Applications of Machine Learning for Predicting Selection Outcomes in Antibody Phage Display

A Thesis Submitted to the
College of Graduate Studies and Research
in Partial Fulfillment of the Requirements
for the degree of Master of Science
in the Department of Computer Science
University of Saskatchewan
Saskatoon

By
Daniel Hogan

Permission to Use

In presenting this thesis in partial fulfilment of the requirements for a Postgraduate degree from the University of Saskatchewan, I agree that the Libraries of this University may make it freely available for inspection. I further agree that permission for copying of this thesis in any manner, in whole or in part, for scholarly purposes may be granted by the professor or professors who supervised my thesis work or, in their absence, by the Head of the Department or the Dean of the College in which my thesis work was done. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to the University of Saskatchewan in any scholarly use which may be made of any material in my thesis.

Requests for permission to copy or to make other use of material in this thesis in whole or part should be addressed to:

Head of the Department of Computer Science
176 Thorvaldson Building
110 Science Place
University of Saskatchewan
Saskatoon, Saskatchewan
Canada
S7N 5C9

Abstract

Antibodies form an essential component of the adaptive immune system, but they also have important scientific and clinical applications. These applications exploit the proven ability of antibodies to bind strongly and specifically to nearly any biomolecular target (e.g. protein) of interest. To produce antibodies for scientific and clinical applications, researchers can use a wet-lab technique called antibody phage display. Antibody phage display starts with a library of diverse antibody fragments and selects and amplifies those fragments that bind to the target. Antibody phage display combined with next-generation sequencing (NGS) technology has the potential to yield greater insight into the selection process.

Machine learning is an area of artificial intelligence uniquely suited to recognizing patterns in large datasets, like those produced by NGS.

The research goals of this thesis were to (1) train machine learning models to predict the selection of antibody fragments in antibody phage display using only the sequence of the fragment; (2) validate the ability of the trained models to generalize to different experiments; and (3) reverse engineer the trained models to gain greater insight into the learned patterns and the selection process.

Antibody phage display data produced by the Geyer lab (University of Saskatchewan, SK) using two libraries called F and S was used to train a set of machine learning models: naive Bayes network (NB), linear model (LM), artificial neural network (ANN), support vector machine (SVM) with a radial basis function kernel (RBF-SVM), a SVM with a string kernel (SSK-SVM), and a random forest (RF). In addition, key parameters of the RBF- and SSK-SVM were tuned using a gridsearch. The trained models were then used to predict which antibody-displaying phage would be observed after the 5th round of panning, and their prediction accuracy on this data was used to help select models for subsequent analyses. The models selected were the RBF- and SSK-SVM. To achieve the second research goal, data originating from library F was used to train the two SVMs while library S data was used to test them. Finally, the two SVM models trained on library F were deconstructed to understand what features of the input correspond to negative predictions, and what features correspond to positive predictions.

The ANN, SVMs, and RF models had the best average classification accuracy (81.5%), but of this group, there was not one classifier that performed significantly better than the others. These classifiers could be used to help non-experts select clones from either library F or S for further wet-lab analyses.

The SVMs trained on library F and tested on library S achieved an average classification accuracy of 66.7%, significantly better than would be achieved by relying on chance. These two SVMs could be used to help non-experts select clones for further wet-lab analyses, provided the library being used is not too different from library S.

Finally, deconstructing the SVMs trained on library F yielded insight into the basis for their predictions. The predictions of the RBF-SVM were found to be highly dependent on the molecular weight of the relevant binding region (i.e. CDRH3).

Acknowledgements

There are many people I would like to thank for helping me complete this thesis. First, I would like to thank my supervisor, Dr. Tony Kusalik, for his guidance both when I was preparing for my graduate studies and during the completion of this thesis. I would also like to thank Bharathi Vellalore for helping me to understand antibody phage display, a central topic of this thesis. I would also like to thank Rahwa Zeresenai for her unwaivering support and encouragement over the past two years. Finally, I would like to thank my parents, Michael and Janie Hogan, for encouraging me to pursue my interests, which ultimately lead to the writing of this thesis.

Contents

List of Tables

List of Figures

List of Abbreviations

ANN	artificial neural network
CDRH1,2,3	heavy-chain CDR 1, 2, or 3
CDRL1,2,3	light-chain CDR 1, 2, or 3
CDR	complementarity determining region
CFS	correlation-based feature selection
ISP	Ion sphere particle
MUSI	multiple specificity identifier
NGS	next-generation sequencing
RBF	radial basis function
SSK	string subsequence kernel
SVM	support vector machine
mAb	monoclonal antibody
pAb	polyclonal antibody

Chapter 1

Introduction

The general aim of this thesis was to apply methods from an area of artificial intelligence called machine learning to a lab technology used for antibody discovery called antibody phage display. This introduction provides (1) a brief background for understanding this aim, (2) the motivation for this aim, (3) the research goals of this thesis, (4) an outline of the methodology of this thesis, and (5) an explanation of how this document is organized.

To defend against microbes and prevent infection, the human body is equipped with multiple layers of defence including the two main branches of the immune system: the innate immune system and the adaptive immune system. The innate immune system, evolving much earlier in history, recognizes molecular hallmarks of pathogenicity and responds to these hallmarks by sending cells and proteins to eliminate the hallmark-bearing pathogen. Some pathogens, however, do not carry any hallmarks that the innate immune system can recognize. To handle threats like these, vertebrates have evolved an adaptive immune system, which consists of specialized cells and proteins that recognize molecules that are not part of the organism. Proteins called antibodies play a pivotal role in this recognition mechanism.

Antibodies help the adaptive immune system distinguish self from non-self and focus resources toward eliminating the latter. The body is capable of producing a variety of antibodies that recognize the vast majority of non-self molecules. Even though each antibody only recognizes a specific molecular signature, the variety of antibodies produced by the body is so immense (roughly 10^{12} [1]) that, for any given target, it is very likely that there exists an antibody that can bind to that target. Antibody structure and function, and clonal selection—the process the body uses to produce pathogen-fighting antibodies—is discussed in the background of this thesis (Section 2.1).

The ability of organisms to produce antibodies that bind strongly and selectively to proteins has been exploited to produce polyclonal antibodies (pAbs) for use in diagnostics (e.g. ELISA) and therapeutics. In addition to pAbs, there is great interest in developing monoclonal antibodies (mAbs). MAbs differ from pAbs in that all of the antibodies in a mAb originate from the same cell and are thus identical in sequence and structure, whereas the the antibodies in a pAb originate from different cells and are thus heterogeneous. MAbs are attractive because, among other reasons, they are easier to study and easier to reproduce [2].

Since the production of the first monoclonal antibodies in 1975 and the first FDA licence in 1986, mAbs have become an important weapon in the clinician's arsenal [3]. Today, there are approximately 30 mAbs

approved by the FDA for treating human disease and conditions like cancer, chronic inflammatory diseases, transplantation rejection, infectious disease, and cardiovascular diseases [3]. The importance of mAbs is underscored by their global market value, which stands at approximately 20 billion USD per year; and the success of mAbs like Ramicade and Rituxan, which have annual sales exceeding 1 billion USD [4].

There are a variety of methods for developing mAbs with an affinity towards a target of interest. One such method is called antibody phage display. Antibody phage display uses bacterial viruses, called bacteriophage, to achieve clonal selection of target-binding antibodies in the lab. The procedure begins with a library of bacteriophage expressing antibody fragments on their capsid and carrying the corresponding gene in their genetic payload. In a process called panning, phage displaying target-binding antibody fragments are enriched by incubating the phage in a target-coated well, rinsing the unbound phage away, and then amplifying the immobilized target-bound phage by infecting a bacterial culture. A more detailed explanation of antibody phage display is given in Section 2.2.

One of the main goals of antibody phage display is to isolate phage that bind strongly, and specifically, to the target of interest. For therapeutic applications, target affinity is critical for increasing efficacy, reducing the required dosage, and easing side effects [5]. Because antibody phage display cannot generate antibody fragments that do not already exist in the library, the diversity of the initial library is critical to the success of the technique. Even if the library is sufficiently diverse and contains target-binding phage, antibody phage display may still fail due to other phage out-competing the target-binding phage, essentially masking them from discovery.

In order to understand library diversity and the enrichment process that happens during panning, studies have incorporated next-generation sequencing (NGS) [6, 7, 8]. NGS allows researchers to identify every sequence in a phage pool and approximate its concentration; however, to conduct an in-depth analysis on this data, intelligent and efficient computational methods are needed. Making sense of large datasets is a focus of machine learning. In addition, machine learning stresses computational efficiency and makes few assumptions about the process that gave rise to the data. An overview of machine learning as well as some of the specific techniques used in this thesis is given in Section 2.4.

This thesis will explore machine learning methods of leveraging NGS outputs from antibody phage display experiments with a view toward the following goals: (1) comparing the effectiveness of various machine learning techniques to this problem domain, (2) choosing the best machine learning method and validating its performance, and (3) demonstrating how machine learning can be used to inform the design of antibody libraries with enhanced specificity. These research goals are described further in Chapter 3.

To realize the goals of this thesis, NGS sequence outputs from real antibody phage display experiments conducted by the Geyer lab were processed to make them suitable for input to machine learning methods. A software package called Weka was used to train various machine learning methods for the task of predicting whether or not a specific clone will be observed after 5 rounds of panning given the CDRH3 sequence of that clone. Cross-validation was used to compare the performance of the machine learning methods and hone in

on one of the best techniques, called Support Vector Machines, which is described in Section 2.4.5. The SVMs were trained to predict outcomes of antibody phage display and then they were dissected to understand the basis for their performance and to suggest ways of modifying the antibody phage display library to improve specificity toward the targets used in the experiments. A complete description of the methodology is given in Chapter 4.

Listed in order, this thesis includes the following chapters: Background, Research Goals, Methodology, Results & Conclusions, and Discussion & Future Work. Background will overview theory, techniques, and literature that are necessary to understand the rest of the text. Research Goals states the specific objectives of this thesis. Methodology lays out the work that was done to complete the research goals, and provides the necessary detail for reproducing the work. Results & Conclusions presents the observations and conclusions made during execution of the methodology. Discussion provides general commentary including the implications of the results, conjectures, and directions for future work.

Chapter 2

Background

2.1 Antibodies

2.1.1 Overview

To avoid infection the human body must fight off microbes like viruses, bacteria, and parasites and to mount a defence against these pathogens, the body must have the capacity to distinguish them from self-molecules. A big part of this recognition mechanism is the responsibility of proteins called antibodies.

Antibodies are symmetrical Y-shaped proteins capable of recognizing and binding specific molecular surfaces (called epitopes) with two of its three branches (called the variable regions). During the development of antibody-producing cells, called B-cells, the antibody-coding genes of these cells are systematically randomized so that each B-cell produces its own antibody variant that recognizes a unique molecular surface. Because the number of antibody variants produced by this randomization process is so immense, the body has the capacity to produce antibodies that recognize nearly any molecular surface. To eliminate antibodies that are self-reactive and leave only those that react to foreign molecules, B-cells producing antibodies that are self-reactive are culled out in a process called negative selection. The result is an antibody repertoire that reacts to almost any threat but not to the body [9].

To accommodate such a large diversity of antibodies, the concentration of each antibody in the body is minuscule. In response to an infection, the body uses a process called clonal selection to increase the concentration of antibodies that bind to the invading pathogen. The process of clonal selection depends on the cells that produce antibodies, called B-cells. In addition to producing free-floating antibodies, a B-cell is decorated with membrane-bound proteins resembling antibodies, called B-cell receptors (BCRs). When a suitable antigen binds to a BCR, the BCR sends a signal to the B-cell that causes it to proliferate. As the B-cell proliferates, it also upregulates the production of antibodies. In this way, only antibodies that can actually aid in the battle against the invading pathogen are actually produced [9].

2.1.2 Structure

A detailed view of an antibody molecule is shown in Figure 2.1. Conceptually, the structure of an antibody has the shape of the letter Y. This Y is composed of four chains: two identical heavy chains (Hc) and two

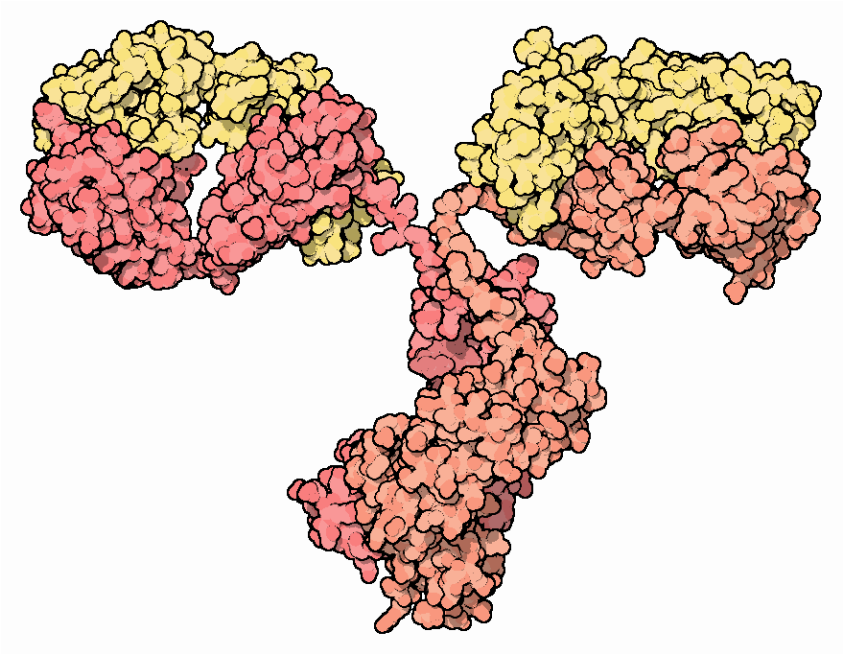
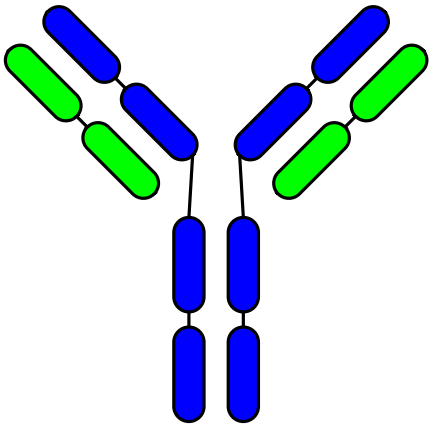


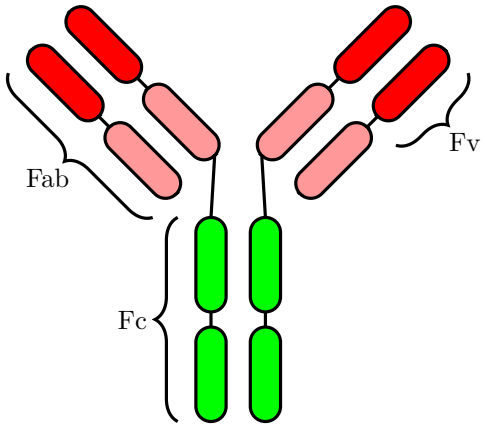
Figure 2.1: Structure of an antibody [10, 11]. Heavy chains are coloured red and light chains are coloured yellow.

identical light chains (Lc). The C-terminal halves of both heavy chains associate to form the stem of the Y while the N-terminal halves are divided between the two top branches, each associating with one of the light chains (Figure 2.2a) [9].

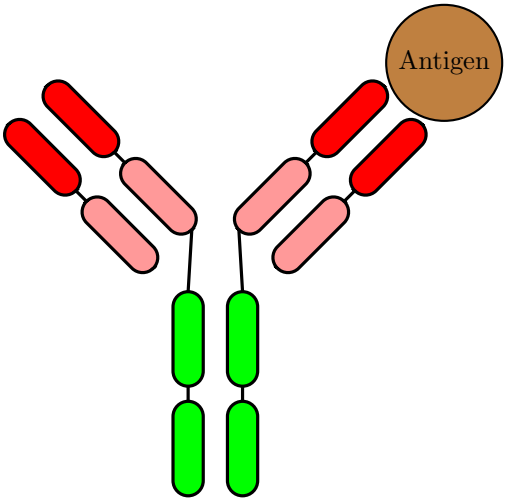
The parts of an antibody can be classified based on their function within the molecule. In order of increasing specificity, these regions are Fc, Fab, Fv (Figure 2.2b), and CDRs. The Fc region refers to the stem of the Y and is also called the constant fragment because it is identical in every antibody. The Fc region is also the part that is recognized and bound by other components of the immune system. The two remaining branches of the Y-shaped antibody are called Fabs, or antibody binding fragments. Within each Fab is the Fv region, also called the variable region. Finally, at the N-terminal tip of the Fv region resides the CDRs (complementarity determining regions). The CDRs are six loops (three from the heavy chain and three from the light chain) which are responsible for sticking to the molecular surface of the antigen (termed the epitope). Three of these CDRs (CDRH1, CDRH2, and CDRH3) come from the heavy chain. The other three (CDRL1, CDRL2, and CDRL3) come from the light chain. Antibody diversity is created by randomizing the genes encoding these loops. The reader can learn more about antibodies in the text by Sompayrac [9].



(a) Antibody molecule coloured by chain. Heavy chains are coloured blue and light chains are coloured green.



(b) Antibody molecule coloured by region. The constant region (Fc) is shown in green, the Fabs are shown in red (light and dark), and the variable regions (Fv) are shown in dark red.



(c) An antibody bound to an antigen. The surface making contact with the antigen is formed by the 6 CDRs (not shown) of the variable region. The surface of the antigen making contact with the antibody is called the epitope.

Figure 2.2: A diagram of an antibody molecule.

2.2 Antibody Phage Display

2.2.1 Overview

The observation that (1) human antibodies are well-tolerated by the body, (2) antibodies are highly specific to their target, and (3) an antibody can be made for any given target, has lead to a great investment in using antibodies for research, diagnostics, and therapeutics [3]. Much of the interest in antibodies is focused on monoclonal antibodies (mAbs). A mAb is a collection of antibodies that are identical copies, or clones, of one another. To avoid confusion, in the remainder of this document, the term clone means "a collection of identical antibodies". This definition will replace the alternate meaning of the word, which is "a single copy of an antibody". For example, a mAbs consists of a single clone, whereas polyclonal antibodies (pAbs), which are heterogeneous collections of antibodies, consist of multiple clones.

Antibody phage display is a lab technique used for developing mAbs with affinity toward targets of interest. Whereas the proliferation of antigen-binding antibodies within the human body is an example of *in vivo* clonal selection, antibody phage display is a method for *in vitro* clonal selection. Antibody phage display exploits the biology of bacteriophage (viruses that infect bacteria) to achieve this selection.

2.2.2 *In Vitro* Selection via Panning

Antibody phage display begins with a library: a collection of phage-antibody hybrids displaying antibody fragments on their surface. An antibody phage display library typically contain over 10^{10} different clones [12], comparable to the diversity of the human antibody repertoire. One might then expect that, for a given target, there exists an antibody in the library that can bind to that target. Such a hypothesis is tested using a selection process called panning. Panning has three basic steps: incubate, wash, and amplify. When conditions are ideal, these steps enrich target-binding phage. Panning can be repeated a number of times to achieve further enrichment. In the incubation step the phage are pipetted into target-coated wells. With time, phage that display antibodies with target-affinity become immobilized on the surface of the well. The next step is to rinse the well several times to wash away any unbound phage. With separation of bound and unbound phage achieved, the final step is to either recover the DNA of the phage or to amplify the bound phage remaining in the well so that further enrichment steps can be completed. Amplification is achieved by infecting a suitable bacterial culture.

Sequencing of the antibody fragments present in the phage results in thousands to millions of sequences in which a number of highly redundant sequences can be found. Barring sequencing errors, each set of redundant sequences come from a set of identical phage, called clones. The general idea is that the more abundant a sequence, the more abundant the clone, and thus the higher affinity that clone has for the target. Next-generation sequencing of antibody phage display samples is common [6, 7, 8]. The reader can find an overview of phage display in the work of Carmen & Jermutus [14].

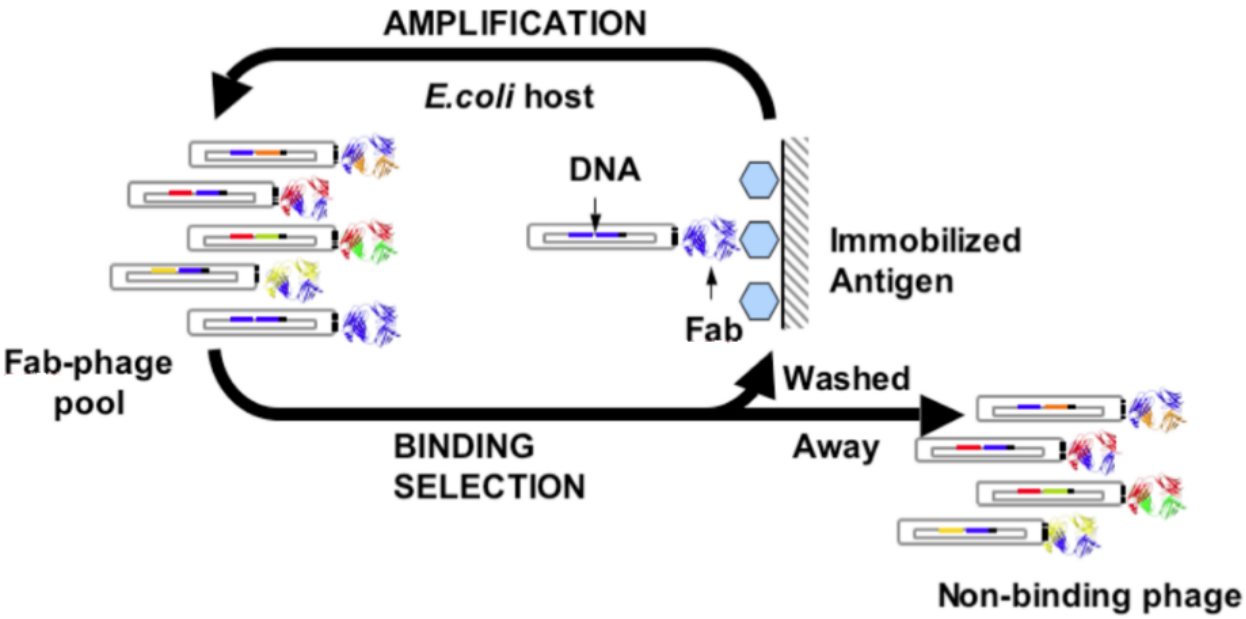


Figure 2.3: A pool of Fab-phage are selected and amplified to enrich those that bind to the target (also called the antigen). (1) A pool of Fab-phage are incubated in an antigen-coated well; (2) Unbound Fab-phage are removed by washing the well with solution, leaving Fab-phage that bound the antigen; and (3) the antigen-bound Fab-phage are eluted from the antigen and amplified in an *E. coli* host. The process can be repeated to further enrich antigen-binding Fab-phage [13]

2.2.3 Antibody Library

In antibody phage display, a population of phage is called a library. One common, albeit simplistic, metric for the quality of a library is its diversity, or number of distinct clones. Some state-of-the-art libraries have estimated diversities of over 10^{10} clones, rivalling the diversity of the human antibody repertoire [12].

Libraries are constructed with DNA coding for antibody fragments with diverse variable regions. Each DNA fragment is then inserted into a vector coding for a bacteriophage capsid protein such that the expression product of the vector is a fusion protein between the capsid and the antibody fragment. The recombined vector is then transfected into a bacterial culture infected with the same type of bacteriophage. During virion assembly, the fusion protein is incorporated into the phage capsid alongside the wild-type capsid proteins, producing bacteriophage that display the antibody fragment on their surface. Moreover, the DNA of the vector contains specific signals that allow it to be packaged within the phage progeny. The result is bacteriophage carrying the DNA of the antibody that decorates their surfaces. The linkage between antibody DNA and antibody fragment turns out to be crucial for *in vitro* selection [14].

The DNA used to construct antibody phage display libraries can be derived from the antibody repertoire of an organism, or synthesized using a template antibody and a suitable mutagenesis technique. Constructing an antibody library synthetically offers greater control over the makeup of the constructed library. The CDRs of synthetic libraries can be made to follow a specific design. For example, the libraries studied in this thesis, library F and library S, were constructed synthetically according to the specification shown in Table 2.1 [14].

