , the set of possible 2-mers are {‘LP’, ‘LT’, ‘ML’, ‘PL’, ‘PW’, ‘ST’, ‘TM’, ‘TP’, ‘TT’, ‘WS’} among which only {‘ML’,‘TP’} are present in at least two instances. Hence the feature set will be a one-hot encoded vector of length 2 with ‘ML’ as the first index and ‘TP’ as the second index. The feature representation for the enzymes will be as follows:

#### Grouped Amino Acid encoded motif builder

This representation technique is similar to the k-mer motif builder, but it performs an additional pre-processing step on the primary sequence before searching for all possible sequence motifs of length k. In the preprocessing step, the 20 amino acid types were categorized into five classes according to their physicochemical properties, e.g. hydrophobicity, charge and molecular size. The five classes were the aliphatic group (: GAVLMI), aromatic group (: FYW), positive charge group (: KRH), negative charged group (: DE) and uncharged group (: STCPNQ). For example, a primary sequence ‘MLTPWS’ is represented as ‘’. Using this compressed representation, k-mer motif builder is used to construct a feature set as described in the above subsection.

#### Positional feature builder

For this feature representation technique, a total of 43 residue positions were selected from the aligned enzyme sequences based on their importance in prior acyl-ACP thioesterase mutagenesis studies as well as from structural analysis of the UcFatB binding pocket \cite1. Of these 43 positions, 15 were selected based on a previous study which successfully converted a long-chain acyl-ACP thioesterase from *Cuphea viscosissima* to have short-chain preference by interchanging the residues which varied in a sequence alignment cite2. An additional 3 positions were selected based on an early mutagenesis study of the acyl-ACP thioesterase from *Umbellularia californica*, which converted the predominantly C12-specific thioesterase to a predominantly C14-specific thioesterase using sequence analysis of the homolog from *Cinnamomum camphora* to guide the design cite3. Inspection of the *Umbellularia californica* structure, namely residues within the *α1* helix and the *β2*, *β4*, and *β5* helices, led to the identification of 5 additional positions cite1. The remaining positions not encompassed within the criteria above were selected based on a sequence alignment among 6 thioesterases from 3 plant species. Each of the 3 plant species had a representative of a short-chain and a long-chain acyl-ACP thioesterase. The three species included were *Umbellularia californica* (accession numbers Q41635.1 and Q41634), *Cuphea palustris* (accession numbers Q39554 and Q39555), *Cuphea hookeriana* (accession numbers AAC49269.1 and AAC48990.1), and *Cuphea viscosissima* (accession numbers AEM72522.1 and AEM72523.1). After selecting the 43 positions, amino acids in those positions were represented as a one hot encoded feature vector. Each position apart from the 20 types of amino acids may also contain a gap (result of a multiple sequence alignment). Thus a 21-dimensional one hot encoded feature vector is used to represent each position.

### Ensemble Formulation

Our ensemble framework involved three base learners which provided an output to a meta learner that predicted the enzyme class. Although all of our base learners were trained using the same principle (SVM), the heterogeneity among them was governed by the three different feature representation techniques described above. The workflow of our model is presented in Figure 3.

#### Base Model Training

The dataset was divided into a separate training and validation set based on a 75%-25% split. The dimensionality of the feature space was reduced using Principal Component Analysis because of our small sample size. The primary aim of the feature space was to reduce the number of parameters required to train the model and make our model more generalizable. A Support Vector Machine model was trained using the lower dimensional training set. 5-fold cross validation was carried out on the training set to learn the optimal set of hyperparameters. The optimal hyperparameters for each base model is given in Appendix B. The optimal set of hyperparameters were used to retrain an SVM on the entire training set. Finally, the trained SVM was used to predict the enzyme classes of the held-out validation set. This procedure was performed 10,000 times using different random seeds (which changed our training/validation set thus affecting our model) to check the dependence of our model on training data and assess it’s robustness.