Library F is a synthetic antibody phage display library that uses a constant antibody framework and variable CDR-H1, H2, H3, and L3, with most diversity focused toward CDR-H3. The length of CDR-L3 and H3 varies from 8 to 12 residues and 7 to 23 residues, respectively. Library S was designed around library F, but unlike library F, contains no variability in CDR-H1 and H2, and has a CDR-H3 that can vary in length from 7 to 25 residues. The specification for library F and library S is shown in Table 2.1 [13].

2.2.4 Panning Target

In antibody phage display, the molecule one wishes to develop an antibody for is called the target. Seven protein targets used in experiments carried out by the Geyer Lab are shown in Table 2.2.

2.3 Computational Biology

2.3.1 Sequence Alignment

Biological sequences, such as DNA and proteins, are easily represented using strings of symbols (i.e. sequences of characters). For example, the DNA sequence consisting of the bases guanine-adenine-thymine-thymine-

Table 2.1: CDR specifications for library F and library S. One-letter abbreviations are used to signify amino acids. Brackets signify that any of the contained amino acids may appear at the position in the sequence. Superscripts denote repetitions in the sequence.

CDR	Library	Specification
L1	F	RASQSVSSAVA
L1	S	RASQGISNYLA
L2	F	YSASSLYS
L2	S	YAASSLQS
L3	F	QQ[YSGAFWHPV]$^{3\text{-}7}$[PL][IF]T
L3	S	QQ[YSGTAPHREFWVL]^{4}PLT
H1	F	AASGFN[IL][YS][YS][YS][YS][IM]H
H1	S	AASGFTFSSYGMH
H2	F	[YS]I[YS][PS][YS][YS][SG][YS]T[YS]
H2	S	VISYDGSNKY
H3	F	AR[YSGAFWHPV]$^{1\text{-}17}$[AG][FLIM]DY
H3	S	AR[YSGTAPHREFWVL]$^{1\text{-}10}$[AGDY]FDY and AR[YSGAFWHPV]$^{7\text{-}15}$YYYY[GY][MF]DV

Table 2.2: Summary of protein targets.

Short Name	Full Name	Gene	Description [15]
Axl	Tyrosine-protein kinase receptor	AXL	Cell signalling receptor that helps regulate cell survival, cell proliferation, migration, and differentiation.
Jagged1	Protein jagged-1	JAG1	Ligand for Notch receptors. Believed to affect cell-fate decisions during hematopoiesis.
Jagged2	Protein jagged-2	JAG2	Ligand for Notch receptors. Affects limb, craniofacial, and thymic development.
Mer	Tyrosine-protein kinase Mer	MERTK	Cellular signal receptor that regulates cell survival, migration, differentiation, and phagocytosis.
Notch1	Neurogenic locus notch homolog protein 1	NOTCH1	Receptor for jagged-1 and jagged-2 (see JAG1/JAG2).
Notch2	Neurogenic locus notch homolog protein 2	NOTCH2	Receptor for jagged-1 and jagged-2 (see JAG1/JAG2).
Notch3	Neurogenic locus notch homolog protein 3	NOTCH3	Receptor for jagged-1 and jagged-2 (see JAG1/JAG2).

adenine-cytosine-adenine can be represented as the string "GATTACA". This unambiguous representation is easily manipulated by computers, which enables computers to carry out useful biological operations like sequence alignment.

Sequence alignment is usually carried out to determine whether two sequences are similar enough to assume they share some characteristic (e.g. evolutionary history, protein structure, function etc.). In silico, determining how similar two sequences are is carried out by finding the alignment which maximizes an objective function called the scoring function. The scoring function expresses, in formal terms, how good an alignment is. The resulting alignment can itself be represented as two strings, one shown above the other as in Figure 2.4a. Sequence alignment can be performed on more than two sequence. Such an alignment is called a multiple sequence alignment (MSA). An example of an MSA is shown in Figure 2.4b.

2.3.2 Position-weight matrix

A position-weight matrix (PWM) is one way to represent a MSA. In a position-weight matrix, the rows represent each of the 20 amino acids and the columns represent each position in the MSA. The value stored in the element at row i and column j is the probability of observing amino acid i at position j of the MSA.

```
GATTACA
 ||||
-ATTA--

GATTACA
 |||  |
-ATT--A
```

(a)

```
GATTACA
-ATTA--
GA-TACA
```

(b)

Figure 2.4: (a) Two possible alignments of the DNA sequence ATTA with the DNA sequence GATTACA. A priori, the first alignment is better because it does not contain internal gaps. (b) A multiple sequence alignment of the DNA sequences GATTACA, ATTA, and GATACA.

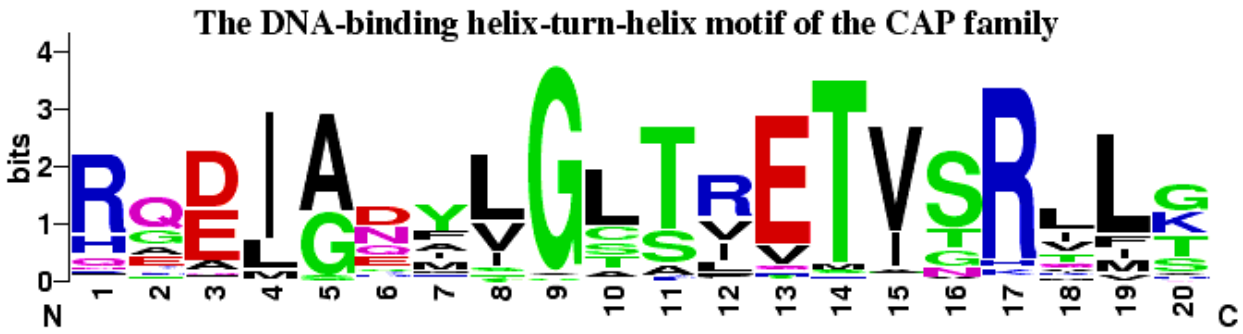


Figure 2.5: An example of a sequence logo [16].

2.3.3 Sequence Logo

A sequence logo is a visualization of an MSA. An example is shown in Figure 2.5. A sequence logo shows the positions of the MSA along a horizontal axis. Above each position, a number of pictures are stacked vertically. Each picture in the stack depicts a single symbol, but the symbol is stretched or squashed to occupy a specific amount of vertical space. The vertical space occupied by the picture signifies the probability in bits ($-\log_2 P$) of observing the depicted symbol at the corresponding position in the MSA.

2.4 Machine learning

2.4.1 Overview

In data analysis, one deals with sets of repeated measurements, e.g. species, petal length, and petal width for each iris in a garden. In their entirety, these measurements form a dataset.

The measurements in a dataset are often dealt with mathematically as vectors, e.g. an iris of species versicolor having a petal length of $1.4''$, and a petal width of $0.2''$ can be encoded as the 3-dimensional vector $(\text{versicolor}, 1.4'', 0.2'')$. If all of the measurements are numerical, the vector can be thought of as a point in

N-dimensional space, where N is the number of measurements associated with the observation. Thought of this way, the observation is called a datapoint.

Machine learning deals with the development and application of algorithms that extract meaningful patterns from large datasets. Sometimes these patterns reveal hidden structure in the dataset. Looking for these kinds of patterns is dealt with in the area of unsupervised machine learning. An example of unsupervised machine learning is shown in Figure 2.6. Other times, these patterns are used to predict the unknown attributes from partial observations, e.g. predicting the species of an iris from its petal length and width. Looking for these kinds of patterns is dealt with in the area of supervised machine learning [17].

To predict unknown attributes, supervised machine learning techniques use a function or combination of functions that takes a datapoint as input and outputs a number or label. This number or label output by the model becomes the prediction for the unknown attribute. Before the model can make reasonable predictions, however, it must first be trained with a dataset. Training allows the model to learn apparent relationships between the known attributes, also called predictor variables, and the unknown attributes, also called response variables.

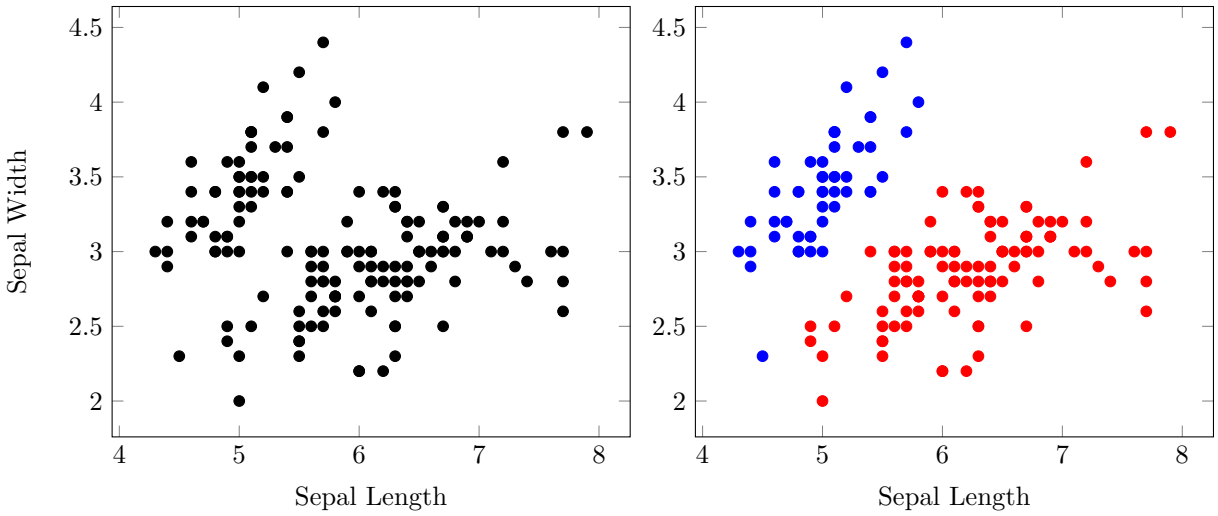
Supervised machine learning can be further separated into classification and regression problems. A regression problem arises when the response variable is continuous (in the mathematical sense). An example of a regression problem is trying to predict the height of an individual based on a set of genetic predictors. A classification problem arises when the response variable is categorical (i.e. can be enumerated). An example of a classification problem would be trying to diagnose a patient as either infected or healthy based on a set of clinical observations.

2.4.2 Training and learning

The functions that make up a machine learning model contain a number of adjustable parameters that affect the predictions the model makes. A training algorithm is an optimization procedure that adjusts these parameters in order to minimize the prediction error on the training dataset. For example, in a linear model having the form $y = mx + b$, the slope m and y-intercept b are the parameters that are optimized during linear regression, a sort of training algorithm.

2.4.3 Testing and validation

After training a machine learning model on the training dataset, it is necessary to test the model on another dataset, called the testing dataset. The purpose of testing is to show that the model has not simply learned to remember the training dataset, but has actually learned meaningful patterns that generalize to observations outside of the training dataset.



(a) Before clustering.

(b) After clustering.

Figure 2.6: A hypothetical example of unsupervised machine learning. (a) The initial data consisting of two measurements: sepal length and sepal width. The initial data was subjected to clustering, which tries to divide the data into well-defined groups. (b) The same data coloured red or blue according to the clusters identified by a clustering procedure.

2.4.4 Hyperparameters

In addition to normal parameters, which are optimized during training, machine learning models often have parameters which can be set by the user. These parameters are called hyperparameters. Hyperparameters can have a broader impact on the resulting model than normal model parameters. An important consideration in machine learning is finding the hyperparameter values that work best for a particular problem.

2.4.5 Machine Learning Techniques

Correlation-Based Feature Selection

In machine learning, raw input is often processed into a smaller set of variables, called features, which are then fed into the machine learning tool to predict the response variable. The processing of raw input into features is called feature extraction.

For a given prediction task, there may be features that do not correlate well with the response variable. These features are said to be noisy. There may also be features that correlate so well with each other that their combined predictive power is worth no more than the predictive power of each feature alone. These features are said to be redundant. Correlation-based features selection (CFS) is a procedure invented by Mark A. Hall [18] that selects noisy and redundant features to discard.

The main contribution of CFS is a method for measuring the merit of a feature set. The method uses an equation that measures the average correlation between each predictor and the response variable, but

penalized for correlation between predictors. CFS can handle not only continuous variables but ordinal, nominal, and binary variables as well.

The Weka interface to CFS provides a number of common search strategies for finding the features that maximize the CFS equation. One of these search strategies—the one used in this thesis—is called best-first search.

The best-first search implementation can start with either (1) the empty set of features or (2) the set of all features. In this thesis, the empty set of features was used, so only this method will be described but before best-first search is described, some basic terms and concepts need to be defined:

Child: A feature set A is said to be the child of another feature set B if A contains all of the features of B plus one more.

Expansion: The expansion of a feature set is the enumeration of all its children.

Best-first starts by expanding the empty set (e.g. Figure 2.7a). Expansion of the empty set results in the discovery of a number of features sets, each containing only a single feature. The search continues by choosing the best unexpanded set to expand next (e.g. Figure 2.7b). The search stops when 5 consecutive expansions do not improve upon the CFS score of the best set. When the stopping criteria is met, the algorithm returns the best set that was discovered.

Figure 2.7: (a) The empty feature set is expanded, resulting in the discovery of three new feature sets. (b) The feature set with the best CFS score ($\{b\}$) is expanded, resulting in the discovery of two more feature sets.

Decision Trees and Random Forests

A decision tree is a classifier that uses a structured set of rules for classifying new instances (Figure 2.8). To classify a datapoint using a decision tree, the rule at the root of the tree is applied first. If the rule is satisfied, the next rule applied is the left descendent of the root, otherwise it is the right descendent. This proceeds down the tree until a leaf node is reached. The label contained in the leaf node becomes the predicted class of the datapoint.

A decision tree can be built to correctly classify every datapoint in a dataset; however, the ability of such a tree to generalize to other data is typically poor. A random forest is an attempt to counteract the poor generality of a decision tree. A random forest is a collection of decision trees, each trained on a different subset of the training data [19]. After training, the random forest contains a collection of decision trees that are d ifferent, yet are trained to do the same thing. By averaging over the predictions of each decision tree, a classifier that is less prone to overfitting results (i.e. less dependent on the training data).

For random forests, bagging and random feature selection are two methods for injecting randomness into the training of each decision tree. In bagging, the training dataset is resampled before training each decision tree. In random feature selection a random subset of features is used to train each decision tree [20].

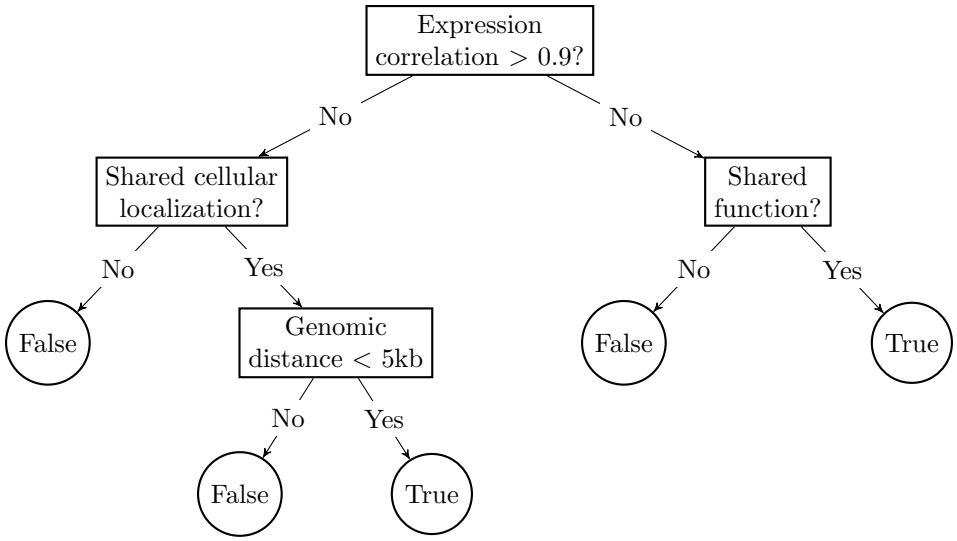


Figure 2.8: A hypothetical decision tree for deciding whether a pair of proteins interact [21].

Logistic Model

A logistic model is a prediction tool used for binary classification problems. The two classes of a binary classification problem will be called the positive and negative classes. The input to a logistic model is a set of numeric variables and the output is a number between 0 and 1 that approximates the probability that the input belongs to the positive class. For a given input, if the logistic model produces a value of 0.5 or greater, the model predicts that the input belongs to the positive class; otherwise, the model predicts that the datapoint belongs to the negative class. A logistic model is trained by maximizing the likelihood function of the model. The likelihood function expresses the probability that the model generates the observed data, and is described in the text by Witten and Frank [19].

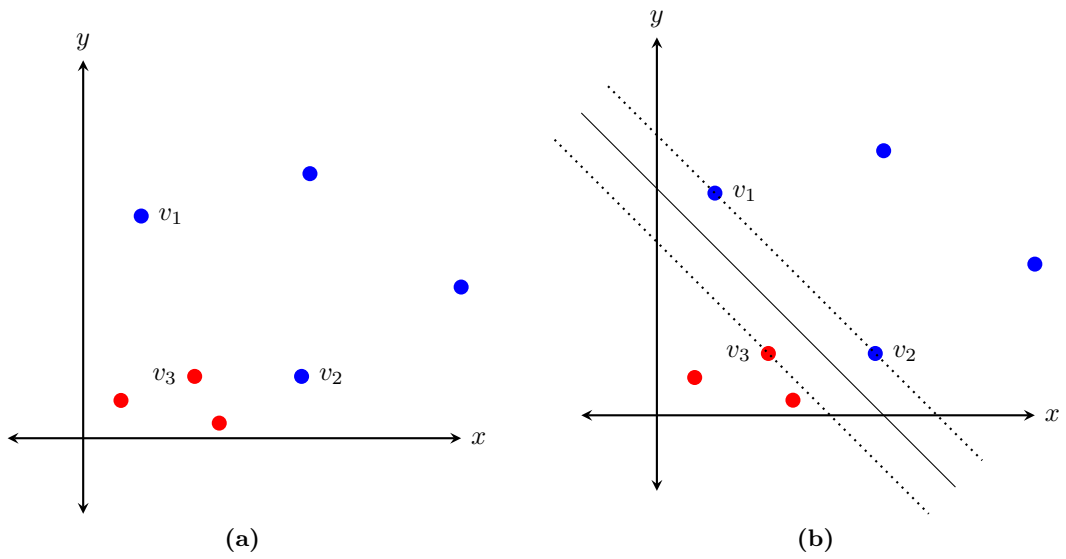


Figure 2.9: An example of a maximum-margin separating plane. (a) Datapoints with two variables represented on the x and y axes. The red dots belong to one class and the blue dots belong to another. (b) The maximum-margin separating plane shown as a solid black line. Datapoints labelled v_1, v_2, and v_3 are the support vectors of this plane. The margin is the space between the two dotted black lines.

Support Vector Machine

A support vector machine (SVM) is a popular machine learning model used for binary classification problems. SVMs can be understood by first visualizing data as coloured points in k-dimensional space, where the colour denotes the class of the data point and the value on each of the k axes is the value of each of the k predictor variables (Figure 2.9a). The goal in training an SVM is to find the maximum-margin separating plane, which is the plane that separates points of different colours such that there is a maximum amount of space between the plane and the points (Figure 2.9b). Visually, if the plane has a thickness, the goal is to find the thickest plane that separates the points. Once the maximum-margin separating plane is found, predictions for new datapoints depend on what side of the plane the new datapoint falls. The reader can learn more about SVMs in the text by Alpaydin [17].

Support vector machines get their name from the fact that the maximum-margin separating plane can be defined in terms of a subset of the datapoints in the training dataset, called the support vectors. The decision to classify a new data point as belonging to either the positive or negative class is made based on the result of a linear combination of the inner-products between the new data point and each of the support vectors. For example, in Figure 2.9b, the datapoints labelled v_1, v_2, and v_3 are the support vectors of the maximum-margin separating plane; the maximum-margin separating plane is defined completely by these 3 datapoints.

In addition to linear separating planes, support vector machines can classify using non-linear surfaces by mapping the training data into another space with a special function and then performing the training procedure in this new space. This mapping can be achieved implicitly by substituting the inner-product in the decision function with another function. These functions are called kernels.

For real classification problems, separating all of the datapoints into their respective classes with a plane may be impossible. For this reason, most SVMs use a soft-margin, which allows some of the training data to be misclassified. The parameter C is introduced to control the balance between (1) the goal of making the margin as large as possible, and (2) the soft-constraint that all training data be on the correct side of the plane and outside of the margin [22]. Lowering the value of C softens the constraint, but the value of C must remain positive.

Radial Basis Function

One example of an SVM kernel is a radial basis function (RBF). A RBF is shown in Equation 2.1, where γ is an adjustable parameter and $\mathbf{x}$ and $\mathbf{x}'$ are two vectors. Intuitively, the RBF kernel is a similarity function that maps pairs of vectors into the interval $[0, 1]$. The RBF kernel is at a maximum (equal to 1) when the pair of vectors are equal. As the distance between the vectors increases, the RBF kernel decays to 0. γ controls how fast a RBF decays and must be positive.

$$K(\mathbf{x}, \mathbf{x}') = \exp(-\gamma||\mathbf{x} - \mathbf{x}'||) \tag{2.1}$$

When fed a vector as input, a RBF-SVM classifies the vector through the following procedure: For each support vector, the RBF-SVM calculates the distance to the input vector, multiplies this distance by $-\gamma$, then takes the exponential of the result. The resulting term is multiplied by the weight of the support vector. Finally, the RBF-SVM sums the resulting terms and adds a constant term, which was also learned during training. If the sum is positive, the RBF-SVM classifies the input as positive; otherwise, it classifies the input as negative.

Strings, Subsequences, and the String Subsequence Kernel

A string is a sequence of symbols from a predefined alphabet. For example, a DNA sequence is a string that is made from the symbols A, T, G, and C. A substring of a string s is a string that can be made into s by adding symbols to either end. A subsequence of a string s is a string that can be made into s by adding symbols anywhere in the string. For example, consider the string s = GATTACA. ATTA is a substring of s but ATTC is not; however, ATTC is a subsequence of s.

In the context of molecular biology, DNA or amino acid sequences can be represented by strings. A substring of a string s can therefore be thought of as an ungapped local alignment on s that does not contain mismatches. Likewise, a subsequence can be thought of as a local alignment on s that does not contain

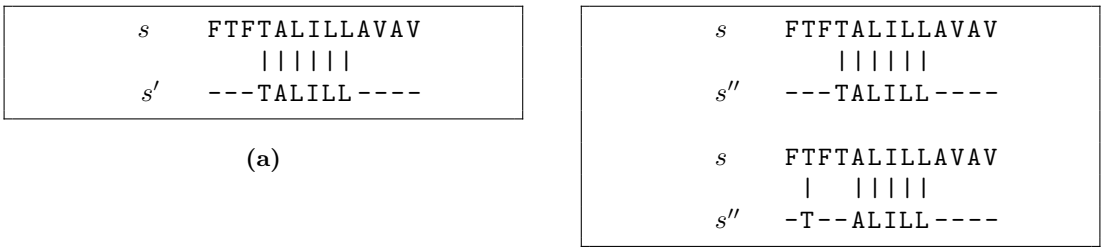


Figure 2.10: Substrings and subsequences in a molecular biology context. (a) A substring s' of s is an ungapped local alignment of s' onto s that contains no mismatches. (b) A subsequence s'' of s is a local alignment of s'' onto s that contains no mismatches and only contains gaps in the s'' part of the alignment.

mismatches and only contains gaps in the subsequence part of the alignment. Examples of a substring and subsequence in this context are shown in Figure 2.10.

The string subsequence kernel (SSK) is a more exotic SVM kernel because it operates on strings instead of vectors [23]. Intuitively, the SSK is the number of subsequences shared between two strings and weighted by the number of gaps in the shared subsequences. The SSK kernel is parameterized by two parameters, the decay λ and the subsequence length n. Intuitively, the decay λ determines how much shared subsequences are penalized for containing gaps.

Artificial Neural Network

An artificial neural network (ANN) is a machine learning tool composed of nodes and edges. An example of a small ANN is shown in Figure 2.11. The nodes in an ANN are processing units which sum together signals from adjacent nodes, apply a function, and send the resulting signal to nodes further down the network through its outgoing edges. The edges in an ANN not only connect node outputs to node inputs, but also multiply the signals they carry by a weight.

In feedforward ANNs (the kind of ANN used in this thesis) nodes are organized into layers, where the first layer contains nodes that receive the input, and the final layer contains nodes that produce the output. The layers in between the input layer and output layer are called hidden layers. Every node in layer i is connected to every node in layer $i + 1$. In addition, each layer has a bias node capable of shifting the entire signal up or down. Figure 2.11 shows an example of a feedforward ANN.

The goal of training an ANN is to find values for the edge weights that minimize the prediction error of the ANN with respect to the training dataset. This minimization problem has no direct mathematical solution, but a numerical procedure called gradient descent can find values that are locally optimal.

Gradient descent requires initial values for the edge weights. The initial values may be supplied by the user and drastically affect the outcome. Using the initial values, the direction of steepest descent is determined by taking the derivative of the training error with respect to each edge weight. Each edge weight is then adjusted

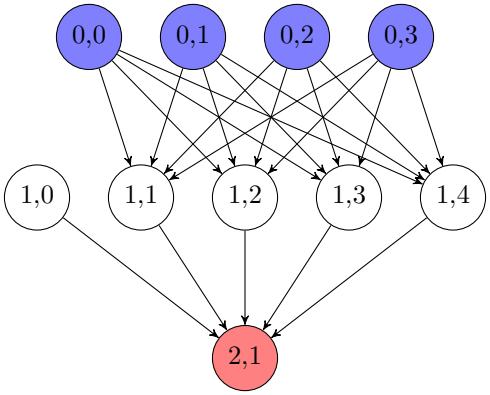


Figure 2.11: An example of a small artificial neural network. Nodes are labelled x, y where x and y denote the layer and node, respectively. By convention, layer 0 is the input layer and node 0 is the constant bias. The nodes in the input layer (shown in blue) take on the values of the predictor variables. The signal propagates down the edges to adjacent nodes. At each edge the signal is multiplied by a weight. At each node the signals are added together, transformed with the activation function, and sent through the outgoing edges. Eventually the signal reaches the output node (shown in red) which predicts the value of the response variable.

to move in this direction by a certain amount. The size of this step is determined by the learning rate. The gradient descent process is often compared to the trajectory of a ball that has been placed randomly on a curved surface. The ball has a random initial position (the initial edge weights) and the elevation of the ball (the training error) is determined by the surface (the training error). The ball moves in the direction of steepest descent, eventually stopping at the bottom of the basin. By analogy, the edge weights move in the direction of steepest descent (with respect to the training error) and stop when the edge weights reach values where any movement, no matter the direction, increases training error. Optionally, a momentum may also be introduced to allow the ball to roll against the direction of descent and jump from one basin to another. The reader can learn more about ANNs in the text by Bishop [24].

Naive Bayes Network

A naive Bayes network is a simple machine learning model in which every predictor variable X is assumed to be conditionally dependent on the response variable Y but independent of all the other predictor variables. Naive Bayes networks are trained by estimating the conditional probability ($P(X|Y)$) for each predictor variable using maximum likelihood estimation [19]. A trained naive Bayes network is used for prediction by applying Bayes' rule (Equation 2.2). The graph of a naive Bayes network is shown in Figure 2.12.

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)} \tag{2.2}$$

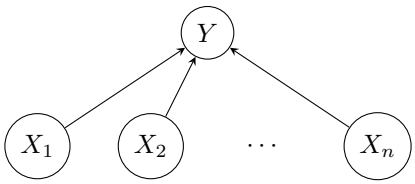


Figure 2.12: A naive Bayes network. Nodes denote variables and arrows denote conditional dependencies between variables.

2.4.6 Related Machine Learning Applications

Machine learning has many fruitful applications in the field of molecular biology. One example is the detection of CpG islands with hidden Markov models [25]. Another example is the prediction of protein secondary structure from protein sequence using artificial neural networks [26]. A third example is the prediction of interacting protein pairs using random forests [27]. Machine learning has also been used in drug development for screening millions of drug candidates for those that are likely to fail [28].

2.5 Statistical Methods

Statistical tests allow one to support or refute hypotheses by deferring to statistical probabilities.

2.5.1 Wilcoxon Signed-Rank Test

The Wilcoxon signed-rank test is a non-parametric (i.e. distribution-free) test that can be used to compare the medians of two related samples. The samples must be related in the sense that there is a natural way to pair the data. An example of two such samples arises when a medical intervention is being tested on a patient group and measurements are taken before and after the intervention. In this example, the pre-intervention measurements form one sample, and the post-intervention measurements form the other. In addition, the measurements in both samples can be paired according to the patient they came from. Pairing the samples has the advantage of controlling for variables that differ between patients.

Using the example of the previous paragraph, the Wilcoxon signed-rank test helps answer the question "are the measurements taken after the intervention consistently higher (or lower) than the measurements taken before the intervention?" [29].

2.5.2 Friedman Test

If the data can be tabulated like the example shown in Table 2.3, the Friedman test can be used to test the hypothesis that the observations in one or more of the rows are statistically higher (or lower) than the other rows, while controlling for variables that differ between columns. The Friedman test is similar to the Wilcoxon signed-rank test except that the Friedman test is not limited to pairs of datapoints. The drawbacks

Table 2.3: The grades receieved by 3 students in the subjects math, science, and english. The Friedman test could be used to test the hypothesis that one or more students received statistically higher (or lower) marks than the others.

Student	Math	Science	English
A	65	50	95
B	60	70	90
C	80	85	60

of the Friedman are that (1) it is less sensitive than the Wilcoxon signed-rank sum test (i.e. more likely to falsely accept the null hypothesis) and (2) it does not indicate which row is dominant [29].

2.5.3 Mann-Whitney U test

Like the Wilcoxon signed-rank test, the Mann-Whitney U test can be used for comparing two samples, except that the samples do not need to be paired [29].

2.5.4 Multiple Hypothesis Testing and the Bonferroni Correction

If, instead of a single hypothesis, a group of hypotheses are being tested using a dataset, the probability of falsely rejecting a null hypothesis increases in proportion to the number of hypotheses. Using the example from Section 2.5.1, such a situation might arise if, instead of testing one intervention, multiple interventions are being tested.

One way to account for the increased chance of error associated with multiple hypotheses is to use the Bonferroni correction [29]. The Bonferroni correction ensures that the change of falsely rejecting a single null hypothesis is less than the significance level α.

2.6 Software

2.6.1 MUSI

Clustering is one example of unsupervised machine learning. The goal of clustering is to find groups (i.e. clusters) of datapoints that are similar to each other but different from datapoints in other groups. An example application of clustering is classifying patients diagnosed with a particular disease into disease subtypes based on clinical observations and disease outcomes.

MUSI (Multiple Specificity Identifier) [30] is a software package that performs clustering on a set of short peptide sequences. The MUSI algorithm first performs a mulitple sequence alignment (Section 2.3.1) on the peptide sequences before selecting a set of PWMs (Section 2.3.2) to represent the MSA. The selection procedure returns the maximum number of PWMs (fit using expectation maximization) that satisfy the

MUSI criteria: (1) each PWM (i.e. cluster) has enough sequences and (2) no two PWMs are too similar. Each PWM returned by the selection procedure corresponds to a single cluster.

Chapter 3

Research Goals

Next-generation sequencing has enabled scientists to examine the DNA of biological systems (e.g. humans, eukaryotes, bacteria, viruses, and environments) with unprecedented detail. With the high volume of information produced by NGS, however, comes the oft-cited challenge of storing, organizing, processing, and analyzing this sequence data. Comprehending such high volumes of data is impossible to do manually. Not excluded from these difficulties is the area of antibody phage display, where phage populations containing millions of genetically distinct clones can be characterized in a single NGS run, allowing researchers equipped with the right tools to perform an in-depth analysis of the sequence landscape present within the population.

The deluge of sequence data produced by applying NGS to antibody phage display presents a unique challenge to the computational biologist. Each sequenced sample provides a detailed snapshot of the make-up of the contained phage; but the factors that account for changes in the population observed between snapshots is understood primarily on a qualitative level. While this high-level understanding coupled with the researcher's own experience has been sufficient to bring about major advances in the field [31], there is still opportunity to bring to bear machine learning techniques to this exciting new area.

The aim of this thesis is to explore applications of machine learning to antibody phage display in the following context: Given the sequence of a clone and a target, can the presence or absence of a clone after the 5th panning round be predicted? As the presence or absence of a clone in the third, fourth, and fifth rounds of panning is one of the first criteria the analyst uses in selecting clones for further experimentation, this prediction task is a good candidate for machine learning.

The aim of this thesis was distilled into three research goals, which are presented in the following three sections (Sections 3.1 to 3.3).

3.1 Compare the Performance of Various Machine Learning Techniques in Application Area

The field of machine learning is rife with techniques for learning complex patterns from data and making predictions based on those learned patterns [17, 24, 19]. No single technique is strictly dominant, but each has areas of application where it excels. The first research goal of this thesis is to develop a general methodology for applying different machine learning models to the prediction task at hand, and then to

use this methodology to compare the classification accuracy of a variety of machine learning models. The general methodology must prescribe methods for (1) identifying and extracting informative features from the CDRH3 sequence data and (2) tuning the hyperparameters of the models.

3.2 Assess the Generality of the Prediction Pipeline

The usefulness of a trained machine learning model depends crucially on its ability to generalize to data not observed in the original training set, a concept called generality. The second research goal is to determine to what extent the best machine learning technique (chosen based on the results from the previous research goal) can generalize to antibody phage display experiments that use different libraries. Such a finding would suggest that the selected machine learning technique can find relationships between a clone's CDRH3 sequence and its outcome that are independent of the library.

3.3 Present a Methodology for Interpreting the Trained Models

Decisions for library construction are made on the basis of experience, intended targets, and resources. Libraries may go through an iterative process of improvement by altering the original specification or construction protocols. Machine learning models vary in their explanatory power; that is, the ease at which their predictions can be understood in terms of the material inputs. By reverse engineering the learned classifier in the prediction pipeline, general properties can be derived which correlate with the experiment outcomes and the fate of clones. These properties can be interpreted as a prescription for further specialization of the antibody library into a focused antibody library. The third goal of this thesis is to develop one possible methodology for extracting this information from the learned models.

Chapter 4

Data & Methods

4.1 Work Performed by the Geyer Lab

Using antibody phage display, the Geyer Lab [13] selected clones from library F and library S for affinity to seven targets named Axl, Jagged1, Jagged2, Mer, Notch1, Notch2, and Notch3 (Table 2.2). Screening was carried out in a series of 14 antibody phage display experiments, each experiment using a different library-target combination. To achieve sufficient enrichment of target-binding clones, 5 rounds of panning were used in each experiment. After each round of panning, samples of the resulting phage population were sequenced using an Ion Torrent sequencer.

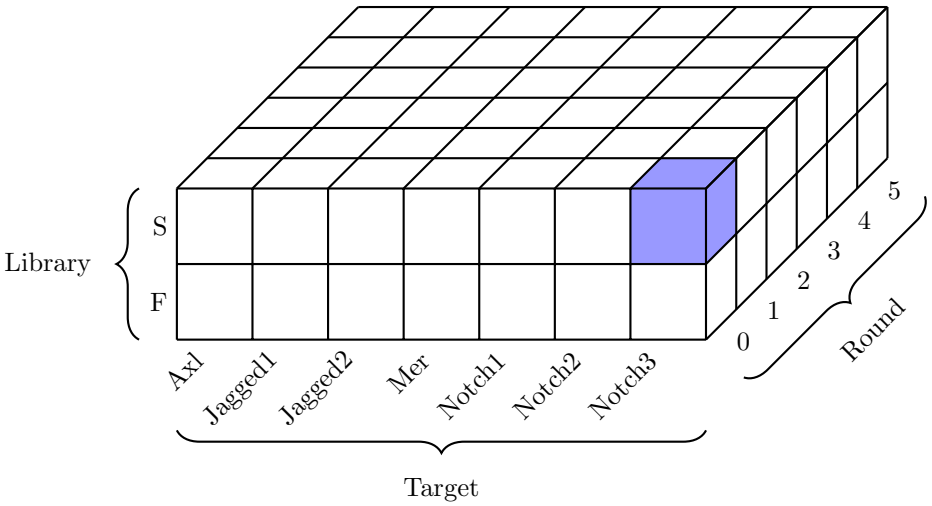
For sequencing, the Geyer Lab [13] amplified the CDRH3 region of selection outputs (phage samples) using designed primers; then performed emulsion PCR on the amplicons using proprietary Ion sphere particles (ISPs); and finally sequenced the enriched ISPs on an Ion semiconductor chip. The sequences output by Ion Torrent were prepared for data analysis with a sequence of steps that included the removal of reads that diverged significantly from the library specification and reads with a low overall quality.

Because the sequenced samples form the basis of the following analysis, an unambiguous terminology for referring to different collections of them was devised. Samples can be identified uniquely by the library, target, and panning round from which they were collected; therefore, they can be said to form a 3-dimensional array with library, target, and panning round represented on the 3 axes. Figure 4.1 shows this 3-dimensional representation. Using this analogy, when an analysis is described as being performed on the S-Notch3 sample array, samples from S-Notch3 rounds 0-5 are implied (Figure 4.1b).

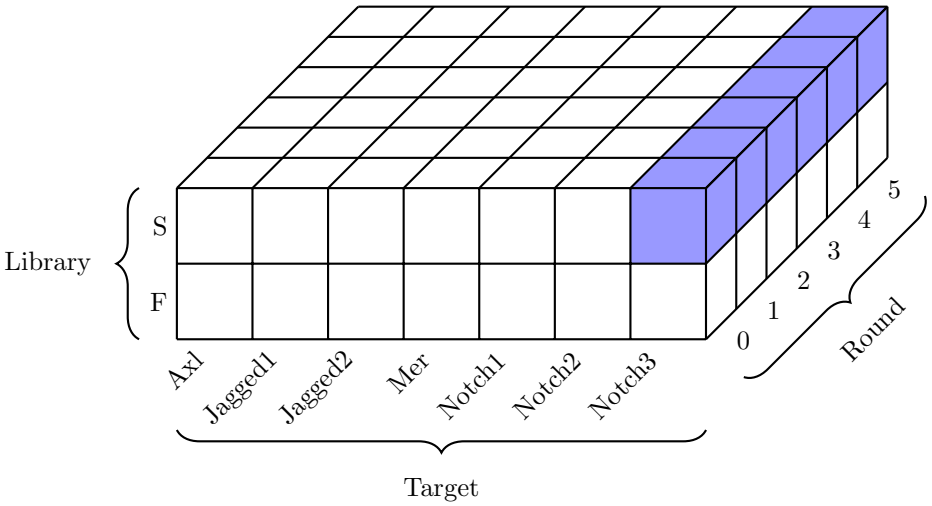
For a typical sample, thousands of sequences were identified, many of which were identical. It was assumed that identical sequences came from identical clones and that the number of identical sequences was proportional to the concentation of that clone.

4.2 Data Organization

The CDRH3 DNA sequences produced by the Geyer Lab were translated into amino acid sequences using the `transeq` program from the EMBOSS package [32]. The resulting protein sequences were inserted into a PostgreSQL [33] table called `reads`. The sequences in the `reads` table were grouped by the sequence, library,



(a) Library S-Notch3-Round 1 sample.



(b) Library S-Notch3 sample array.

Figure 4.1: The phage samples viewed as a 3-dimensional array of cubes. (a) The sample collected following round 5 of panning Library S against Notch3, denoted S-Notch3-5. (b) The samples collected in all Library S versus Notch3 panning rounds, denoted S-Notch3.

target, and round columns; and then counted. These unique entries were inserted, along with their counts, into a table called `clones`.

The reason for managing data with a DBMS like PostgreSQL was two-fold: (1) PostgreSQL ensures that the entered data is valid and (2) querying a flatfile requires that, at worst, the entire file first be read, a time-consuming operation, whereas PostgreSQL stores data in a native binary format which is designed for rapid retrieval of information.

4.3 Feature Extraction

The sequences in the `clones` table were preprocessed to extract a diverse collection of potentially informative features. These features can be divided into 3 groups. The first of these groups contains the amino acid compositions of the 20 amino acids (Section 4.3.1). The second contains the counts of each dipeptide (Section 4.3.2). The third contains a variety of physicochemical properties (Section 4.3.3). After extracting these features, a preliminary analysis was conducted to gain a better understanding of the data.

4.3.1 Amino acid composition

The amino acid composition of a clone was the fraction of the clone's variable CDRH3 sequence made up by each of the 20 possible amino acids. An example calculation is shown in Table 4.1.

Table 4.1: Calculating the amino acid composition for the sequence YYYYVYDFDY.

Amino Acid	Sequence	Composition
D	YYYYVYDFDY	0.2
F	YYYYVYDFDY	0.1
V	YYYYVYDFDY	0.1
Y	YYYYVYDFDY	0.6

4.3.2 Dipeptide counts

The dipeptide counts for a clone were the number of each of the 384 different dipeptides present in the clonal variable CDRH3 sequence. Out of the 400 possible dipeptides, 16 were omitted from this set of features because they were not observed in any of the sequenced samples. A sequence of length l contained $l-1$ dipeptides, so the sum of the dipeptide counts for a clone with a sequence of length l was equal to $l-1$. Calculating the dipeptide counts for the sequence YYYYVYDFDY is shown in Table 4.2.

The motivation for using dipeptide counts instead of fractions was based on an analogy with binding motifs in protein sequences. Let X be a well-known binding motif for target T and let A and B be two proteins that have X in their sequence. Also, assume that the number of amino acids in protein B is half the number in protein A. In this hypothetical scenario, knowing how many times X occurs in A or B is more relevant than whether these proteins bind to T than knowing the fraction of A or B made up by X because the answer does not depend on the size of the protein. Following the analogy, if a dipeptide is viewed as a micro-motif of the CDRH3, which can vary is size, then the question of how many times that dipeptide

occurs in the CDRH3 is more informative than the question of what fraction of the sequence is made up by that dipeptide.

Table 4.2: Calculating the dipeptide counts for the sequence YYYYVYDFDY.

Sequence	Dipeptide
YYYYVYDFDY	YY
YYYYVYDFDY	YY
YYYYVYDFDY	YY
YYYYVYDFDY	YV
YYYYVYDFDY	VY
YYYYVYDFDY	YD
YYYYVYDFDY	DF
YYYYVYDFDY	FD
YYYYVYDFDY	DY

(a) Decomposing the sequence into dipeptides.

Dipeptide	Count
YY	3
YV	1
VY	1
YD	1
DF	1
FD	1
DY	1

(b) Counting the dipeptides.

4.3.3 Physicochemical Properties

In addition, a collection of physicochemical properties were estimated for each clone's sequence. These features included the following eight physicochemical properties, calculated using the R package Peptides (version 1.1.1) [34].

Aliphatic index: From Ikai [35]: "The relative volume of a protein occupied by aliphatic side chains (alanine, valine, isoleucine, and leucine)."

Boman index: The average solubility value of the amino acids in the sequence.

Charge: The net charge of the sequence.

Hydrophobicity: The average hydrophobicity index of the amino acids in the sequence.

Instability Index: An index of how rapidly a protein will degrade. A protein that has an index less than 40 is considered stable (e.g. has a very long half life) [36].

Length: The number of amino acids in the sequence.

Molecular weight: The combined molecular weight of the amino acid sequence.

Isoelectric point: The pH at which the protein has a net charge of 0.

4.4 Comparison of Machine Learning Techniques

4.4.1 Overview

Each clone that was observed after 5 rounds of panning received a label of PERSISTENT and each clone that was observed in the naive library, but not after round 5 received the label TRANSIENT. A panel of classifiers was trained to classify clones as either PERSISTENT or TRANSIENT based on the clonal sequence. The panel of classifiers consisted of a logistic model (LM), random forest (RF), support vector machine (SVM), artificial neural network (ANN), and naive Bayes network (BN). Although by no means exhaustive, this list of classifiers covers many of the popular machine learning classifiers used today. Using Weka [37], each classifier in the panel was trained and tested on the features calculated for each library-target sample array. Because there were 2 libraries, 7 targets, and 5 different classifiers, the analysis consisted of 70 subanalyses ($2 \times 7 \times 5$).

4.4.2 Dataset Preparation

For each experiment, the unique clones observed in round 5 were labelled positive. The positive dataset was then subtracted from the unique clones identified in the naive library (round 0) to form the negative dataset. Because the number of negative clones was much greater than the number of positive clones, the negative dataset was randomly down-sampled until the size of the two datasets matched.

4.4.3 Feature Selection

Generally, the number of classifier parameters increases with the dimensionality of the feature space. A large number of classifier parameters is associated with (1) longer training times, (2) overfitting, and (3) less interpretable models. Because of these issues, dimensionality reduction is highly desirable.

A total of 413 features were extracted from each clone. These features were made up of 8 physicochemical properties, 20 amino acid compositions, and 384 dipeptide counts. Uninformative features were removed from the data to reduce the dimensionality of the feature space without compromising the ensuing analysis.

Two components were needed to find a subset of features (i.e. remove uninformative features): (1) a criterion for evaluating the informativeness of a subset of features, and (2) a search algorithm. The criterion was correlation-based feature selection (CFS) [18] and was described further in Section 2.4.5. CFS penalizes noisy and redundant features when evaluating informativeness. An exhaustive search of feature subsets to find the subset with the maximum CFS criterion was prohibitive because of the large size of the space of feature subsets ($2^{\#\text{-of-features}}$). For computationally tractability, a best-first search was used with a stopping point of 5 consecutive non-improving nodes.

The search algorithm was initialized with the empty set of features and proceeded through the feature subset space by adding one feature at a time. The feature subset search was run on a dataset consisting of 250 positive clones and 250 negative clones from each library-target sample array.

4.4.4 Training and Validation

Each classifier was trained and tested using 10-fold cross-validation on the feature-reduced datasets. The following parameters were used for each classifier and training algorithm used to train that classifier.

Logistic Model: Default parameters were used.

Bayesian Network: Default parameters were used.

Random Forest: The random forest consisted of 100 decision trees. Each decision tree was trained using the entire training dataset. Randomness was injected into the training procedure by randomly selecting $\log_2(\text{num-of-features} + 1)$ features from the feature-reduced dataset prior to training each decision tree. This was the default heuristic used by Weka.

Support Vector Machine (RBF kernel): The support vector machine was equipped with an RBF kernel. The cost C was set to 1 and γ was set to 0.01. Different parameter settings were investigated in the analysis described in Section 4.6.2.

Artificial Neural Network: The number of hidden layers in the artificial neural network was set to 1. The number of nodes per hidden layer was set to $\frac{\text{num-of-features}+2}{2}$, the default heuristic used by Weka. The learning rate and momentum (described in Section 2.4.5) were left at their default values of 0.3 and 0.2, respectively. Weka normalizes (centers and scales to unit variance) features by default prior to training. This was not changed. Different numbers of hidden layers were also investigated as described in Section 4.5.

4.4.5 Quantifying Prediction Performance

The classification accuracy was used for comparing the performance of the classifiers studied in this thesis. The accuracy is the fraction of clones correctly classified (Equation 4.1).

$$\frac{\text{\#-of-correctly-classified-clones}}{\text{total-\#-of-clones-in-dataset}} \tag{4.1}$$

4.5 Improving ANN Classification Accuracy

In an attempt to improve the classification performance of the ANN, different numbers of hidden layers were tried. ANNs with 1, 2, and 3 hidden layers were trained and tested. Figure 4.2 shows the three ANNs that were used.