A picture containing meter, clock

Description automatically generated

Figure 3: Workflow of our ensemble model. Different feature representations create separate models, and the final model output is an average of the predictions made by each individual model.

#### Principal Component Analysis (PCA) formulation

The steps involved in PCA were, 1) Standardization: The dataset was initially scaled so that each variable contributes equally.

where is the scaled value, is the original value, is the mean of the variable and is the standard deviation of the variable, 2) Covariance Matrix Computation: The covariance of each variable was calculated using the following formula:

The result was a square matrix of dimensions where is the number of variables, 3) Eigen Vector and Eigen Value Computation: The eigen vector for a matrix was calculated by solving the following formula:

4) k eigenvectors with the largest eigenvalues determination: The eigenvectors were sorted in decreasing order of their eigenvalues and the first k eigenvectors became the Principal Components. A matrix is created by combining the k eigenvectors, 5) Recasting data along the Principal Components: Finally, the original sample space was projected along this new subspace via the following equation:

where was the new sample space and was the previous sample space.

#### Support Vector Machine (SVM) Formulation

Given a training set of instance-label pairs where is a vector of features and is the label, support vector machine [16] solves the following optimization problem:

subject to:

Here the training vector is mapped to a higher dimensional space by the function . is the weight vector that represents the training parameters. is a penalty parameter for the error term . The objective of the SVM is the find a separating hyperplane with the maximal margin defined as between each class.

#### Meta Learner Formulation

Our Meta Learner is a hard-voting based majority vote classifier that compares the predictions of the base learners and outputs the class of enzymes that received the maximum votes. Thus, if the predictions of the three base learners are respectively, then the prediction of the meta learner is . In case there is a three-way tie between outputs of the base learners, the prediction of the positional feature based model is selected as the ensemble output.

## Results

### Positional feature builder-based model performed marginally better than other base models

The positional feature builder based model that requires expert intervention to preselect the positions that might be responsible for Thioesterase specificity showed slightly better performance compared to the two other automated feature encoder based models. The mean accuracy score for the positional model was 0.771 compared to 0.747 and 0.753 for the kmer based model and grouped amino acid based models respectively across 10000 simulations. In terms of worst case accuracy as well, the positional model outperformed the other two (0.44 versus 0.4 for both). The histogram of test set accuracy for the positional model over 10000 simulations is shown in Figure X while for the other two models are given in Appendix C.

### Ensemble model significantly increases worst case accuracy over base models

The primary purpose of using an ensemble framework was to decrease the amount of variance in our prediction which in our case was expected given our extremely small sample size. Our results indicate that the ensemble model significantly increases the worst case prediction accuracy over our best performing base model from 0.44 to 0.52. It also marginally decreases the standard deviation of prediction output from 0.805 for the positional model to 0.777 in case of the ensemble model. The histogram of test set accuracy for the ensemble model over 10000 simulations is shown in Fig Y.

The model was trained with the following set of hyperparameters using scikit-learn [17] :

* PCA components = 20
* k-mer length k = 3
* SVM C regularization parameter = 1
* SVM kernel = radial basis function

A random seed was specified to reproduce results. The class weight was set to “balanced” to take care of imbalance in our dataset. The “balanced” mode automatically adjusts the weights inversely proportional to the class frequencies in the input data. This step was taken to ensure that the model would penalize an incorrect prediction for Enzyme Class 2, which has the least training instances, by a greater factor than that for Class 1 or 3. The initial model gave a test set accuracy of 0.64. The hyperparameters of the PCA and SVM were optimized and the test accuracy increased to 0.68. The optimized hyperparameters were:

* PCA components = 40
* SVM C regularization parameter = 10
* SVM kernel = radial basis function

Thereafter a study of model training and test accuracy against k-mer length k was performed. The result is shown in Figure 4. It can be seen that k-mer length of 11 performed best on both training and test set. The accuracy and recall score achieved on the test set was 0.8 while precision score was 0.837. The model has failed to correctly classify 5 among the 25 enzymes of the test set which are as follows:

1. Umbellularia\_californica\_(UcFatB2)
2. Cuphea\_viscosissima\_(CvB2MT17)
3. Cuphea\_viscosissima\_(CvB2MT6)
4. Cuphea\_viscosissima\_(CvB2MT29)
5. Umbellularia\_californica\_(UcFatB1)

While it has already been reported that the Cupv class of enzymes has extremely similar features and it is hard for a machine learning model to learn useful information from such a feature set, the model may have failed to learn important features relevant to the Umbellularia\_california class of enzymes during training because both of them fell in the test set due to the random nature of our training set creation and unavailability of similar enzymes in the training set. To get a better estimate of the variance in model prediction, the model was simulated 10,000 times with the above mentioned hyperparameters and k-mer length 11 by varying the random seed. This will result in different training data and initialization parameters for SVM model both of which contribute towards the difference in model performance. The histogram of train and test accuracy of the model by varying the random seed are shown in Figure 5 and Figure 6, respectively. The mean train accuracy was 0.93 and the mean test accuracy was 0.7. The standard deviation of train and test accuracy was 0.02 and 0.09, respectively.

## Conclusion

The model variance is high, and the model overfits on the training set as evident by the difference between its test accuracy and train accuracy. The model has also failed to successfully learn the Cupv class of enzymes due to their high sequence similarity. It is important that we develop feature representations that not only capture the most important context related to substrate specificity but also have a much lower dimension. This will help us distinguish between the enzymes of the Cupv class and prevent overfitting.

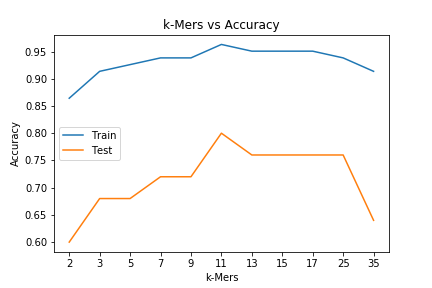


Figure : The accuracy of the model for varying k-mer length k is depicted here. It can be seen that the model accuracy was maximum for k=11 in case of both train and test dataset

A screenshot of a cell phone

Description automatically generated

Figure : Model training accuracy variance by changing the random seed

A screenshot of a cell phone

Description automatically generated

Figure : Model test accuracy variance by changing the random seed

## Future Work

The model variance can be reduced by:

* Increasing the training set.
* Adding relevant features.
* Creating an ensemble model, where an ensemble consists of multiple models and each individual model is trained separately on the entire training set. Finally, their combined prediction is taken into account.

While increasing the training set is beyond our control since we have exhaustively searched literature and databases for characterized TE, we have discussed two other ways to decrease model variance and increase model performance.

### Addition of relevant features:

We can add features to our model that are more relevant to our problem statement and simultaneously have less dimensions. This can help reduce the model variance since these features will have greater ability to extract useful information from enzyme sequences and also prevent overfitting due to lesser predictor parameters owing to its lower dimension. The following feature representations were discussed which can increase the accuracy score of our model:

**Grouped Amino Acid Composition (GAAC):** In GAAC encoding [18], the 20 amino acid types can be categorized into five classes according to their physicochemical properties, e.g. hydrophobicity, charge and molecular size. The five classes include the aliphatic group (*g1*: GAVLMI), aromatic group (*g2*: FYW), positive charge group (*g3*: KRH), negative charged group (*g4*: DE) and uncharged group (*g5*: STCPNQ). GAAC descriptor is the frequency of each amino acid group, which is defined as:

where *N(g)* is the number of amino acids in group *g*, *N(t)* is the number of amino acid type *t,* and *N* is the length of the protein/peptide sequence. This new group encoding can be used along with k-mer motif builder described above. Grouping reduces the feature space because multiple amino acids will count as one and the total number of common motifs will increase. This results in reduction of the number of predictor parameters which can increase the generalization ability of the model.