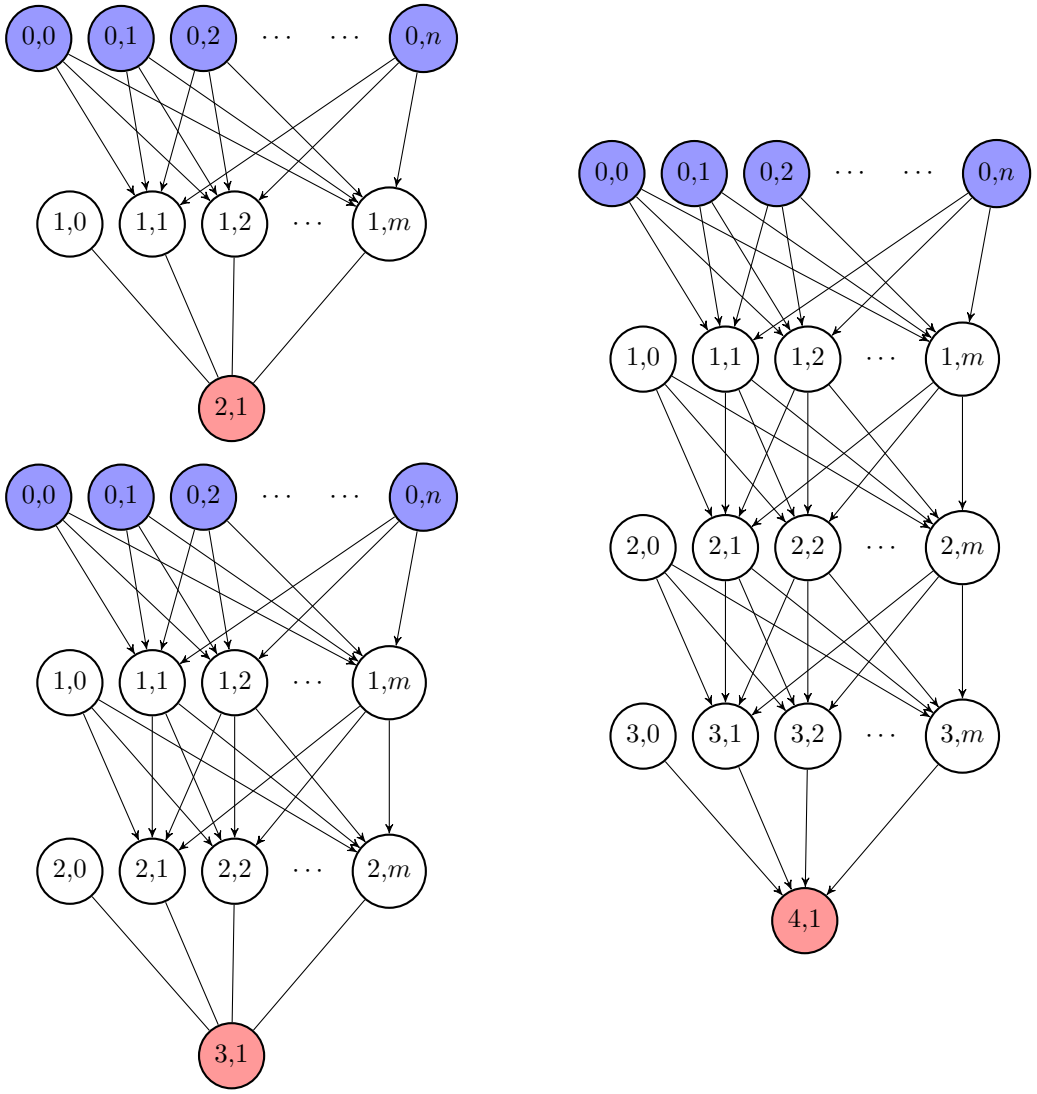


Figure 4.2: Artificial neural networks with 1, 2, and 3 hidden layers. The input layer is coloured red and the output layer is coloured blue. n is the number of inputs to the ANN and m is the number of nodes per hidden layer. By default, Weka sets m to $\frac{n+2}{2}$.

4.6 Improving SVM Classification Accuracy

Two methods of improving SVM prediction accuracy were explored. The first was the introduction of a new type of kernel, called the String Subsequence Kernel (SSK). Whereas the RBF kernel, which was used in the original comparison, operates on numerical vectors, the SSK kernel operates on strings allowing the clonal sequence to be used as input. The second method of improving the SVM prediction accuracy was a gridsearch to determine optimal parameter settings. The gridsearch was performed on both the origin RBF-SVM and the newly introduced SSK-SVM.

4.6.1 String Subsequence Kernel

The input to an SVM equipped with an RBF kernel is vectors in $\mathbb{R}^n$, where n is the number of numerical features describing each data point. The raw data used in this thesis consists of variable length sequences and it was only by extracting various numerical features from these sequences that the sequences could be projected into $\mathbb{R}^n$ so that the RBF-SVM could be used. To avoid having to choose a mapping of CDRH3 sequences to a vector space, which may not be information preserving, a kernel which operates directly on sequences was used. This kernel is called the string subsequence kernel (SSK).

4.6.2 Gridsearch

Gridsearch was used to tune the parameters of the RBF- and SSK-SVMs. Because the computation time of gridsearch scales exponentially with the dimensionality of the grid (i.e. the number of hyperparameters), gridsearch was performed on just two hyperparameters at a time. For the RBF-SVM, the gridsearch was performed on the parameter space formed by C and γ of the RBF-SVM and covering $C = 10^x$ for $x = -5, -4, \ldots, 4, 5$ and $\gamma = 10^x$ for $x = -5, -4, ..., 4, 5$. At each vertex of the grid and for each target, the classification accuracy was measured using 10-fold cross-validation. For the SSK-SVM, the gridsearch was performed on the parameter space formed by λ and n. The search covered $\lambda = 0.1, 0.2, \ldots, 0.9$ and $n = 1, 2, \ldots, 6$. At each vertex of the grid and for each target, the classification accuracy was obtained using 10-fold cross-validation.

4.6.3 Comparing the Tuned SVMs to the Original Classifiers

For both the optimized SVMs and the classifiers tested in Section 4.4, the cross-validation accuracies of each classifier-dataset combination were averaged to obtain a mean accuracy. The mean accuracy of each SVM was then compared to the mean accuracy of the original classifiers.

4.7 Cross-Library Validation of SVMs

In a separate analysis, the tuned RBF- and SSK-SVM were trained to predict clone persistence in one library and then tested on the data from another library to see if the model generalized to a different experimental setting. The library chosen to train the SVMs was library F because experiments by the Geyer lab showed that library F worked well on all of the targets. Data from library S was used to test the SVMs. For each target, an SSK-SVM was trained on the Library F sample array for that target and then tested on the corresponding sample array in Library S. Classification accuracy was measured using 10-fold cross-validation. The same analysis was conducted for the RBF-SVM.

The reasons for choosing the RBF- and SSK-SVM for this analysis and subsequent analyses are manifold: (1) the RBF- and SSK-SVM were among the best performing classifiers in the analyses that was described

in Section 4.4 (results are presented in Section 5.2.5); (2) compared to the ANN, the SVMs had a smaller number of hyperparameters to tune; and (3) compared to RF, the SVMs were easier to interpret; and (4) the SVM training procedure is guaranteed to find the best parameter settings. In addition, the SSK-SVM works with sequence inputs. Using the CDRH3 sequence as input to the SSK-SVM avoided the problem of choosing an intermediate vector representation of the CDRH3 sequence.

4.8 Interpreting the Trained SVMs

The SVMs that were trained on library F were inspected to understand the basis for their predictions. As explained in the background (Section 2.4.5), SVMs make decisions based on a learned subset of training examples called support vectors and a learned set of weights. These support vectors together with their weights will be used to interpret the RBF- and SSK-SVMs.

4.8.1 Interpreting the Trained RBF-SVM

As described in Section 2.4.5, the decision function is a linear combination of the kernel between the input and each support vector; therefore, the function resembles Equation 4.2, where $K(\cdot,\cdot)$ is a RBF, $\mathbf{x}$ is the input, $w_1, w_2, \ldots, w_n$ are the learned weights and $\mathbf{v_1}, \mathbf{v_2}, \ldots, \mathbf{v_n}$ are the learned support vectors.

$$f(\mathbf{x}) = b + w_1 K(\mathbf{x}, \mathbf{v_1}) + w_2 K(\mathbf{x}, \mathbf{v_2}) + \cdots + w_n K(\mathbf{x}, \mathbf{v_n}) \tag{4.2}$$

As described in Section 2.4.5, the RBF is a similarity function that decays exponentially from 1 to 0 as the Euclidean distance between the vectors increases; thus, each RBF in the decision function 4.2 creates a peak or valley (depending on the magnitude of the w_i). The decision function, when visualized in two dimensions with a heatmap, contains "hotspots" at the positive-weighted support vectors and "coldspots" at the negative-weighted support vectors. To visualize the decision function of the trained RBF-SVMs, the decision function was projected onto two dimensions and plotted as a heatmap.

4.8.2 Interpreting the Trained SSK-SVM

As described in Section 2.4.5 the SSK maps a string s to a space of dimension $|\Sigma^n|$, where each dimension is the weighted sum of occurrences of a particular word $w \in \Sigma^n$ as a subsequence in s. The result of the gridsearch over the parameter space of the SSK-SVM (to be described in Section 5.2.4) revealed that a substring length of 2 (i.e. $n = 2$) was optimal, therefore the SSK used in subsequent analyses mapped strings to a space of dimension 400 (i.e. one dimension per dipeptide). For example, the value of dimension AR for sequence s was the number of times the dipeptide AR occurred as a subsequence in s. The decision function of an SVM can be understood in terms of its support vectors, so the support vectors for the SSK-SVM were decomposed into their underlying 400-component vectors consisting of dipeptide occurences. A positive

weight on a dipeptide indicated that the presence of that dipeptide (as a subsequence) in a sequence made it more likely to be predicted as a positive by the SSK-SVM. A negative weight meant the opposite.

4.9 Statistical Methods

In this thesis, non-parametric tests were generally preferred to parametric tests because of certain assumptions required for the latter. One of the assumptions required for parametric tests is that the observations follow a probability distribution (e.g. Gaussian distribution). As will be seen in future sections, no such assumption can be made. The type of non-parametric tests used in this thesis were (1) the Mann-Whitney U test, (2) the Wilcoxon signed-rank test, and (3) the Friedman test. The Bonferroni correction was used wherever multiple hypotheses are tested. Unless indicated otherwise, hypothesis tests used a signficance level of 0.05 (i.e. if the $p < 0.05$, the null hypothesis was rejected).

Whenever a test is used to support an observation like "the median of A is greater than B", the null hypothesis is the opposite; that the median of A is equal to or less than B. This is called a one-tailed test. And whenever a test is used to support an observation like "the medians of A and B are not equal", the null hypothesis is, again, the opposite; that the medians of A and B are equal. This is called a two-tailed test.

The results of statistical tests are given in parenthesis, as close as possible to where they are mentioned. Sometimes only a p-value is given (e.g. $p = 0.05$) and sometimes a confidence interval is given (e.g. M=1.0 CI=[-1.0,3.0]). When the statistic is a confidence interval, the median is denoted by the variable M, and the 95% confidence interval is denoted by the variable CI.

4.10 Hardware and Software

The classifier cross-validation jobs were run on the University of Saskatchdwan Bioinformatics Eldorado server (hostname eldorado.usask.ca, 2 × 8-core Intel Xeon 2.60GHz, 384GB memory) taking advantage of multiple cores where possible. Visualizations were generated on a laptop running Mac OS X.

Data manipulation was carried out at the database level with PostgreSQL and externally using Python with PANDAS and BIOPYTHON [38]. The Weka machine learning workbench provided the routines for feature selection and cross-validation [37]. LaTeX and the GGPLOT2 R package (v2.1.0) were used for visualizing results [39].

Chapter 5

Results & Conclusions

5.1 Preliminary Analysis

The sequenced samples from the 14 experiments were submitted to a preliminary analysis in order to gain a better understanding of the data. The preliminary analysis had three parts: (1) sequence clustering, (2) calculating the diversity over panning round, and (3) computing the physicochemical property distribution.

5.1.1 MUSI Clusters and Sequence Logos

MUSI was used to cluster sequences from the round 5 samples [30] and then to generate sequence logos for each cluster. The sequence logos for Axl and Mer round 5 clusters along with the size of each cluster are shown in Tables 5.1 and 5.2. The remaining sequence logos are shown in Tables A.1 to A.5.

Table 5.1 shows that MUSI identified 10 clusters in sample F-Axl-5 (Table 5.1a). The C-terminal DY was highly conserved across all clusters because the library F specification only allowed DY at the C-terminus. Likewise, positions 12 and 13 were highly conserved across all clusters because the specification only allowed A and G at position 12 and F, L, I, and M at position 13. Particularly striking was conservation of glycine midway along the sequences (positions 6, 7, and 8) in clusters 2, 4, 9, and 11 (from the top).

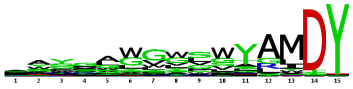
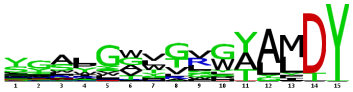
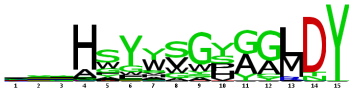
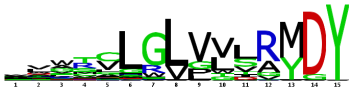
For sample S-Axl-5 MUSI identified only 4 clusters (Table 5.1b). The C-terminal positions were less conserved than in sample F-Axl-5 because the S library specification allowed more residues in these positions. Salient features of the clustering included a conserved N-terminal YGSYY motif and a variable mid-region followed by a low-complexity poly-Y, T, or L region.

For samples F-Mer-5 and S-Mer-5 MUSI identified 6 and 3 clusters, respectively (Table 5.2). In sample F-Mer-5 (Table 5.2a) there was a prominent N-terminal YYPGS motif and sequences appear to be 17 residues long on average. In sample S-Mer-5 (Table 5.2b), sequence length varied considerably, and there was a distinct lack of consensus at the N-terminal half of the sequence.

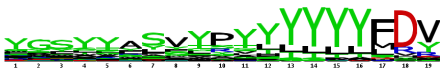
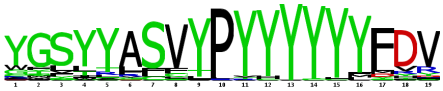
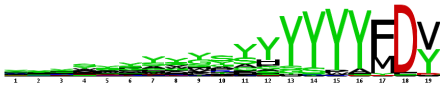
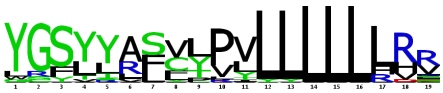
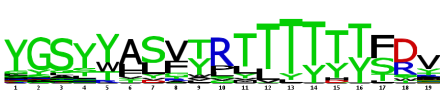
5.1.2 Diversity over Panning Round

The number of distinct clones in each sample is shown in Figure 5.1. From this figure, one can see that the diversity in each experiment trended downward over panning round, which agreed with basic principles

Table 5.1: Sequence logos for clusters in samples F-Axl-5 and S-Axl-5. The first row in each table shows the sequence logo for the entire sample. The column labelled "size" shows the number of distinct CDRH3 sequences falling into each cluster.

Size	Logo
190	
21	
17	
40	
17	
5	
15	
13	
20	
16	
21	

(a) Library F

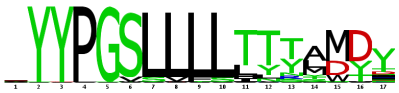
Size	Logo
171	
49	
78	
21	
20	

(b) Library S

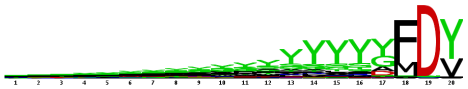
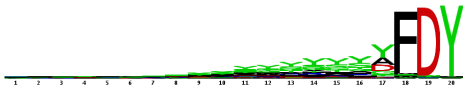
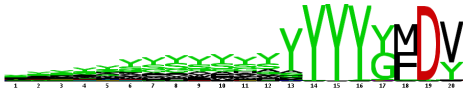
[40]. In library F the initial diversity ($1.008\,56 \times 10^5$) was an order of magnitude greater than library S (1.0801×10^4); however, after the first round of panning, diversity in library F dropped significantly relative to library S. The result of this drop was that, after round one, the diversity of the two libraries were nearly identical.

The greater drop in library F diversity relative to library S may be explained by distinguishing sequence diversity from functional diversity. Whereas sequence diversity is simply the number of distinct CDRH3 sequences, functional diversity only counts CDRH3 sequences that result in functional antibody fragments. It is possible that the great drop in library F sequence diversity was an initial culling of non-functional clones (e.g. clones expressing aberrant antibody fragments).

Table 5.2: Sequence logos for clusters in samples F-Mer-5 and S-Mer-5. The first row in each table shows the sequence logo for the entire sample. The column labelled "size" shows the number of distinct CDRH3 sequences falling into each cluster.

Size	Logo
84	
15	
6	
14	
41	
7	

(a) Library F

Size	Logo
969	
526	
442	

(b) Library S

Experiment S-Mer presents one exception to the general downward trend observed in Figure 5.1. Whereas sequence diversity of most samples decreased by 2 orders of magnitude from rounds 1 to 5, the sequence diversity of S-Mer decreased less than one order of magnitude. As a consequence, the sequence diversity of S-Mer-5 is much higher than other round 5 samples. This result seems to contradict MUSI, which identified only two clusters in S-Mer-5 (Table 5.2b). However, these observations are not necessarily incompatible because the sequence logos in Table 5.2b show a lack of consensus in N-terminal half of the sequences, which means that the clusters subsume a lot of the diversity in these regions.

5.1.3 Distribution of Physicochemical Properties

The physicochemical properties listed in Section 4.3 were calculated using the Peptides R package (version 1.1.1) [34]. Violin plots were created to show the distribution of physicochemical properties in each sample. The most striking of these plots are shown in Figure 5.3. The rest of the plots can be found in Figures A.2 to A.5 of Appendix A. Figure 5.3 shows that there was a distinct trend to the distribution of some physicochemical properties. For example, aliphatic index trended upwards over panning rounds. In the same figure, the length plot, which depicts the length distribution of distinct CDRH3 sequences, shows that the change in length distribution over panning round was different for each target. Additionally it shows that, for the same target, the change of the mean length was similar across libraries. These observations support the hypothesis that targets exhibit strong length preferences during panning [13]. The figure also showed that, in many

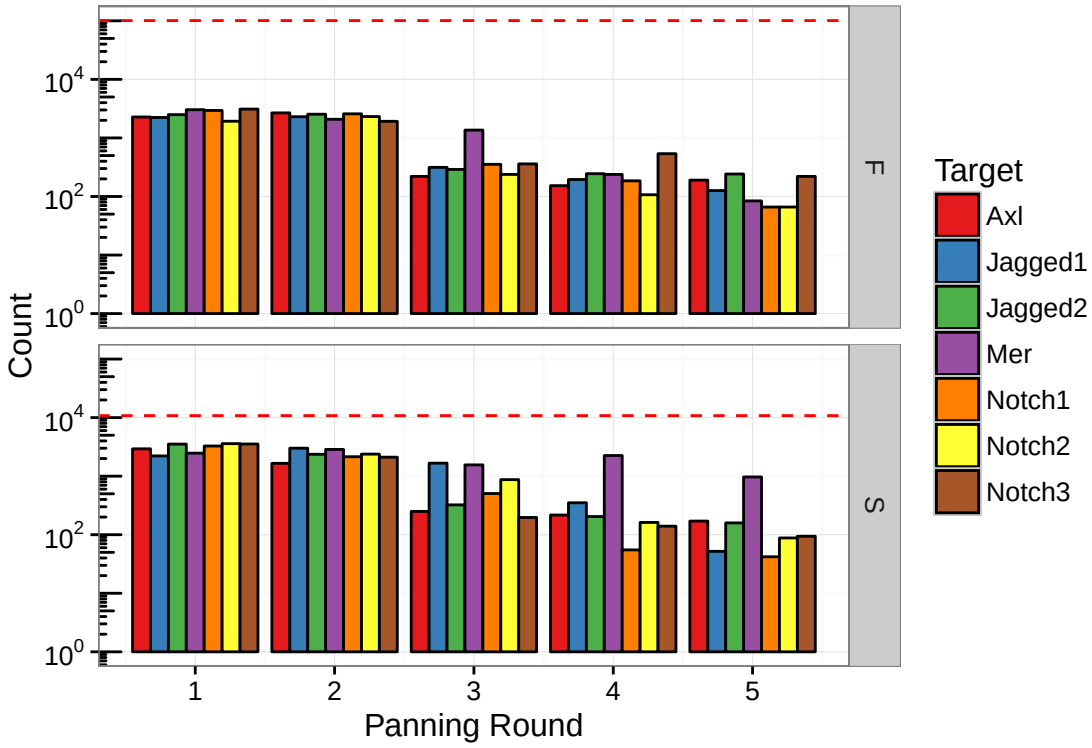


Figure 5.1: Log-scale bar graph of phage sample diversity over 5 rounds of panning for libraries F and S. The panning round is shown on the x-axis and the logarithm of the number of unique clonal sequences is shown on the y-axis. The red line running horizontally across each facet shows the initial library diversity (round 0).

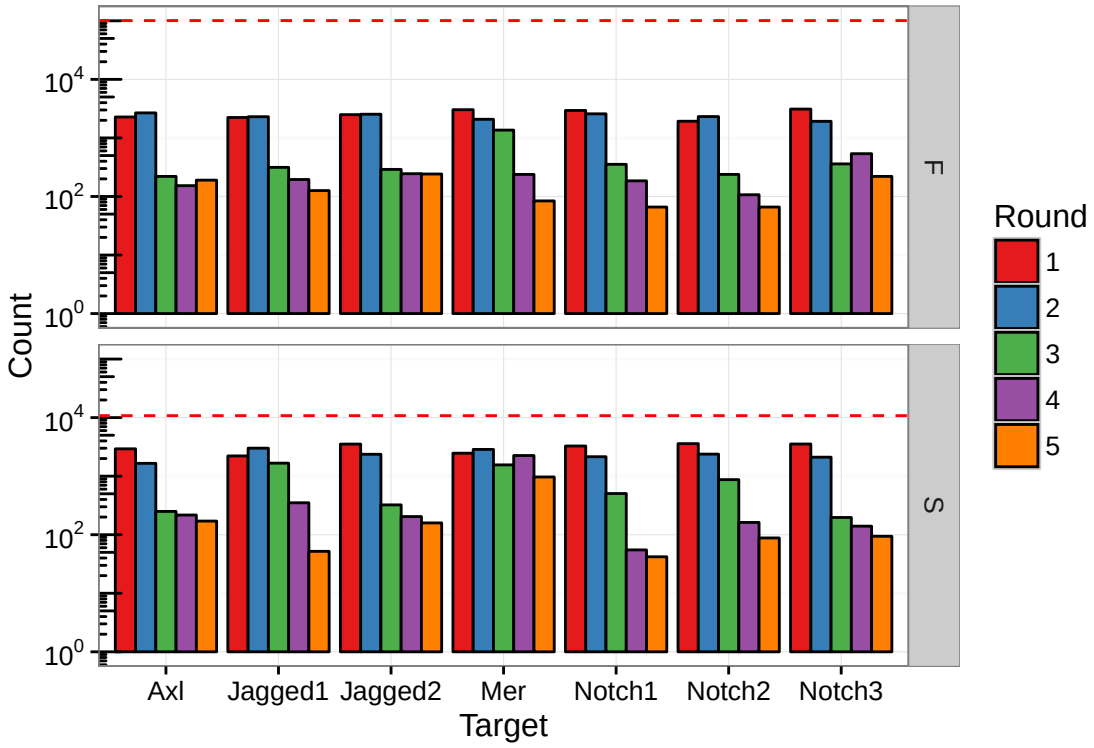


Figure 5.2: Log-scale bar graph of phage sample diversity over 5 rounds of panning for libraries F and S. The target is shown on the x-axis and the logarithm of the number of unique clonal sequences is shown on the y-axis. The red line running horizontally across each facet shows the initial library diversity (round 0).

cases, the length distribution in round 5 was multi-modal, suggesting that clones in round 5 had multiple specificities.

The two clusters identified by MUSI in S-Jagged1-5 (Table A.1) seem to support the hypothesis that two modes correspond to multiple specificities. The sequences in the first cluster identified by MUSI are approximately 11 amino acids long (positions 2-12 in Table A.1b; position 1 largely absent), while the sequences in the second cluster are approximately 6 amino acids long (positions 7-12 in Table A.1b). These lengths, 6 and 11, are the approximate modes of the length distributions shown in Appendix A.3.

The Wilcoxon ranked-sum test was used in conjunction with the Bonferroni correction to test the hypothesis that the median X in round 0 is not equal to round 5, where X was one of the physicochemical properties described in Section 4.3.3. The test found that, using a significance level of 0.01, the null hypothesis was rejected for the following physicochemical properties: aliphatic index ($p = 6.914 \times 10^{-5}$), Boman index ($p < 2.2 \times 10^{-16}$), instability index ($p < 2.2 \times 10^{-16}$), length ($p < 2.2 \times 10^{-16}$), and molecular weight ($p < 2.2 \times 10^{-16}$). Rejecting the null hypothesis meant that the difference of the median X between round 0 and 5 was statistically significant and, therefore, that panning selected clones with different median values of X than those clones present in either library F or library S. The conclusion is that these physicochemical properties have a statistically significant effect on clone outcomes.

5.2 Predicting Persistent Clones

5.2.1 Feature Selection

A total of 413 features were calculated from each CDRH3 sequence. These features were made up of 8 physicochemical properties, 20 amino acid compositions, and 384 dipeptide counts. A subset of these features were selected for inclusion in the machine learning datasets. The motivation and method for selecting these features was laid out Section 4.4.3. The selected features contained the following 20 features: aliphatic index; bowman index; charge; molecular weight; isoelectric point; the relative frequency of amino acids C, D, L, and S; and the absolute frequency of dipeptides DY, ER, LG, LL, LR, NW, QA, QG, QS, TT, and VL.

5.2.2 Cross-Validation

As explained in Section 4.4, a panel of five classifiers was trained on the feature-reduced datasets arising from Section 5.2.1. The task was to classify clones as PERSISTENT or TRANSIENT based on their CDRH3 sequence. The panel consisted of a random forest (RF), artificial neural network (ANN), logistic model (LM), support vector machine with an RBF kernel (SVM), and naive Bayes Network (NB).

The feature-reduced dataset arising from Section 5.2.1 was divided into 14 datasets according to which experiment the data had originated from (i.e. the library and target that were used). Each of these 14 datasets were then balanced by randomly down-sampling the negative dataset until its size was equal to the

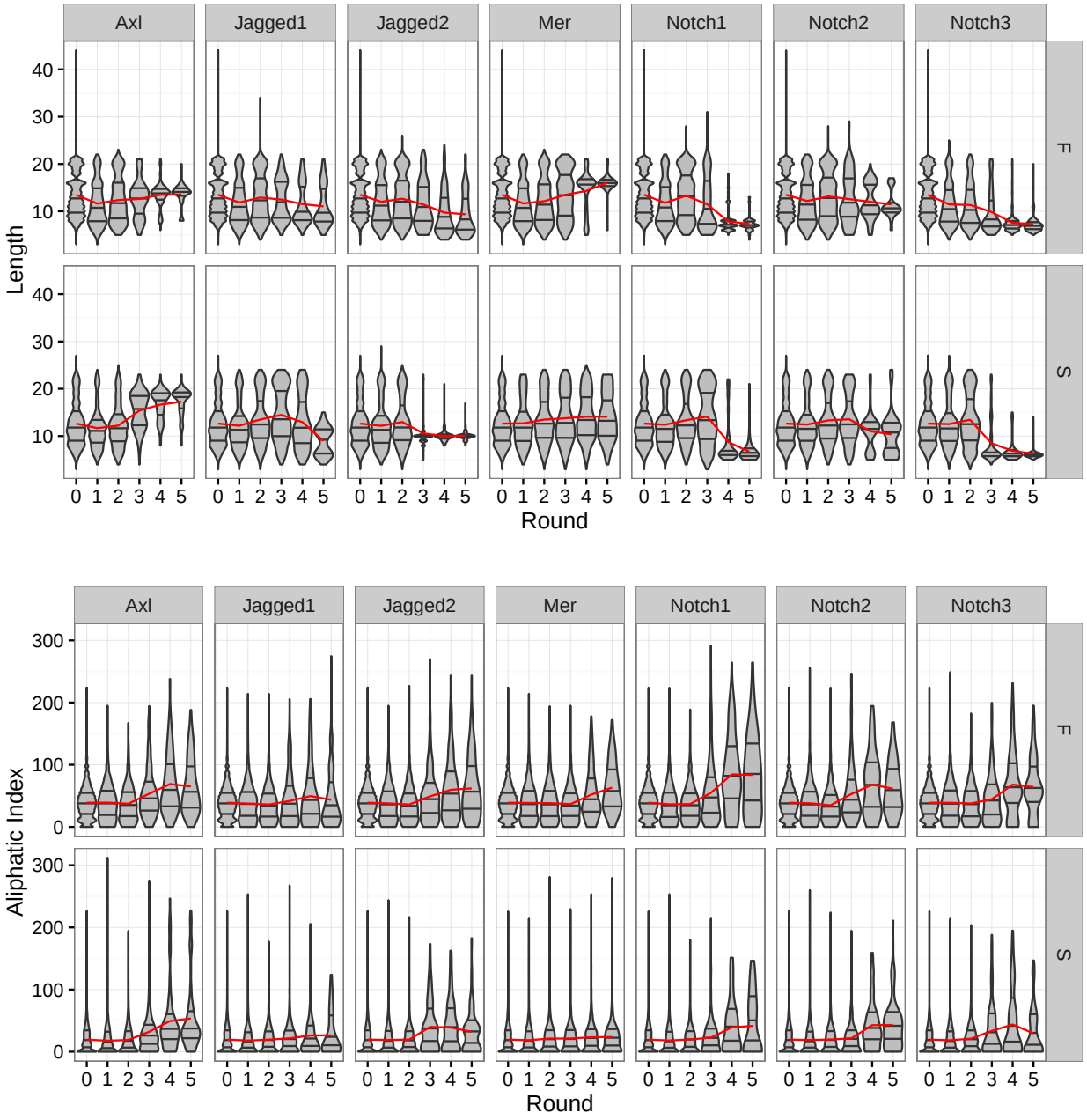


Figure 5.3: Violin plots of the length and aliphatic index across the set of distinct clones identified in each sample. Within each facet of the grid, the area of the violins is held constant; however, across facets, the area of the violins is proportional to the number of distinct clones identified in the experiment (i.e. library-target combination). Horizontal lines within the violins show the 25%, 50%, and 75% quantiles. The red line tracking across each series of violins shows the mean value in each sample.

Library	Target	Size (Rebalanced)
F	Axl	380
	Jagged1	252
	Jagged2	484
	Mer	168
	Notch1	132
	Notch2	132
	Notch3	440
S	Axl	342
	Jagged1	104
	Jagged2	318
	Mer	500
	Notch1	84
	Notch2	176
	Notch3	188

Table 5.3: The number of datapoints in each balanced dataset.

size of the positive dataset. The size of the resulting datasets are shown in Table 5.3. Classifiers were then trained and tested on each of these 14 balanced datasets using 10-fold cross-validation. The classification accuracy was calculated for each left-out fold to estimate the performance of each classifier on unseen data. Figure 5.4 shows the distribution of the accuracy achieved by each classifier-dataset pairing. The average accuracy of each classifier on each of the datasets is shown in Table A.7.

Figure 5.4 shows that, with the exception of the LM and perhaps the NB, the accuracy achieved by the classifiers was comparable. The best one appeared to be the random forest classifier. A one-tailed Wilcoxon signed-rank test showed that, using a significance level of 0.05 (0.0125 with the Bonferroni correction), the RF classifier is at least better than the LM ($p = 7.5 \times 10^{-7}$) and NB (7.2×10^{-4}), but not necessarily better than the ANN (0.047) or RBF-SVM (0.112). A similar test was carried out and confirmed that the LM was the worst classifier ($p = 8.4 \times 10^{-5}, 1.7 \times 10^{-6}, 6.8 \times 10^{-6}, 7.5 \times 10^{-7}$, respectively).

Figure 5.5 shows a multi-faceted boxplot that compares the classification accuracy obtained on each dataset. Seen in this figure is the poor performance of classifiers on the S-Mer dataset relative to the other datasets. A one-tailed Wilcoxon signed-rank test failed to support this observation. In fact, because of the small sample sizes, the Wilcoxon signed-rank test could not, theoretically, produce p-values low enough to reject the null hypothesis because only the ranks of datapoints are considered, not the degree of difference. To account for the degree of difference, a one-tailed t-test using a significance level of 0.05 (0.0038 after Bonferroni correction) was used. The t-test revealed that, assuming a Gaussian distribution, the accuracies achieved

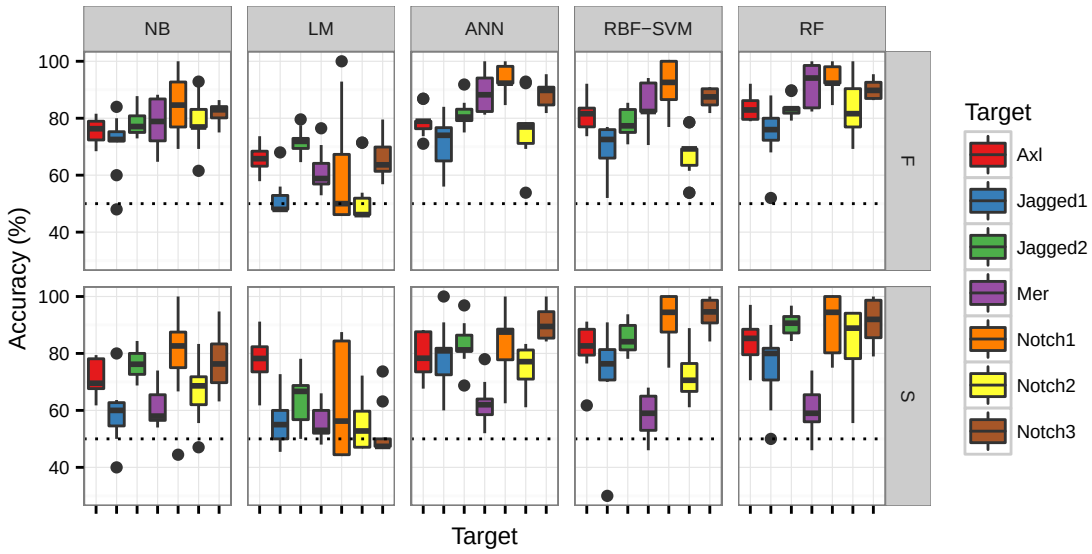


Figure 5.4: Boxplots showing the distribution of the accuracy achieved with each classifier on each of the 14 balanced datasets. The solid circles show outliers. The dotted line running across each plot marks the classification accuracy of a random classifier.

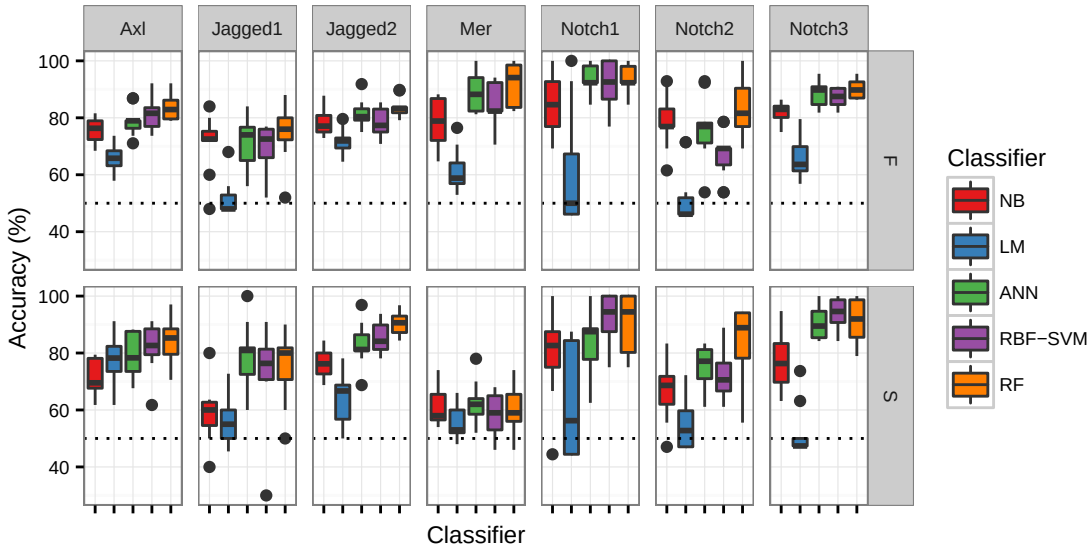


Figure 5.5: Boxplots showing the same thing as Figure 5.4, but with the subplots arranged by target instead of classifier.

on S-Mer were statistically lower than F-Axl ($p = 8.0 \times 10^{-4}$), S-Axl (2.2×10^{-4}), F-Jagged2 (2.6×10^{-5}), S-Jagged2 (1.9×10^{-3}), and F-Notch3 (3.0×10^{-3}), but not F-Jagged1 (0.058), S-Jagged1 (0.043), F-Mer (9.3×10^{-3}), F-Notch1 (0.011), S-Notch1 (0.066), F-Notch2 (0.066), S-Notch2 (0.042), and S-Notch3 (0.033).

5.2.3 Improving ANN Performance

Artificial neural networks can be classified based on the structure of the network formed by their nodes and edges. Feedforward ANNs (described in Section 2.4.5) have nodes organized into layers and edges connecting one layer to the next. One simple way to vary the architecture of a feedforward ANN is by changing the number of hidden layers in the network. Hidden layers were described in Section 2.4.5 and the specific methodology in Section 4.5. The cross-validation procedure used for comparing the panel classifiers was used to determine the affect of adding more hidden layers to the ANN. The accuracy was plotted in Figure 5.6.

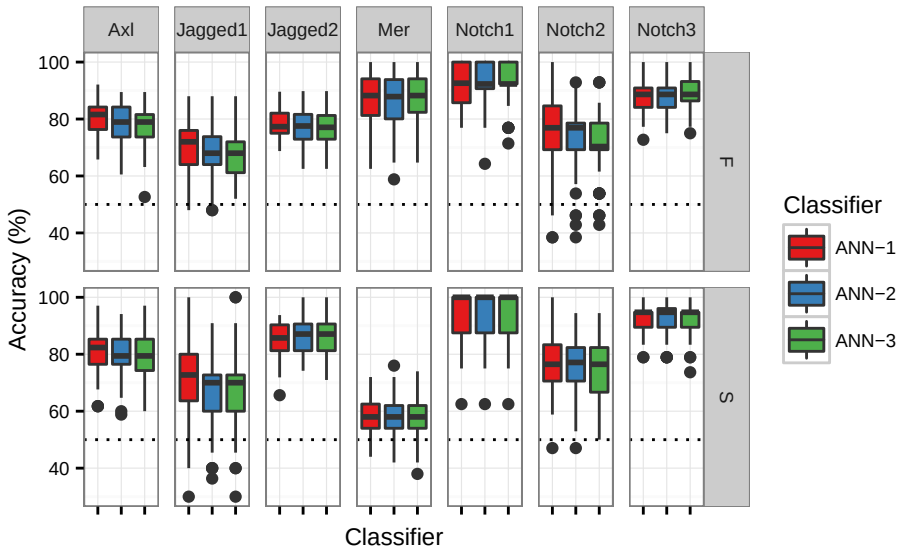


Figure 5.6: Boxplots showing the performance of the artificial neural networks with 1, 2, and 3 hidden layers. The solid circles represent outliers. The dotted line running horizontally across each plot marks the classification accuracy of a random classifier.

Looking at Figure 5.6, it appears that additional layers did not offer any noticeable improvement. The hypothesis that the performance of any one of the ANNs was statistically different from another is rejected by the Friedman test using a significance level of 0.05 ($p = 0.095$).

Across libraries, it appears that the only difference in accuracy occurred on the Mer dataset. Accuracy for the library S Mer dataset was significantly worse than for the library F Mer dataset, and only slightly better than the accuracy achieved with random classification.

5.2.4 Optimizing SVM Parameters

As explained in Section 4.6.2, a gridsearch was used to tune the hyperparameters of the RBF-SVM and SSK-SVM for the prediction task set out in Section 4.4.1. At each vertex of the grid the average classification accuracy achieved using 10-fold cross-validation is shown in Figure A.1. The average accuracy at each vertex across all datasets is shown in Figures 5.8a and 5.8b. In these figures, there appears to be a boundary on each grid, past which classification accuracy drops significantly. From the averaged gridsearch results, the parameters with the highest accuracy were used in all of the subsequent SVM analyses. Using this criteria, the parameters chosen were $C = 10$ and $\gamma = 1$ for the RBF-SVM and $n = 2$ and $\lambda = 0.6$ for the SSK-SVM.

Of all the hyperparameters tested during gridsearch, the RBF-SVM hyperparameters $\gamma = 1$ and $C = 10$ resulted in the highest classification accuracy. As explained in Section 2.4.5, γ is a parameter of the RBF and the RBF kernel acts as a similarity measure between vectors. The similarity measure is one that decays exponentially as the distance between two vectors increases. Intuitively, the parameter γ is the sensitivity of the similarity metric to small changes in the distance. For a large γ, the difference between two vectors is multiplied giving a low measure of similarity. Conversely, for a small γ, the difference between two vectors is scaled down leading to a higher measure of similarity. The affect on a RBF of changing the original $\gamma = 0.01$ to the tuned value $\gamma = 1.0$ is shown in Figure 5.9.

5.2.5 Comparing the Tuned SVMs to the Original Classifier Panel

Figure 5.10 shows that that the tuned SVMs (denoted by a trailing "(t)") compared favourably with the original classifiers; however, a Wilcox ranked-sum test on each pair of the top classifiers (ANN, RBF-SVM, RF, RBF-SVM (t), and SSK-SVM (t)) failed to identify a statistically significant difference between the median accuracies achieved within this group. The p-values for these tests are shown in Table 5.4. A p-value less than 0.005 was required to reject the null hypothesis due to the Bonferroni correction. This corresponded to a significance level of 0.05.

Figure 5.11 and Figure 5.12 shows more clearly the change in mean accuracy after switching from one of the original classifiers to the tuned SVMs. Figure 5.11 shows that the tuned RBF-SVM has nearly the same performance of the RBF-SVM with default parameters. Figure 5.12 shows that the SSK-SVM performs better than the RBF-SVM in most cases and better than all of the classifiers on the Axl and Notch2 datasets.

The average accuracy of each classifier (including the tuned SVMs) on each of the datasets is also shown in Table A.7.

5.2.6 Conclusions

For many of the machine learning methods used in this thesis, the input had to be in the form of a numerical vector. It was found that extracting an array of CDRH3 chemical properties (i.e. the physicochemical, amino acid, and dipeptide properties) was one way of encoding the CDRH3 sequences as a vector for subsequent use

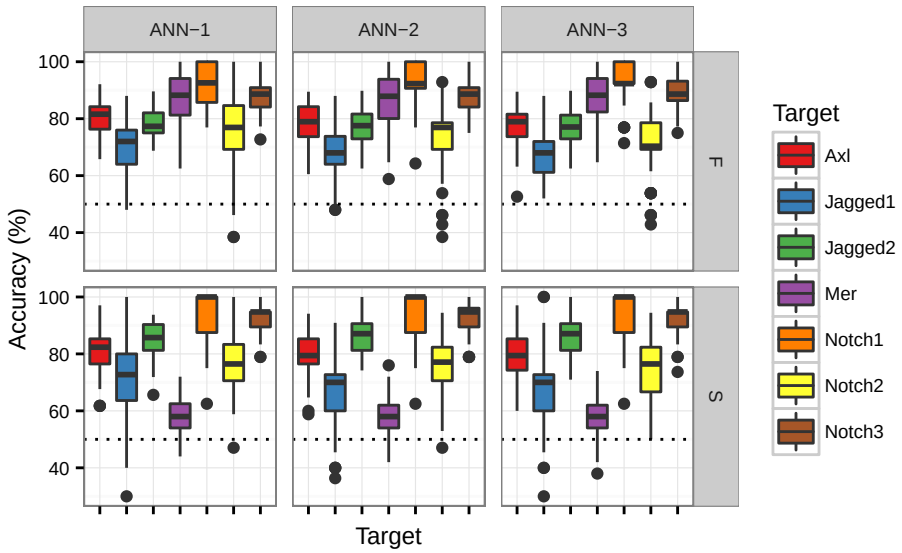
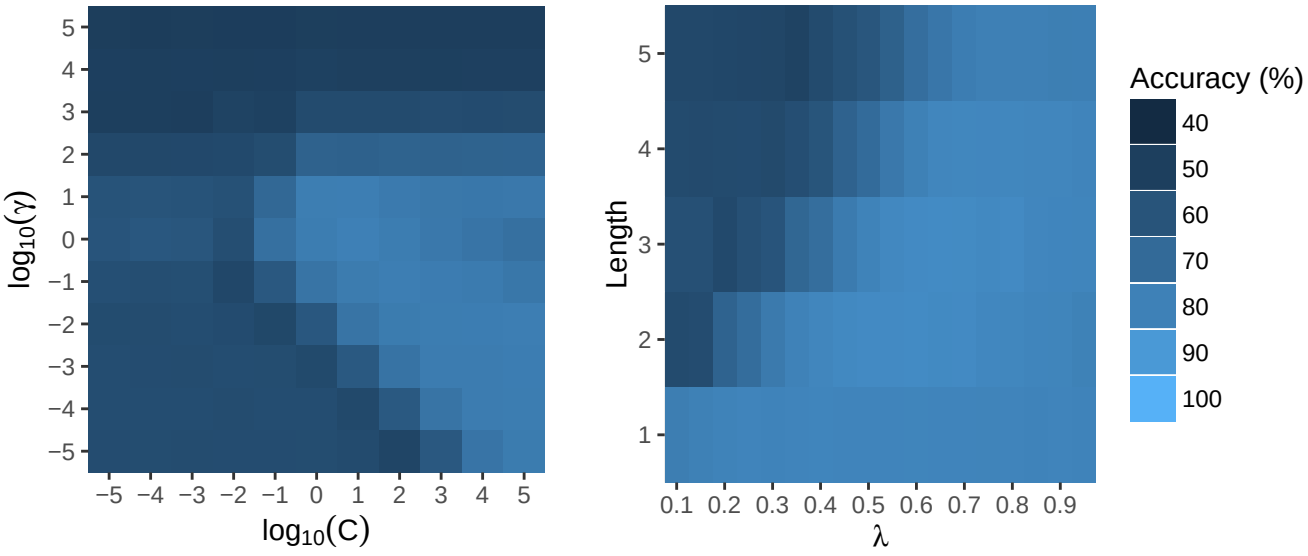


Figure 5.7: Boxplots showing the same thing as Figure 5.6, but with subplots arranged by target instead of classifier.



(a) RBF-SVM **(b)** SSK-SVM

Figure 5.8: Average performance of the RBF- and SSK-SVM using different parameters.

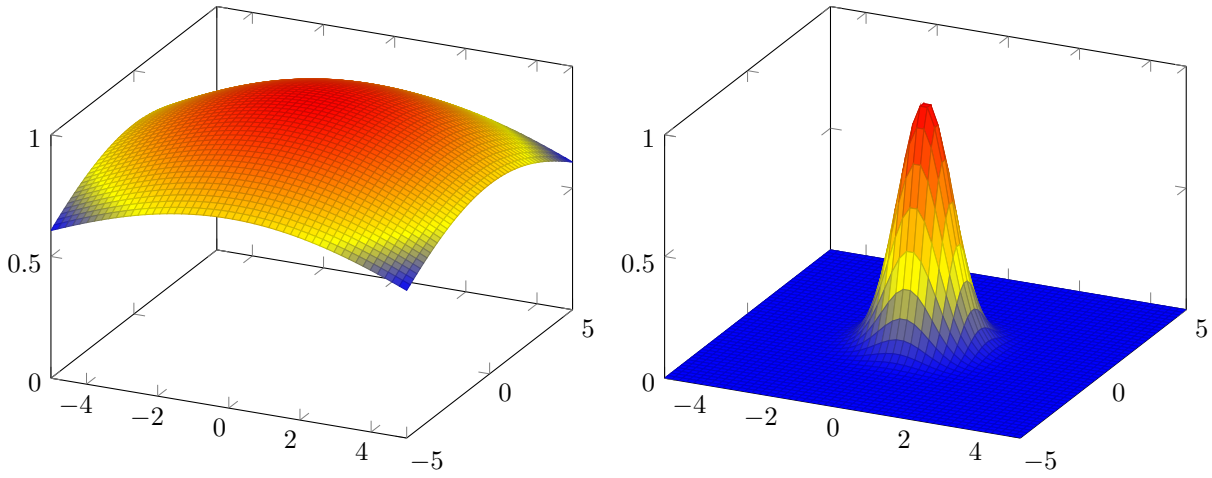


Figure 5.9: A comparison of the effect of the γ parameter on the RBF kernel when one of the vectors is held at the origin. (Left) The value of the RBF kernel over two dimensions using the original setting of $\gamma = 0.01$. (Right) The value of the RBF kernel over two dimensions using the tuned value of $\gamma = 1.0$. The tuned value of γ is much more sensitive to the distance between vectors.