**Composition/Transition/Distribution (CTD):** The Composition, Transition and Distribution (CTD) features represent the amino acid distribution patterns of a specific structural or physicochemical property in a protein or peptide sequence [19]. Seven types of physicochemical properties have been previously used for computing these features. These include hydrophobicity, normalized Van der Waals Volume, polarity, polarizability, charge, secondary structures and solvent accessibility. These descriptors are calculated according to the following procedures: (i) Twenty amino acids are divided into three groups for each of the seven different physicochemical attributes based on the main clusters of the amino acid indices of Tomii and Kanehisa [20]. (ii) The sequence of amino acids is transformed into a sequence of certain structural or physicochemical properties of residues. The groups of amino acids are listed in Table 1. The feature creation technique is described below:

Table : Amino acid physicochemical attributes and the division of the amino acids into three groups according to each attribute.

|  |  |  |  |
| --- | --- | --- | --- |
| Attribute | Division | | |
| Hydrophobicity | Polar: RKEDQN | Neutral: GASTPHY | Hydrophobicity: CLVIMFW |
| Normalized van der Waals volume | Volume range: 0-2.78  GASTPD | Volume range: 2.95-94.0  NVEQIL | Volume range: 4.03-8.08  MHKFRYW |
| Polarity | Polarity value: 4.9-6.2  LIFWCMVY | Polarity value: 8.0-9.2  PATGS | Polarity value: 10.4-13.0  HQRKNED |
| Polarizability | Polarizability value: 0-1.08  GASDT | Polarizability value: 0.128-120.186  GPNVEQIL | Polarizability value: 0.219-0.409  KMHFRYW |
| Charge | Positive: KR | Neutral: ANCQGHILMFPSTWYV | Negative: DE |
| Secondary structure | Helix: EALMQKRH | Strand: VIYCWFT | Coil: GNPSD |
| Solvent accessibility | Buried: ALFCGIVW | Exposed: PKQEND | Intermediate: MPSTHY |

* Composition: Taking the hydrophobicity attribute as an example, all amino acids are divided into three groups: polar, neutral and hydrophobic (**Table 1**). The Composition descriptor consists of three values: the global compositions (percentage) of polar, neutral and hydrophobic residues of the protein. The Composition descriptor can be calculated as follows:

where *N(r)* is the number of amino acid type *r* in the encoded sequence and *N* is the length of the sequence.

* Transition: The Transition descriptor T also consists of three values. A transition from the polar group to the neutral group is the percentage frequency with which a polar residue is followed by a neutral residue or a neutral residue by a polar residue. Transitions between the neutral group and the hydrophobic group and those between the hydrophobic group and the polar group are defined in a similar way. The transition descriptor can then be calculated as:

where *N(r,s)* and *N(s,r)* are the numbers of dipeptides encoded as “*rs*” and “*sr*” respectively in the sequence, while *N* is the length of the sequence.

* Distribution: The Distribution descriptor consists of five values for each of the three groups (polar, neutral and hydrophobic) namely the corresponding fraction of the entire sequence, where the first residue of a given group is located, and where 25, 50, 75 and 100% of occurrences are contained. For example, we start with the first residue up to and including the residue that marks 25/50/75/100% of occurrences for residues of any given group and then we simply divide the position of this residue by the length of the entire sequence.

The addition of these structural and physicochemical properties can help distinguish between enzymes which have highly similar sequences but different substrate affinity. For example according to Jing et al. [3], two groups of residues affect Cupv specificity; those that locate to a cavity and those that map near the surface of the protein. The former is the assumed binding pocket of the acyl moiety of the substrate, and its shape and depth would directly affect the acyl chain length specificity of the enzyme. The latter defines positively charged surface patches, near the catalytic residues, that binds the negatively charged ACP moiety of the substrate and indirectly affects substrate specificity by increasing the rate of catalysis. Hence, we hypothesize that size, hydrophobicity and the charge of residues can be added as features to our model to increase the accuracy of the prediction. Other physicochemical properties like normalized Van der Waals volume, Polarity, Polarizability and Solvent accessibility have also been used before to predict enzyme-substrate interaction with great accuracy [21].