	ANN	RBF-SVM	RF	RBF-SVM (t)	SSK-SVM (t)
ANN	1.00	0.96	0.09	0.55	0.29
RBF-SVM	0.96	1.00	0.22	0.70	0.25
RF	0.09	0.22	1.00	0.06	0.60
RBF-SVM (t)	0.55	0.70	0.06	1.00	0.16
SSK-SVM (t)	0.29	0.25	0.60	0.16	1.00

Table 5.4: p-values from the Wilcoxon ranked-sum test comparing pairs of classifiers. Only the classifiers with the best classification accuracy were tested. None of the p-values were less than 0.005; therefore, none of the tests rejected the null hypothesis.

with these machine learning tools. Moreover, it was determined that such an encoding retains information that is useful for predicting clone persistence.

By comparing various machine learning models, ANNs, SVMs, and RFs were shown to be capable of achieving classification accuracy significantly better than achieved by chance. Moreover, the difference in accuracy between the models was not statistically significant. Any of these models are thus suitable candidates for further study of machine learning applications to antibody phage display. The SSK-SVM, which accepts sequences as input thus eliminating the need to choose a vector encoding scheme, are particularly well-suited to this application domain.

5.3 Generalizing the SVM Models

The utility of a machine learning model hinges on its ability to predict data not observed in the training set. This ability is called generality. To test the generality of the RBF- and SSK-SVM, the SVMs were trained

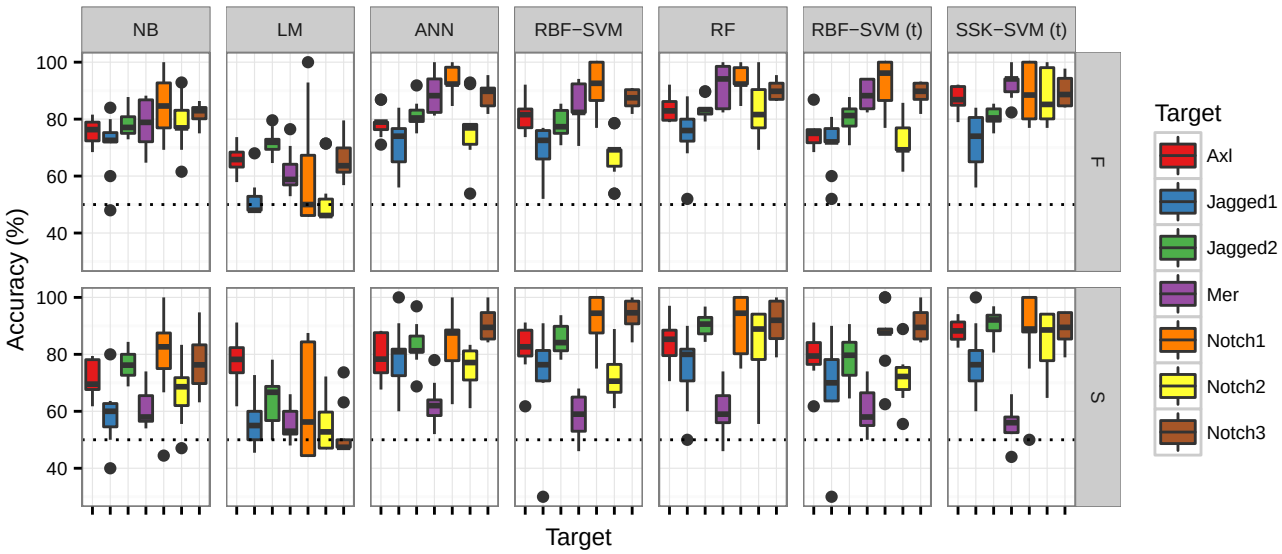


Figure 5.10: Boxplots showing the performance comparison of tuned SVMs and the original classifier panel.

on the library F experiments and tested on library S experiments, keeping the target constant between runs. The hyperparameters of the SVMs were set to the optimal values determined by a gridsearch (Section 5.2.4). The analysis was described further in Section 4.7. The classification accuracy of each trained SVM on the testing datasets (i.e. library S data) is shown in Figure 5.13. This figure shows that the accuracy of all trained SVMs was better than random assignment, which would achieve 50% accuracy.

5.3.1 RBF-SVM Versus SSK-SVM Accuracy on Library S

The difference in mean classification accuracy of the RBF- and SSK-SVM on the testing data is shown in Figure 5.13. Across all targets, the median difference in classification accuracy of the SVMs was not statistically significant because the 95% confidence interval spans 0 (M=1.0%, CI=[-14.8%, 11.0%]); however, for certain targets, the performance diverged considerably. For example, with targets Axl and Notch2, the performance of the RBF-SVM exceeded that of the SSK-SVM by 16.4% and 14.8%, respectively. Conversely, for targets Jagged2 and Notch3, the performance of the SSK-SVM exceeded that of the RBF-SVM by 17.3% and 10.1%, respectively. This shows that, a priori one does not know whether one SVM will perform better than the other, but for a given target it may be the case that one SVM will perform significantly better than the other.

5.3.2 Conclusions

Both the tuned RBF- and SSK-SVM were trained on library F data and tested on library S data. Both models achieved testing accuracy significantly better than achieved by chance, showing that the RBF- and

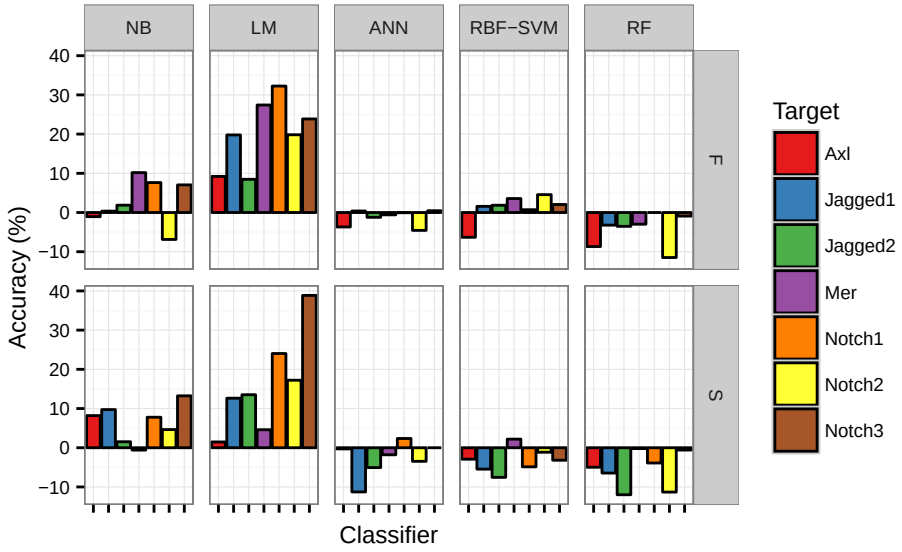


Figure 5.11: Improvement in accuracy after switching from a classifier in the original panel to the RBF-SVM with optimal hyperparameters. Improvement is measured as the increase in classification accuracy.

SSK-SVM are capable of generalizing to other libraries and could potentially be used to help non-experts in selecting clones for further study.

5.4 Interpreting the SVM Parameters

After training the RBF-SVM and the SSK-SVM models on the library F dataset, the model parameters were inspected in order to better understand the basis for their predictions. As explained in Section 2.4.5, the goal of SVM training is to select a subset of the training data, called support vectors, and an equal number of weights such that these weights and support vectors define the maximum-margin separating plane.

5.4.1 SSK-SVM

As explained in Section 4.8.2, the support vectors of the SSK-SVM model can be decomposed into their underlying 400-component vectors (one per dipeptide). Each of these 400 components multiplied by the support vector weight contributes additively to the decision function when the input contains the corresponding dipeptide as a subsequence. Thus, the 400-component support vectors multiplied by their weights can be summed (component-wise) to get an overall term for each dipeptide that determines the impact of a dipeptide on the prediction outcome. The dipeptides that have contributions greater than 1.0 or less than -1.0 are shown in Figure 5.17. All of the dipeptide contributions can be found in Table A.6.

For the trained SSK-SVM model, Figure 5.17 shows which dipeptides in a clonal sequence that have a significant contribution to the class prediction and the sign (+/-) of that contribution. The dipeptides

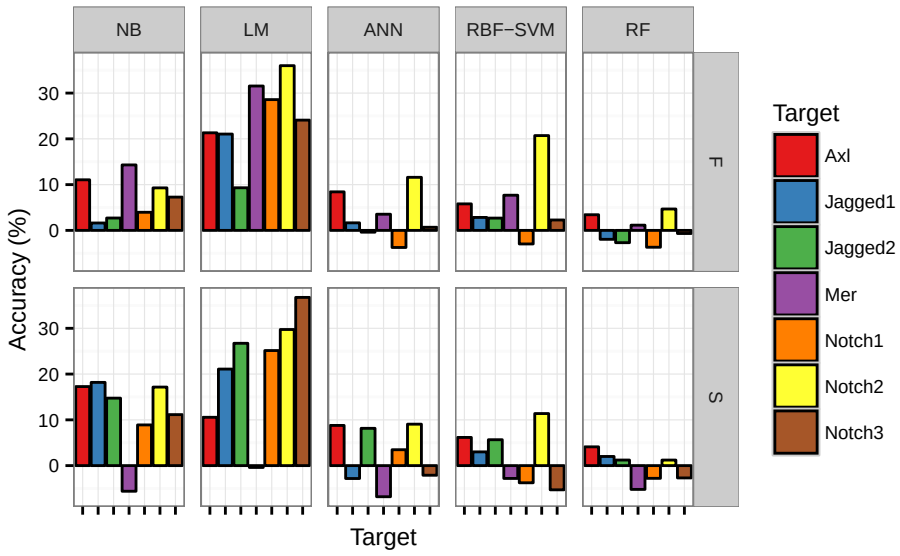


Figure 5.12: Improvement in accuracy after switching from a classifier in the original panel to the SSK-SVM with optimal hyperparameters. Improvement was measured as the increase in classification accuracy.

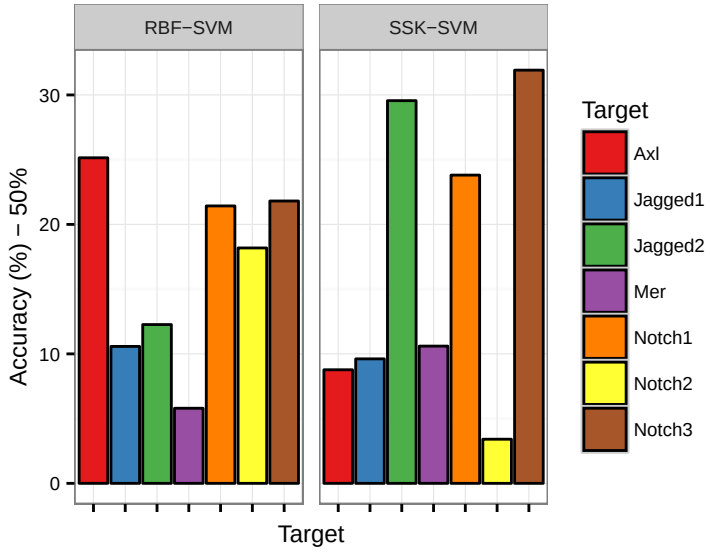


Figure 5.13: Accuracy of the trained RBF- and SSK-SVMs on the 7 library S datasets. The SVMs in this analysis were trained only on the library F data.

observed in Figure 5.17 can be classified into 3 categories: (1) dipeptides with mostly negative contribution across all targets, (2) dipeptides with mostly positive contribution across all targets, and (3) dipeptides whose contribution is target-dependent (e.g. HG, AG, and VG). Unsurprisingly, the large majority of dipeptides fall into the third category.

Table 5.5 shows the dipeptides with the highest and lowest weights for the SVM-SSK trained on each target dataset.

Both the Wilcoxon rank-sum test and t-test were performed on each of the seven contributions of each dipeptide. Using a significance level of 0.05, and the Bonferroni correction, both tests rejected the claim that the median contribution of any of the dipeptides across all targets (including those outside of the 7 studied here) is greater than or less than 0.

5.4.2 RBF-SVM

The input to the RBF-SVM can be thought of a 20-component vector that contains the values of each of the 20 features selected in Section 4.4.3. To understand how the RBF-SVM classifies the input, only a pair of features was considered at a time. For each pair of features x and y, the x and y range covered by all the support vectors was divided into 10×10 grid. Each vertex in the grid was then input to the RBF-SVM. The corresponding output of the RBF-SVM are shown in the tileplots in Figures 5.14 to 5.16 and Figures A.6 to A.8. The tileplots resulting from this procedure can be viewed as 2-dimensional projections of the decision function. A red tile indicates that the corresponding vector parameters are more likely to be associated with the PERSISTENT label, according to the SSK-SVM. A blue tile denotes an association with the TRANSIENT label.

5.4.3 Conclusions

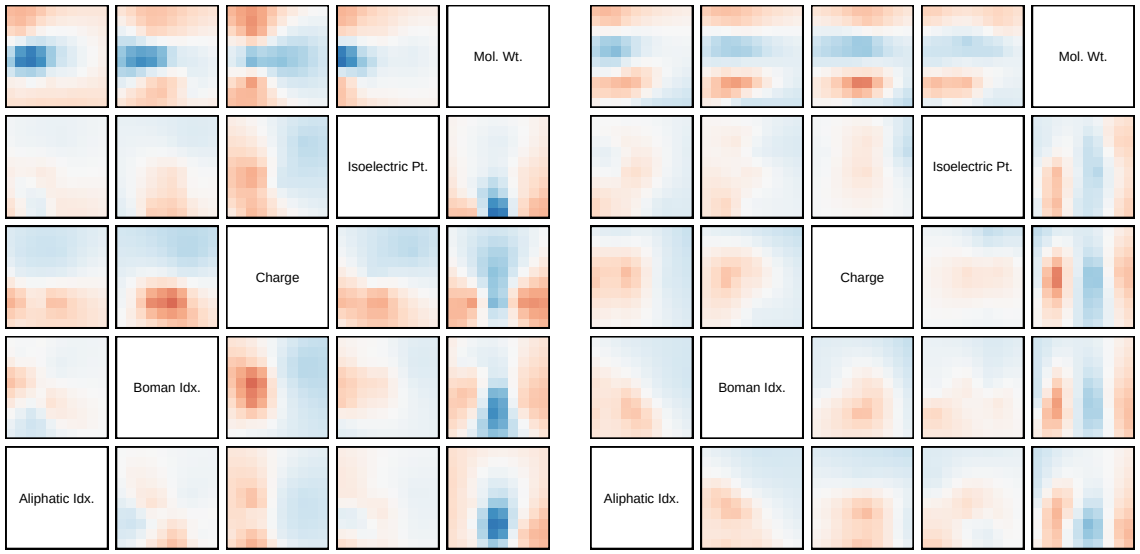
The tuned SVMs models that were trained on library F data were deconstructed to yield greater insight into the basis for their predictions.

Figure 5.14 shows that the predictions of the RBF-SVMs were highly dependent on the molecular weight of the CDRH3 sequences. Molecular weight is highly correlated with CDRH3 length; therefore, this observation is consistent with a study by Bharathi noting CDRH3 length preferences of the seven targets studied in this thesis [13].

Strong dependencies on the amino acid concentrations and dipeptide counts were also observed. For example, in Figure 5.15b, the bottom left corner of the C-D facet is red, suggesting that CDRH3 sequences with low C and low D amino acid concentrations were preferentially selected during panning. Albeit less pronounced, the pattern in Figure 5.16a shows that predictions were sensitive to the LG dipeptide count, with low counts generally preferred for prediction of persistent clones.

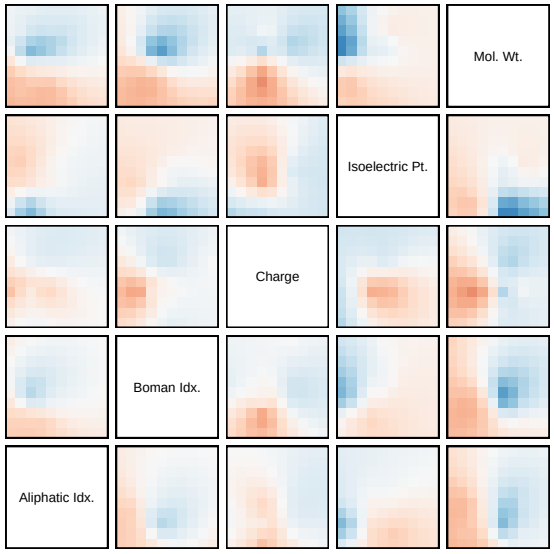
Figures 5.14 to 5.16 suggest that considering at least two variables simultaneously can be highly beneficial for predicting clone persistence. For example, in Figure 5.14a, in the facet showing molecular weight and charge, the tiles in the bottom left and bottom right corners are red, showing that a CDRH3 with a low or high molecular weight (using a relative scale) is preferred, but only when the charge of that CDRH3 is low.

Figure 5.17 shows that some peptides (e.g. DY, GD, and ID) have mostly a positive contribution across the 7 targets. Likewise, some peptides (e.g. LG, LW, and YT) have mostly a negative contribution across

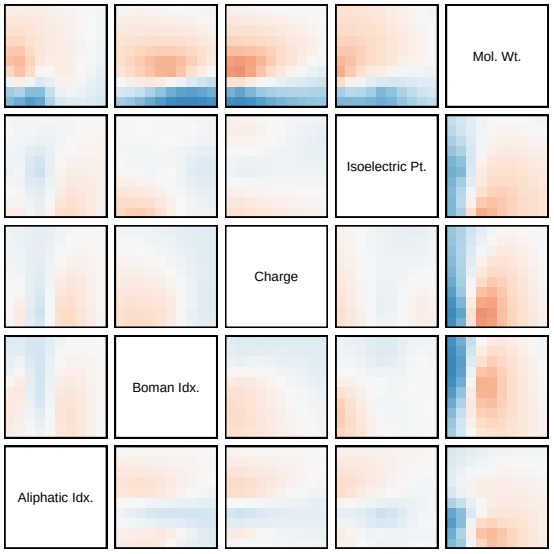


(a) Axl

(b) Jagged2

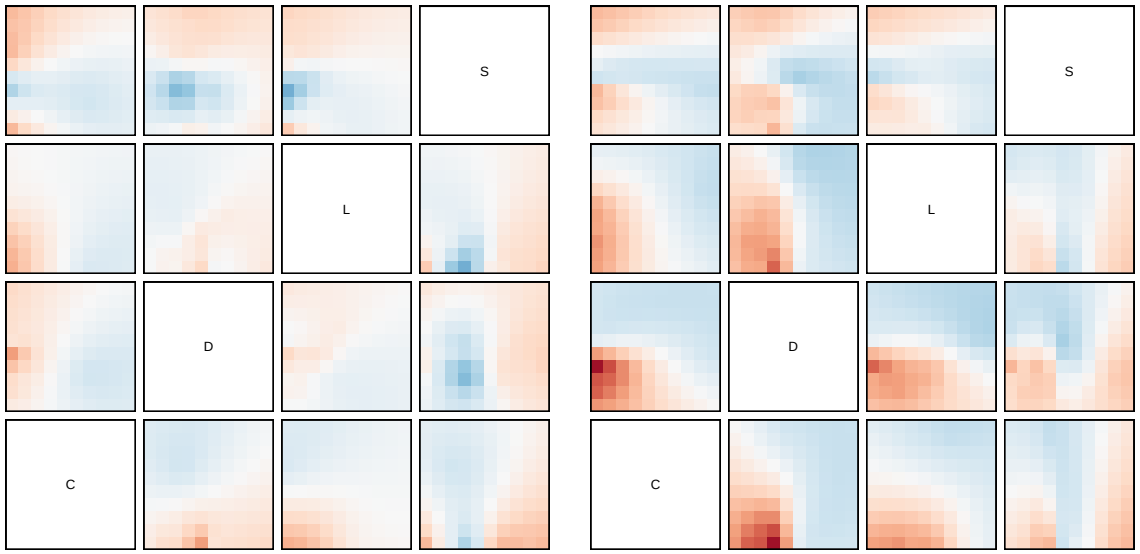


(c) Mer



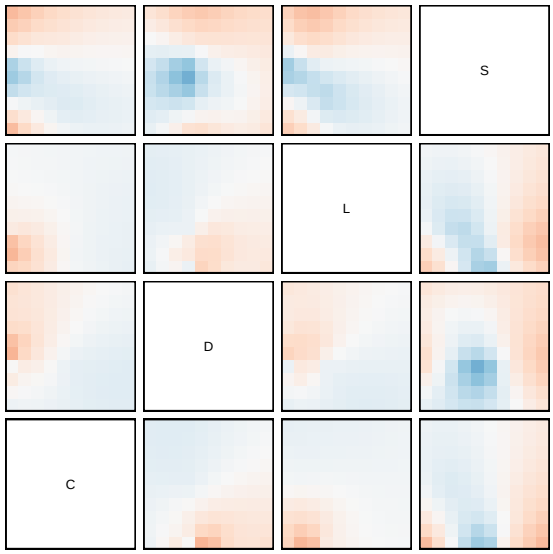
(d) Notch3

Figure 5.14: RBF-SVM support vector projections for the physicochemical properties. Panels (a), (b), (c), and (d) show the projections for the RBF-SVMs trained on datasets F-Axl, F-Jagged2, F-Mer, and F-Notch3, respectively. In each facet, the dimension of the vertical axis is equal to the name of the feature in that row of facets. Likewise, the dimension of the horizontal axis is equal to the name of the feature in that column of facets. The range of each axis is from the minimum support vector to the maximum support vector. Values have been centered; therefore, colour is only an indication of relative magnitude.

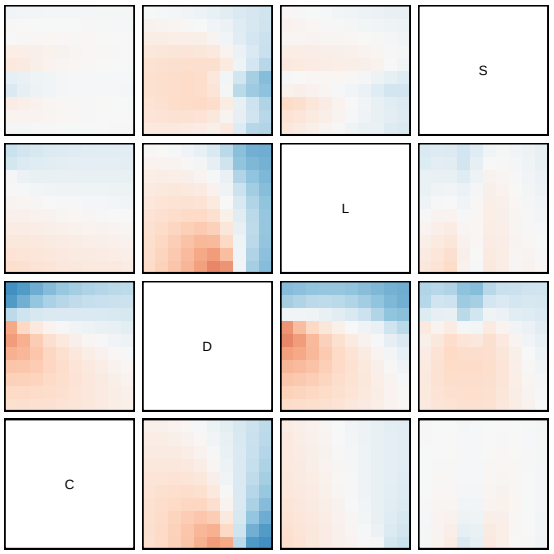


(a) Axl

(b) Jagged2

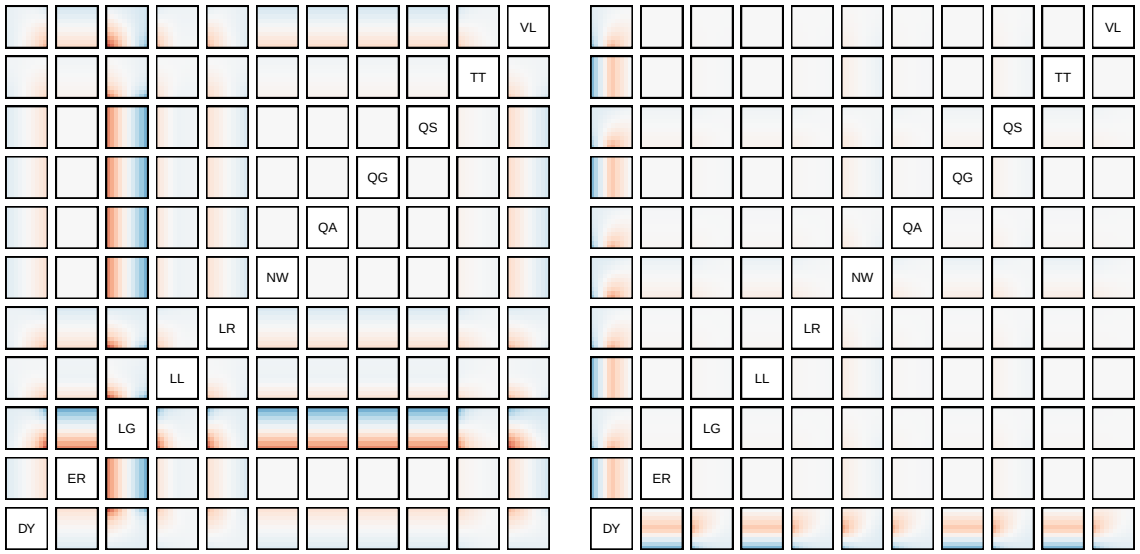


(c) Mer



(d) Notch3

Figure 5.15: RBF-SVM support vector projections for the amino acid compositions. Refer to Figure 5.14 for a description.



(a) Axl **(b)** Jagged2

(c) Mer **(d)** Notch3

Figure 5.16: RBF-SVM support vector projections for the dipeptide counts. Refer to Figure 5.14 for a description.

Table 5.5: Dipeptides with the biggest contribution to each SSK-SVM model. The dataset used to train the SSK-SVM model is shown in the top row.

Axl		Jagged1		Jagged2		Mer	
Dipep	Weight	Dipep	Weight	Dipep	Weight	Dipep	Weight
SP	−1.5505	GH	−1.3492	AP	−1.5584	PS	−1.1571
AY	−1.4057	SM	−1.1722	GH	−1.5152	YM	−0.9283
HG	−1.3518	YG	−1.1051	DD	−1.4967	PG	−0.7832
AG	−1.3415	FW	−1.0176	SH	−1.2213	SV	−0.7051
YM	−1.2585	YT	−1.0072	FV	−1.2193	YY	−0.6474
GI	1.0953	AG	1.1856	ID	1.3375	GM	0.5203
PA	1.1276	HA	1.4545	WY	1.4020	PA	0.5426
PH	1.1357	VG	1.6149	HY	1.4633	GD	0.5853
WA	1.2419	FY	1.6241	VA	1.4663	WA	0.6054
PY	1.3257	AW	2.3992	YM	2.1515	WV	0.6275

Notch1		Notch2		Notch3	
Dipep	Weight	Dipep	Weight	Dipep	Weight
FS	−0.7649	VY	−0.7848	GA	−1.4380
SW	−0.5493	SF	−0.7296	WL	−0.6782
GA	−0.5385	GY	−0.5237	HH	−0.6460
FD	−0.5130	YD	−0.5213	GL	−0.6009
GF	−0.4720	FD	−0.5094	LF	−0.5194
GD	0.9241	SV	0.7618	PS	1.3934
GG	0.9687	HW	0.7628	FA	1.4871
MD	1.0184	MD	0.8076	GS	1.5433
YG	1.0518	SA	0.8677	VS	1.6188
SG	1.1320	HG	1.4999	AS	2.3154

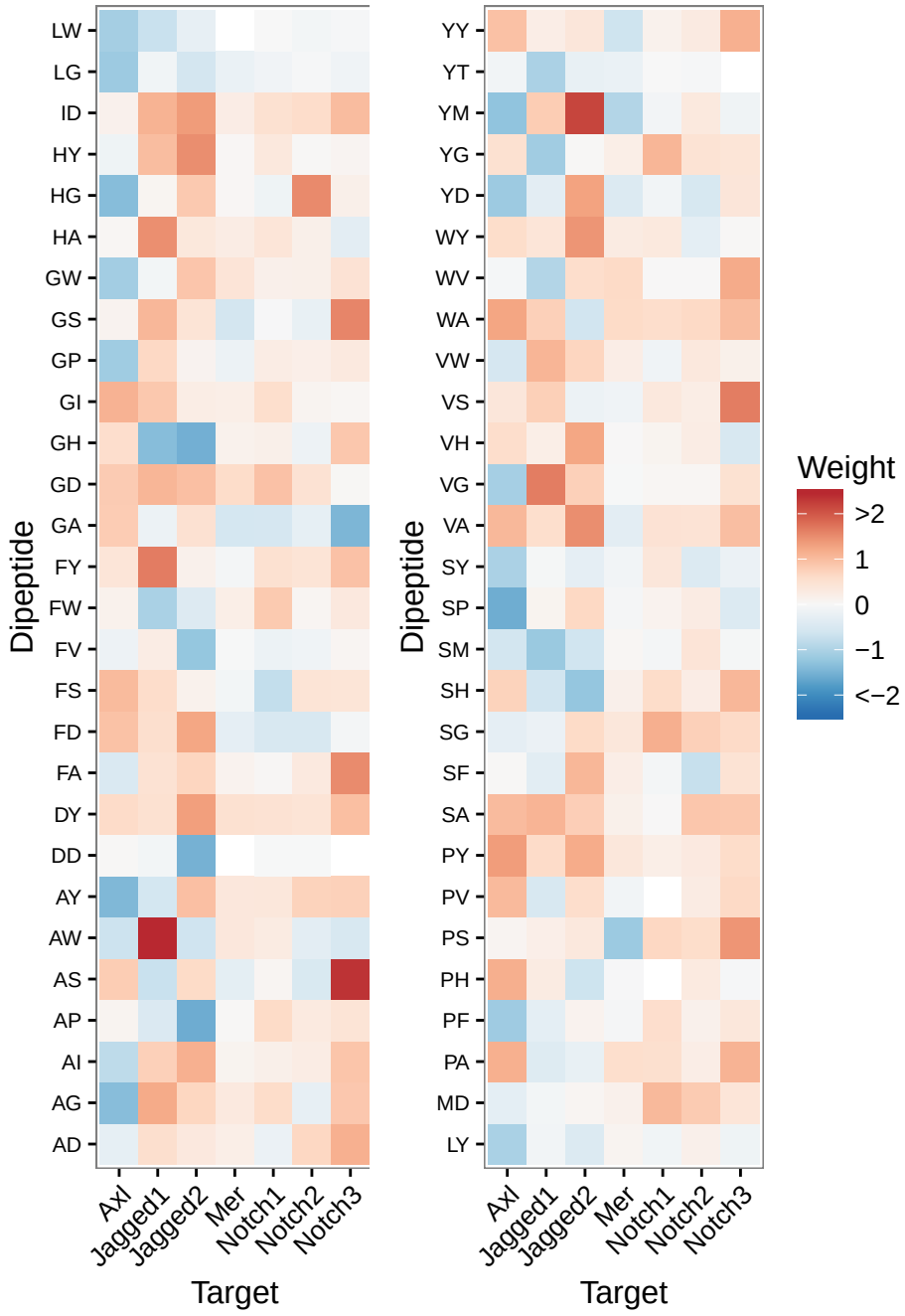


Figure 5.17: Dipeptide contributions for each of the 7 SSK-SVM models.

the 7 targets. Additionally, Table 5.5 shows that each target has a unique set of dipeptides that contribute positively and negatively to the predicted persistence of a clone. The dipeptides that contributed positively for target Mer had contributions of less magnitude than the other targets.

Chapter 6

Discussion & Future Work

6.1 Predicting Persistent Clones

This thesis focused on the task of predicting whether a naive clone from either library F or library S will still be observed after 5 rounds of panning against one of the seven targets described in Section 2.2.4. The labels PERSISTENT and TRANSIENT were used to distinguish between these two possibilities.

6.1.1 Machine Learning Comparison

Initially, five different classifiers were trained to predict a clone as either PERSISTENT or TRANSIENT based on its CDRH3 sequence. These classifiers were a random forest (RF), artificial neural network (ANN), support vector machine (SVM), logistic model (LM), and a naive Bayes network (NB). These five classifiers were cross-validated using the steps set out in Section 4.4 and then compared on the basis of their classification accuracy in Section 5.2.2. A multifaceted barplot of the classification accuracy with facets arranged by classifier is shown in Figure 5.4 and by dataset in Figure 5.5.

Although ANN, RF, and SVMs all achieved similar accuracies, each classifier entailed different advantages/disadvantages. For example, one advantage of the random forest was that little effort was required to tune its hyperparameters because the random forest only had one hyperparameter: the number of decision trees. Preliminary trials showed quite readily that using more than the default 100 decision trees did not produce a noticeable improvement in accuracy. On the other hand, it was difficult to interpret the random forest because it was made up of so many decision trees, each trained using a slightly different subset of the data; therefore, each decision tree was different. Contrasting with the random forest, the artificial neural network had many hyperparameters, including the number of hidden layers, the number of nodes per layer, the activation function of each layer, the initial weights prior to training. The abundance of hyperparameters could be viewed as an advantage or a disadvantage. It could be an advantage because the ability to customize ANNs allows them to be applied to a variety of problems. It could also be a disadvantage because, *a priori*, finding the structure that works best for a given problem has the potential to be a time-consuming process of trial-and-error. One advantage that the SVMs had over the RF and ANN models was that, whereas the outcome of ANN training depends on the initial parameter values and the outcome of RF training is random by design, the outcome of SVM training is deterministic. That is, there is a single globally optimal solution

which the training procedure is guaranteed to converge to. A disadvantage of the SVM is that there are a limited number of practical kernel types, which imposes certain restrictions on the flexibility of SVMs.

From Figure 5.5, it is apparent that the average performance on the S-Mer dataset was significantly lower than on other datasets. Perhaps the poor performance is related to the exceptional nature of the S-Mer dataset noted in Section 5.1.1, Section 5.1.2, and Section 5.4.3 and the independent observation that S-Mer failed to produce target-specific clones [13]. The lack of well-defined clusters in Section 5.1.1 suggests that, contrary to other experiments, there were no learnable patterns in the S-Mer-5 CDRH3 sequences to begin with.

6.1.2 Improving SVM Performance

The best hyperparameters for the SSK-SVM were determined by gridsearch to be $n = 2$ and $\lambda = 0.6$. Intuitively, n and λ are the length of the substrings and the gap penalty, and are described further in Section 2.4.5. The hyperparameter $n = 2$ meant that considering amino acid pairs gave the best performance. The $\lambda = 0.6$ meant that for each additional gap between the two amino acids, the contribution was diminished almost by half. It is interesting to note that when gapped matches were discounted ($\lambda = 0$), classification accuracy decreased, meaning that accounting for gapped matches actually improved classification accuracy. This finding is significant because it supports the use of the SSK over simpler methods that only consider continuous stretches of amino acids, called k-mers [41].

6.2 Generalizing the Trained SVM Models

RBF- and SSK-models were trained on library F data and tested on library S data to determine the extent to which the models can generalize to other libraries.

6.2.1 SVM Performance on Library S

Figure 5.13 shows accuracy of the RBF- and SSK-SVMs compared to the accuracy that would be achieved by a random assignment of the PERSISTENT and TRANSIENT labels. The figure shows that both SVMs surpassed the accuracy that would be achieved by relying on chance by a significant margin. This result was encouraging because, although library F and library S have similar CDRH3 specifications (see Table 2.1), a significant portion of CDRH3 sequences in library S are exclusive to library S. More work needs to be done to determine to what degree the machine learning models generalize to this exclusive portion.

Although only library S was used for testing, this result suggests that the SVMs trained on library F generalize to other experiments using the same target. Assuming this to be true, these SVMs could be used to select clones in experiments that use the same targets provided that the library specification and antibody framework are similar. More testing is needed to verify this hypothesis and future work could focus on cross-validating the accuracy of the trained SVMs on libraries other than F and S.

Because library F experiments were more successful at isolating target-binding clones than library S experiments, it is possible that the SVMs trained on library F could identify target-binding clones from library S experiments that yielded no obvious candidates, and thus were deemed to have failed.

6.3 SVM Models to Library Recommendations

6.3.1 Features of the CDRH3 Sequences

The SSK-SVM function projections in Figure 5.14 show that there is a steep colour transition on most facets involving molecular weight. The steep colour transition indicates that the classification of the SSK-SVM is highly dependent on the molecular weight of the input vector. Stated another way, all else being equal, a vector that falls in the blue region is more likely to be classified as PERSISTENT than a vector that falls in the red region. The high dependence on CDRH3 molecular weight, which is highly correlated with CDRH3 length, is consistent with the effect that CDRH3 length was seen to have on the fate of clones [13].

If one were to use the tileplots in Figures 5.14 to 5.16 to design better CDRH3 sequences, one could change the amino acids in the CDRH3 sequence such that the sequence properties fall further into the red regions of the tileplots than the blue regions. For example, in Figure 5.14a, in the molecular weight versus charge facet, the bottom left and bottom right corners are bright red while the rest is blue. To create a focused antibody library for Axl, one might start with library F and modify it such that more clones have low or high molecular weights, and low charge.

6.3.2 Dipeptide Compositions of the CDRH3 Sequences

The dipeptide contributions suggest a way to construct focused antibody libraries. For example, to focus library F towards all seven targets, one might rework the library specification to increase the occurrence of dipeptides that have mostly positive contributions towards the 7 targets and decrease the occurence of dipeptides with mostly negative contributions towards the 7 targets. Further, to focus library F towards just one of the seven targets, Axl for example, one might additionally increase and decrease the occurence of dipeptides that contribute positively and negatively to Axl, respectively.

The reader may wonder whether any of the dipeptides with contributions mostly positive or mostly negative might have similar effects across targets outside of the 7 studied in this thesis. Assuming the 7 targets studied in this thesis were a random sampling of all possible antibody phage display targets, this hypothesis (i.e. that the median contribution of a dipeptide is greater than or less than 0) can be tested with the Wilcoxon ranked-sum test. Unfortunately, the result in Section 5.4.1, which used the Bonferroni correction, rejected this hypothesis.

Admittedly, the Bonferroni correction is one of the most conservative methods for controlling error in a multiple hypothesis test. Future work will try less sensitive methods, including the Šidák procedure, Holm procedure, and procedures that achieve greater sensitivity by accepting a proportion of false discoveries [42].

6.4 Relation to Other Work

The work of this thesis explored a niche application of machine learning that has received little to no attention in the scientific community. Although attempts to consolidate and make sense of phage display data is not a new concept (bioinformatics applications to phage display has been reviewed by Huang et al. [43]), the vast majority of work applies to general peptide phage display, as opposed to antibody phage display (i.e. display of antibody fragments).

MimoDB 2.0 [44], a database containing peptides with known affinities for various targets, was considered as an additional data source for the machine learning pipeline of this thesis; however, very few antibodies were stored in this database; and of those antibodies, the listed target was different from the 7 targets of this thesis. Future work could focus on creating a database similar in principal to MimoDB, but focused toward antibody fragments. If such a database were developed, the information could be used to train and validate future iterations of the machine learning pipeline presented of this thesis.

Another bioinformatics tool applicable to phage display is called SAROTUP [45]. The goal of SAROTUP is to identify peptides in phage display experiments that contain motifs known to bind contaminants, reagents, and substrates commonly encountered in phage display. These peptides are called target-unrelated peptides because they can become enriched during panning and can therefore be mistaken for true target-binding peptides. SAROTUP was considered for use in the pipeline presented in this thesis, but SAROTUP was designed with general peptide phage display in mind as opposed to antibody phage display, and given the highly structured nature of the antibody variable region (i.e. 6 loops from two different chains form the final binding surface), it was decided that using sequence patterns with no consideration of antibody-specific structure was not conducive to identifying target-unrelated antibodies.

A search for existing applications of machine learning to antibody phage display turned up only one result, a technology called Abtracer. Abtracer is a data-driven technology owned by the company Molcure. Unfortunately, the Molcure website provides no citations to papers describing the construction or validation of the Abtracer pipeline [46]—possibly due to the proprietary nature of the technology. Molcure does, however, provide a sketch of the pipeline, which includes the extraction of over 400 features (only molecular weight, isoelectric point, and amino acid composition are explicitly listed), and developing classifiers via SVMs, random forests, and deep neural networks (i.e. a many-layered artificial neural network).

6.5 Training Data Quantity and Quality

The amount of training data used for training the machine learning models is shown in Table 5.3. The size of the training dataset was deemed suitable for machine learning methods with a low to moderate number of parameters, like the models used in this thesis. Using complex models with many parameters would have lead to overfitting during training.

Some of the features that were extracted from the CDRH3 sequences were not independent of other features (e.g. molecular weight and length were correlated). The CFS routine, which was used for feature selection, attempted to exclude correlated features so that the selected subset of features were nearly independent.

6.6 Contributions

There are several potential uses for the classifiers that were developed in this thesis. The first potential use is in selecting clones for further testing when the most abundant clones were found to be unsuitable (e.g. the clones were target-unrelated, or they lost their binding ability when converted to full antibody molecules). Whereas clone abundance is the first criteria used by analysts for picking clones for further study, studies have shown that in some situations, clone abundance correlated poorly with target affinity [47]. The classifiers developed in this thesis consider only the properties of the clonal sequence; therefore, they can be used when clone abundance does not appear to be a suitable criteria.

6.7 Future Work

The work presented here is a first step towards using artificial intelligence to understand the seemingly chaotic patterns of clone enrichment in antibody phage display. There are a number of ways this work could be extended.

6.7.1 Feature Selection

The set of features extracted from the CDRH3 sequence of a clone was not exhaustive; therefore, future work could consider features outside of those used in this thesis, like secondary structure or the Kidera factors [48], which were derived using principal component analysis and are therefore independent. Another possible feature to include is the abundance of the clone in the naive library. Although including abundance as a feature would make the prediction more library-dependent, it might improve the prediction accuracy substantially. Further, methods beside CFS could be explored for selecting a subset of features to include in the final machine learning dataset.

6.7.2 Alternative Classification Models

Figure 5.4 and the statistical tests in Table 5.4 show that, with respect to classification accuracy, the difference between the ANN, RBF-SVM, and RF was insignificant. Moreover, this result held after tuning the ANN (with respect to the number of hidden layers) and the RBF-SVM, and adding the SSK-SVM to the classifier panel. The similar accuracies of these classifiers suggest that improving classification accuracy beyond that achieved in this thesis may require a drastically different approach. One such approach might be to consider the structural characteristics of all six CDRs and the structural characteristics of the target. Since the six CDRs interact with each other to form the surface that may or may not bind the target, these structures are of paramount importance to the problem of predicting clone persistence. Unfortunately, the structure of CDRs and the target will not be available in many cases, and to try to predict these structures from their sequence encroaches on a very difficult problem in computational biology. An acceptable compromise may be to predict coarse structural features of the CDRs and target (e.g. alpha helices, beta strands, or buried residues) and use those features in a machine learning pipeline instead/in addition to the features used in this thesis.

Another approach that could be the focus of future work is to use machine learning models with dedicated inputs for each position of the CDRH3 sequence. One example of such a model is an ANN that has a set of input nodes for each position of a CDRH3 sequence and instead of a node being set to a property of the entire sequence, a node is set to a property of the amino acid at a particular position in the sequence. Such an ANN could then learn position-dependent patterns in the CDRH3 sequences and maybe achieve better classification accuracy than achieved in this thesis. In pursuing this future work, one would need to consider the increased computational demand and potential for over-fitting that characterize models with many parameters.

Another method for achieving increased classification accuracy are ensemble methods, methods that combine multiple models to improve prediction performance. One simple ensemble method is called "majority voting". Majority voting combines the predictions of several models by selecting the prediction that is output from a majority of the models. To implement a majority voting scheme with the classifiers developed in this thesis, the clonal sequence would be input to each of the models in the ensemble and the prediction would be the class that was predicted by a majority of the models [49].

6.7.3 Predicting Clone Abundance

The aim of this thesis was to predict the presence of absence of a clone after the fifth round of panning based on the clonal sequence. A slightly different prediction task would be to predict the abundance of the clone after the fifth round of panning.

6.7.4 Target-Independent Method

Two variations on the methodology of this thesis could solve related problems of interest to antibody phage display researchers. The first variation would involve combining the datasets from all the experiments and training the machine learning models on the combined dataset. By training the models on all of the data, the hypothesis would be that there are some general properties of sequences that relate to clone persistence, independent of the target.

The second variation would use a dataset from an antibody phage display experiment that used no target. By training models to predict clones that are observed after five rounds of panning against no target, the models might learn properties of CDRH3 sequences that are characteristic of target-unrelated clones. With such a model, target-unrelated clones could be filtered from the sequence outputs of future experiments.

6.7.5 Size of the Training Dataset

It would be interesting to also look at the correlation between the size of the training datasets and the prediction performance of the models. In the case that there is a correlation between dataset size and prediction performance, one could improve this work by increasing the size of the datasets used to train the machine learning models. The number of sequences (i.e. datapoints) collected from an antibody phage display experiment is limited by the sequencing technology and protocol used for sequencing. More sequences could be obtained by increasing the sequencing depth of a single sample. A more labour-intensive approach to increasing the number of sequences would be to perform replicates of the same experiment.

References

[1] Alberts B, Johnson A, Lewis J, Raff M, et al. *Molecular Biology of the Cell.* Garland Science, New York, 4 ed., 2002.

[2] Lipman NS, Jackson LR, Trudel LJ, and Weis-Garcia F. Monoclonal versus polyclonal antibodies: Distinguishing characteristics, applications, and information resources. *ILAR J.*, 46(3):258–268, 2005.

[3] Liu JKH. The history of monoclonal antibody development – progress, remaining challenges and future innovations. *Ann. Med. Surg.*, 3(4):113–116, 2014.

[4] Maggon K. Monoclonal antibody "gold rush". *Curr. Med. Chem.*, 14(18):1978–87, 2007.

[5] Hu D, Hu S, Wan W, Xu M, et al. Effective optimization of antibody affinity by phage display integrated with high-throughput dna synthesis and sequencing technologies. *PLoS One*, 10(6):e0129125, 2015.

[6] Mathonet P and Ullman CG. The application of next generation sequencing to the understanding of antibody repertoires. *Front Immunol.*, 4(Article 265), 2013.

[7] Turner KB, Naciri J, Liu JL, Anderson GP, et al. Next-generation sequencing of a single domain antibody repertoire reveals quality of phage display selected candidates. *PLoS ONE*, 11(2):e0149393, 2016.

[8] Zhai W, Glanville J, Fuhrmann M, Mei L, et al. Synthetic antibodies designed on natural sequence landscapes. *J Mol Biol.*, 412(1):55–71, 2011.

[9] Sompayrac L. *How the Immune System Works.* Wiley, West Sussex, UK, 4th ed., 2012.

[10] Harris LJ, Larson SB, Hasel KW, and McPherson A. Refined structure of an intact IgG2a monoclonal antibody. *Biochemistry*, 36(7):1581–97, 1997.

[11] Berman HM, Westbrook J, Feng Z, Gilliland G, et al. The protein data bank. *Nucl. Acids. Res.*, 28(1):235–242, 2000.

[12] Miersch S and Sidhu SS. Synthetic antibodies: Concepts, potential and practical considerations. *Methods*, 2012. doi:10.1016/j.ymeth.2012.06.012.

[13] Bharathikumar VM. *Next-Generation Sequencing and Motif Grafting Applications in Antibody Discovery.* Ph.D. thesis, 2016 (expected).

[14] Carmen S and Jermutus L. Concepts in antibody phage display. *Briefings in Functional Genomics and Proteomics*, 1(2):189–203, 2002.

[15] Consortium TU. UniProt: a hub for protein information. *Nucl. Acids Res.*, 43(D1):D204–D212, 2015.

[16] Crooks GE, Hon G, Chandonia JM, and Brenner SE. Weblogo: A sequence logo generator. *Genome Res.*, 14:1188–90, 2004.

[17] Alpaydin A. *Introduction to Machine Learning.* The MIT Press, Cambridge, MA, 2nd ed., 2010.

[18] Hall MA. *Correlation-based Feature Selection for Machine Learning.* Ph.D. thesis, The University of Waikato, 1999.

[19] Witten IH and Frank E. *Data Mining.* Morgan Kaufmann, San Francisco, CA, 2 ed., 2005.

[20] Breiman L. Random forests. *Machine Learning*, 45:5–32, 2001.

[21] Kingsford C and Salzberg SL. What are decision trees? *Nat. Biotechnol.*, 26(9):1011–13, 2008.

[22] Ng A. CS229 lecture notes. [Online; accessed 2-July-2016].

[23] Lodhi H, Saunders C, Shawe-Taylor J, Cristianini N, et al. Text classification using string kernels. *J. Machine Learning Res.*, 2:419–444, 2002.

[24] Bishop CM. *Neural Networks for Pattern Recognition*. Oxford University Press, New York, 1995.

[25] Durbin R, Eddy SR, Krogh A, and Mitchison G. *Biological sequence analysis: Probabilistic models of proteins and nucleic acids*. Cambridge University Press, Cambridge, UK, 2002.

[26] Jones DT. Protein secondary structure prediction based on position-specific scoring matrices. *J. Mol. Biol.*, 292(2):195–202, 1999.