### Creation of an ensemble model

Another way of decreasing the model variance is training multiple models or creating an ensemble[22], [23]. The ensemble method we suggest is called stacking which involves training heterogenous models in parallel. The heterogeneity of our models will be governed by the different feature representation techniques, discussed above, that we can use. We propose using these variable feature representations to train separate models and finally take an average of all the model predictions as our final prediction.

Figure 3: Workflow of the proposed ensemble model. Different feature representations create separate models and the final model output is an average of the predictions made by each individual model.

## References

[1] Y. Feng, Y. Wang, J. Liu, Y. Liu, X. Cao, and S. Xue, “Structural Insight into Acyl-ACP Thioesterase toward Substrate Specificity Design,” *ACS Chem. Biol.*, vol. 12, no. 11, pp. 2830–2836, 2017, doi: 10.1021/acschembio.7b00641.

[2] M. J. Grisewood *et al.*, “Computational Redesign of Acyl-ACP Thioesterase with Improved Selectivity toward Medium-Chain-Length Fatty Acids,” *ACS Catal.*, vol. 7, no. 6, pp. 3837–3849, 2017, doi: 10.1021/acscatal.7b00408.

[3] F. Jing, L. Zhao, M. D. Yandeau-Nelson, and B. J. Nikolau, “Two distinct domains contribute to the substrate acyl chain length selectivity of plant acyl-ACP thioesterase,” *Nat. Commun.*, vol. 9, no. 1, 2018, doi: 10.1038/s41467-018-03310-z.

[4] R. Saidi, M. Maddouri, and E. Mephu Nguifo, “Protein sequences classification by means of feature extraction with substitution matrices,” *BMC Bioinformatics*, 2010, doi: 10.1186/1471-2105-11-175.

[5] D. Wang and G. Bin Huang, “Protein sequence classification using extreme learning machine,” 2005, doi: 10.1109/IJCNN.2005.1556080.

[6] B. C. Smith, B. Settles, W. C. Hallows, M. W. Craven, and J. M. Denu, “SIRT3 substrate specificity determined by peptide arrays and machine learning,” *ACS Chem. Biol.*, 2011, doi: 10.1021/cb100218d.

[7] N. K. Mishra, J. Chang, and P. X. Zhao, “Prediction of membrane transport proteins and their substrate specificities using primary sequence information,” *PLoS One*, 2014, doi: 10.1371/journal.pone.0100278.

[8] T. A. Voelker and H. M. Davies, “Alteration of the specificity and regulation of fatty acid synthesis of Escherichia coli by expression of a plant medium-chain acyl-acyl carrier protein thioesterase,” *J. Bacteriol.*, 1994, doi: 10.1128/jb.176.23.7320-7327.1994.

[9] A. Jones, H. M. Davies, and T. A. Voelker, “Palmitoyl-acyl carrier protein (ACP) thioesterase and the evolutionary origin of plant acyl-ACP thioesterases,” *Plant Cell*, 1995, doi: 10.1105/tpc.7.3.359.

[10] L. Yuan, T. A. Voelker, and D. J. Hawkins, “Modification of the substrate specificity of an acyl-acyl carrier protein thioesterase by protein engineering,” *Proc. Natl. Acad. Sci. U. S. A.*, 1995, doi: 10.1073/pnas.92.23.10639.

[11] S. F. Altschul and W. M. and D. J. L. , Thomas L. Madden, Alejandro A. Schäffer1, Jinghui Zhang, Zheng Zhang2, “Gapped BLAST and PSI-BLAST: a new generation of protein database search programs Stephen,” *Nucleic Acids Res.*, vol. 25, no. 17, pp. 3389–3402, 1997.

[12] A. M. Waterhouse, J. B. Procter, D. M. A. Martin, M. Clamp, and G. J. Barton, “Jalview Version 2-A multiple sequence alignment editor and analysis workbench,” *Bioinformatics*, vol. 25, no. 9, pp. 1189–1191, 2009, doi: 10.1093/bioinformatics/btp033.

[13] D. Jurafsky and J. H. Martin, “Language Modeling with N- grams,” *Speech Lang. Process.*, 2016.

[14] A. Krizhevsky, I. Sutskever, and H. Geoffrey E., “Imagenet,” *Adv. Neural Inf. Process. Syst. 25*, 2012, doi: 10.1109/5.726791.

[15] M. van Smeden *et al.*, “Sample size for binary logistic prediction models: Beyond events per variable criteria,” *Stat. Methods Med. Res.*, 2019, doi: 10.1177/0962280218784726.

[16] C. Cortes and V. Vapnik, “Support-Vector Networks,” *Mach. Learn.*, 1995, doi: 10.1023/A:1022627411411.

[17] S. Developers, “Support Vector Machines — scikit-learn 0.17.1 documentation,” *http://scikit-learn.org*, 2014. .

[18] Z. Chen *et al.*, “IFeature: A Python package and web server for features extraction and selection from protein and peptide sequences,” *Bioinformatics*, 2018, doi: 10.1093/bioinformatics/bty140.

[19] I. Dubchak, I. Muchnik, C. Mayor, I. Dralyuk, and S. H. Kim, “Recognition of a protein fold in the context of the SCOP classification,” *Proteins Struct. Funct. Genet.*, 1999, doi: 10.1002/(SICI)1097-0134(19990601)35:4<401::AID-PROT3>3.0.CO;2-K.

[20] K. Tomii and M. Kanehisa, “Analysis of amino acid indices and mutation matrices for sequence comparison and structure prediction of proteins,” *Protein Eng.*, 1996, doi: 10.1093/protein/9.1.27.

[21] B. Niu *et al.*, “Prediction of substrate-enzyme-product interaction based on molecular descriptors and physicochemical properties,” *Biomed Res. Int.*, 2013, doi: 10.1155/2013/674215.

[22] T. G. Dietterich, “Ensemble methods in machine learning,” 2000, doi: 10.1007/3-540-45014-9\_1.

[23] D. Marbach *et al.*, “Wisdom of crowds for robust gene network inference,” *Nat. Methods*, 2012, doi: 10.1038/nmeth.2016.

## Appendix A

### One-Hot Encoded

Each amino acid can be represented as a one-hot encoded vector. Since there are 20 amino acids, a vector of length 20 is defined with each index represented by an amino acid. For a specific amino acid, the index corresponding to that amino acid will be 1 and every other index will be zero. If A is a vector whose index is represented by a specific amino acid , then to represent the amino acid a,

Thus, a sequence of length N can be represented as a N\*20 matrix. This representation of a sequence is not only highly inefficient, but it also fails to capture the contextual information in the data.

## Appendix B

### Linear Regression Assumptions

Linear Regression assumes that:

* There exists a linear relationship between the independent variable, x, and the dependent variable, y.
* The residuals are independent.
* The residuals have constant variance at every level of x.
* The residuals of the model are normally distributed.

### Naive-Bayes Assumptions

Naive Bayes classifier assumes that:

* The effect of the value of a predictor (x) on a given class (c) is independent of the values of other predictors.
* The data is independent and identically distributed.

### Support Vector Machine Assumptions

Support Vector Machine classification problem assumes that:

* The data is independent and identically distributed.

### Logistic Regression Assumptions

Logistic Regression requires that:

* The data is independent of each other.
* There is little or no multicollinearity among the independent variables.
* Assumes linearity of independent variables and log odds.