[27] You ZH, Chan KCC, and Hu P. Predicting protein-protein interactions from primary protein sequences using a novel multi-scale local feature representation scheme and the random forest. *PLoS ONE*, 10(5):e0125811, 2015. doi:10.1371/journal.pone.0125811.

[28] Ramsundar B, Kearnes S, Riley P, Webster D, et al. Massively multitask networks for drug discovery. 2015. `1502.02072v1`.

[29] Dalgaard P. *Introductory Statistics with R*. Statistics and Computing. Springer, New York, 2nd ed., 2008.

[30] Kim T, Tyndel MS, Huang H, Sidhu SS, et al. MUSI: an integrated system for identifying multiple specificity from very large peptide or nucleic acid data sets. *Nucl. Acids Res.*, 2011. doi:10.1093/nar/gkr1294.

[31] Nixon AE, Sexton DJ, and Ladner RC. Drugs derived from phage display. *mAbs*, 6(1):73–85, 2014.

[32] Rice P, Longden I, and Bleasby A. EMBOSS: The european molecular biology open software suite. *Trends in Genetics*, 16(6):276–277, 2000.

[33] PostgreSQL. `https://www.postgresql.org`. Open Source Software; Accessed 2016 July 01.

[34] Osorio D, Rondon-Villarreal P, and Torres R. *Peptides*. `https://cran.r-project.org/web/packages/Peptides/index.html`, 2015.

[35] Ikai A. Thermostability and aliphatic index of globular proteins. *J. Biochem.*, 88(6):1895–8, 1980.

[36] Guruprasad K, Reddy BVB, and Pandit MW. Correlation between stability of protein and its dipeptide composition: a novel approach for predicting in vivo stability of a protein from its primary sequence. *Protein Eng.*, 4(2):155–61, 1990.

[37] Hall M, Frank E, Holmes G, Pfahringer B, et al. The weka data mining software: An update. *SIGKDD Explorations*, 11(1), 2009.

[38] Cock PA, Antao T, Chang JT, Bradman BA, et al. Biopython: freely available python tools for computational molecular biology and bioinformatics. *Bioinformatics*, 25(11):1422–23, 2009.

[39] Wickham H. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2009. ISBN 978-0-387-98140-6.

[40] Derda R, Tang SKY, Li SC, Matochko W, et al. Diversity of phage-displayed libraries of peptides during panning and amplification. *Molecules*, 16(2):1776–1803, 2011.

[41] Leslie C, Eskin E, and Noble WS. The spectrum kernel: A string kernel for svm protein classification. Altman RB, ed., *Proc of Pac Symp Biocomput. 2002*, pages 564–575. Internat Soc for Comp Bio. (ISCB), World Scientific, Kauai, Hawaii, January 2002.

[42] Dudolt S, Shaffer JP, and Boldrick JC. Multiple hypothesis testing in microarray experiments. *Stat Sci.*, 18(1):71–103, 2003.

[43] Huang J, Ru B, and Dai P. Bioinformatics resources and tools for phage display. *Molecules*, 16(1):694–709, 2011. doi:10.3390/molecules16010694.

[44] Huang J, Ru B, Zhu P, Nie F, et al. MimoDB 2.0: a mimotope database and beyond. *Nucleic Acids Res.*, 40(Database issue):D271–D277, 2012. doi:10.1093/nar/gkr922.

[45] Huang J, Ru B, Li S, Lin H, et al. SAROTUP: Scanner and report of target-unrelated peptides. *J Biomed Biotechnol.*, 2010:Article ID 101932, 2010. doi:10.1155/2010/101932.

[46] Molcure Inc. Abcure. `http://molcure.com/abtracer`. Online; Accessed 2016 08 03.

[47] Glanville J, D'Angelo S, Khan TA, Reddy ST, et al. Deep sequencing in library selection projects: what insight does it bring? *Curr Opin Struc Biol.*, 33:146–160, 2015. doi:10.1016/j.sbi.2015.09.001.

[48] Kidera A, Konishi Y, Oka M, Ooi T, et al. Statistical analysis of the physical properties of the 20 naturally occurring amino acids. *J. Protein Chem.*, 4(1):23–55, 1985.

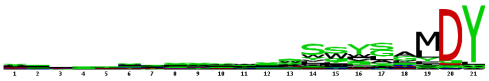
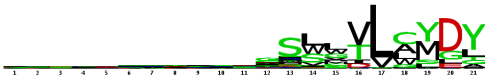
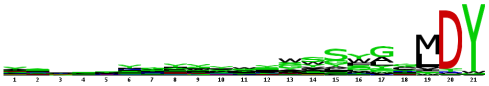
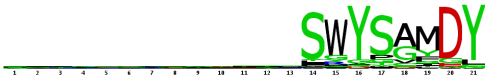
[49] Zhou ZH. *Ensemble Methods: Foundations and Algorithms.* CRC Press, Boca Raton, 2012.

Appendix A

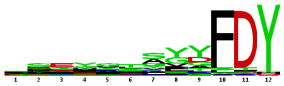
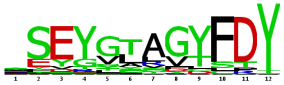
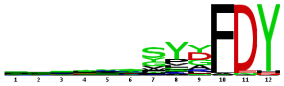
Supplementary Figures

A.1 MUSI Clusters

Table A.1: Sequence logos for clusters in samples F-Jagged1-5 and S-Jagged1-5. The first row in each table shows the sequence logo for the entire sample. The column labelled "size" shows the number of distinct clonal sequences falling into each cluster.

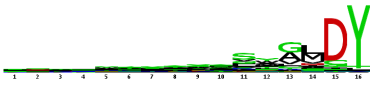
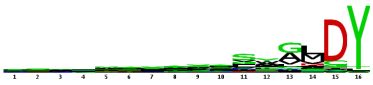
Size	Logo
All	
11	
77	
36	

(a) Library F

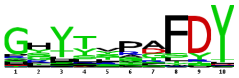
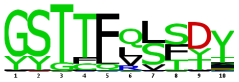
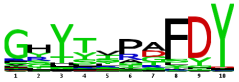
Size	Logo
All	
19	
32	

(b) Library S

Table A.2: Sequence logos for clusters in samples F-Jagged2-5 and S-Jagged2-5. The first row in each table shows the sequence logo for the entire sample. The column labelled "size" shows the number of distinct clonal sequences falling into each cluster.

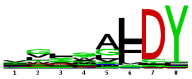
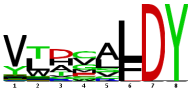
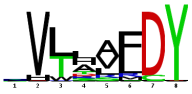
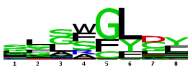
Size	Logo
All	
242	

(a) Library F

Size	Logo
159	
13	
8	
120	
9	
7	

(b) Library S

Table A.3: Sequence logos for clusters in samples F-Notch1-5 and S-Notch1-5. The first row in each table shows the sequence logo for the entire sample. The column labelled "size" shows the number of distinct clonal sequences falling into each cluster.

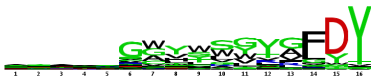
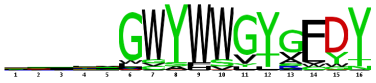
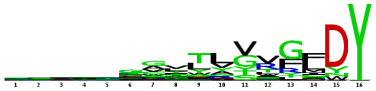
Size	Logo
66	
17	
9	
12	
17	
8	

(a) Library F

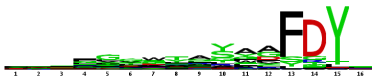
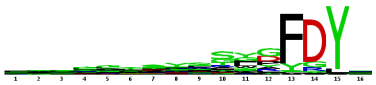
Size	Logo
42	
42	

(b) Library S

Table A.4: Sequence logos for clusters in samples F-Notch2-5 and S-Notch2-5. The first row in each table shows the sequence logo for the entire sample. The column labelled "size" shows the number of distinct clonal sequences falling into each cluster.

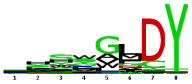
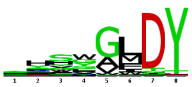
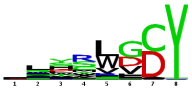
Size	Logo
66	
18	
8	
11	
16	
10	

(a) Library F

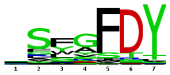
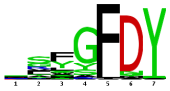
Size	Logo
88	
27	
60	

(b) Library S

Table A.5: Sequence logos for clusters in samples F-Notch3-5 and S-Notch3-5. The first row in each table shows the sequence logo for the entire sample. The column labelled "size" shows the number of distinct clonal sequences falling into each cluster.

Size	Logo
220	
165	
54	

(a) Library F

Size	Logo
94	
12	
32	
49	

(b) Library S

A.2 SVM Gridsearch

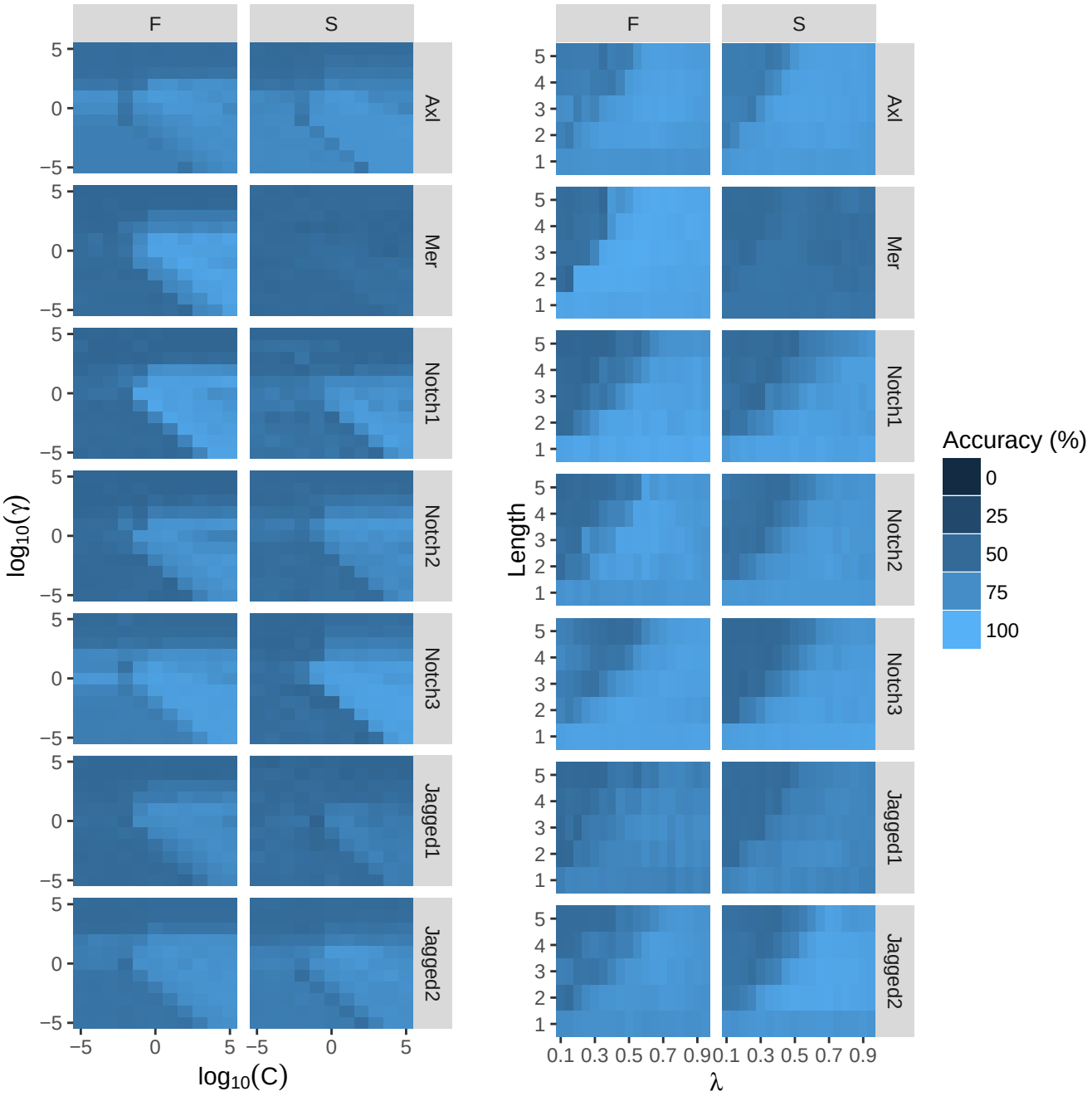


Figure A.1: Tileplots showing the cross-validated classification accuracy for (left) the RBF-SVM and (right) the SSK-SVM.

A.3 Physicochemical Property Distributions

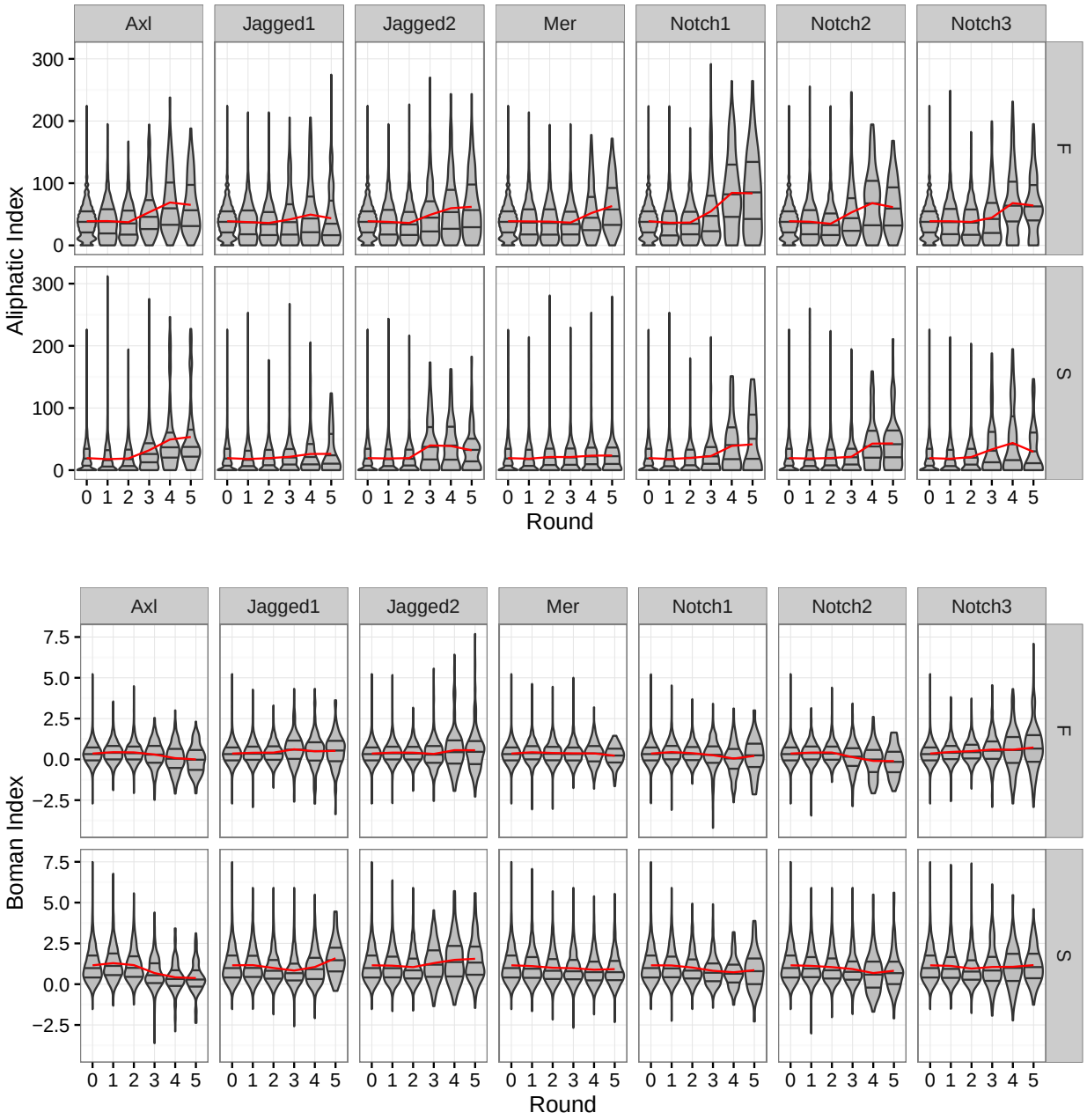


Figure A.2: Violin plots showing the distribution of the aliphatic and Boman indices across the set of distinct clones identified in each sample. Refer to Figure 5.3 for a description.

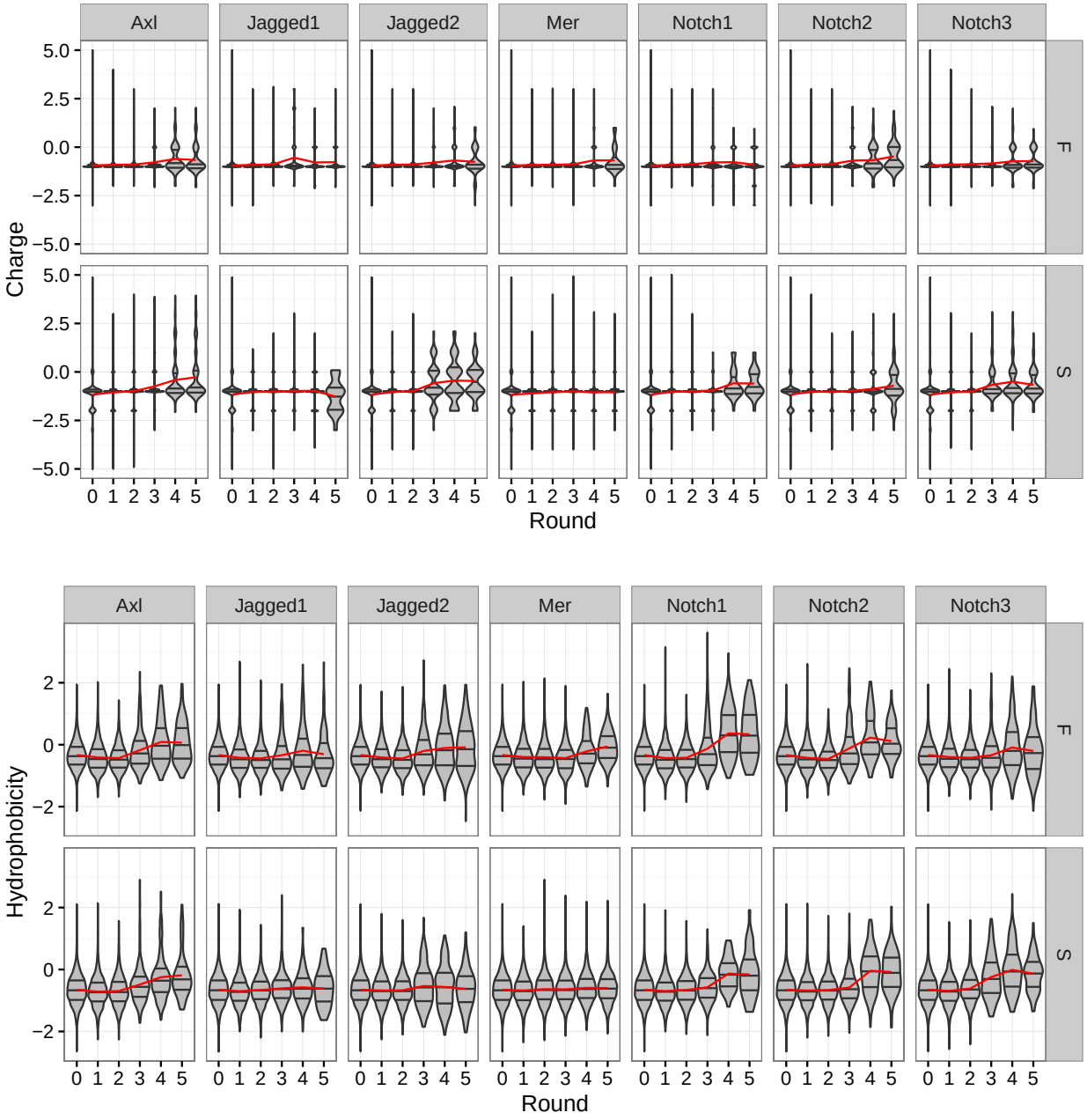


Figure A.3: The distribution of the charge and hydrophobicity across the set of distinct CDRH3 sequences identified in each sample. Refer to Figure 5.3 for a description.

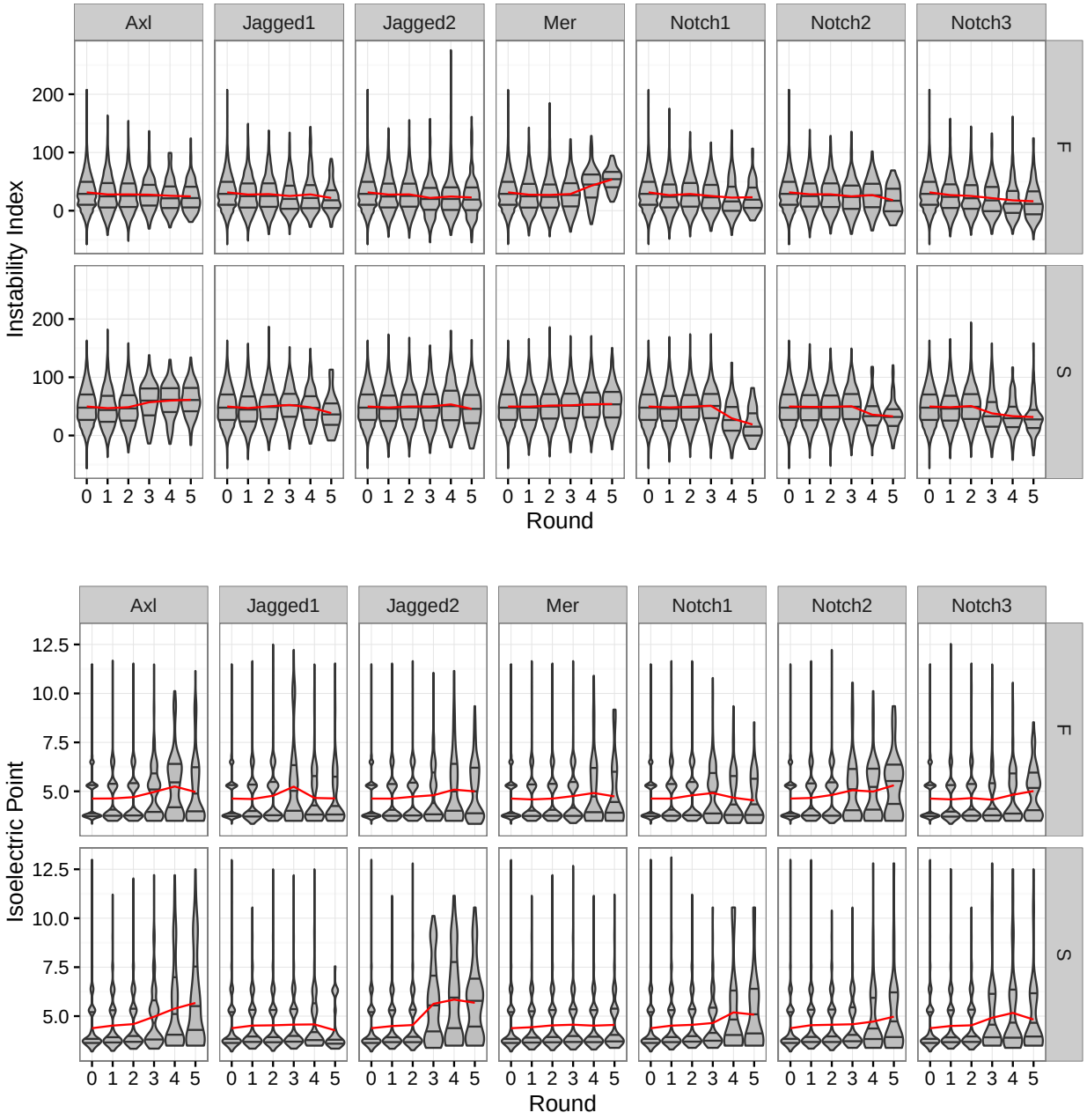


Figure A.4: The distribution of the instability index and isoelectric point across the set of distinct CDRH3 sequences identified in each sample. Refer to Figure 5.3 for a description.

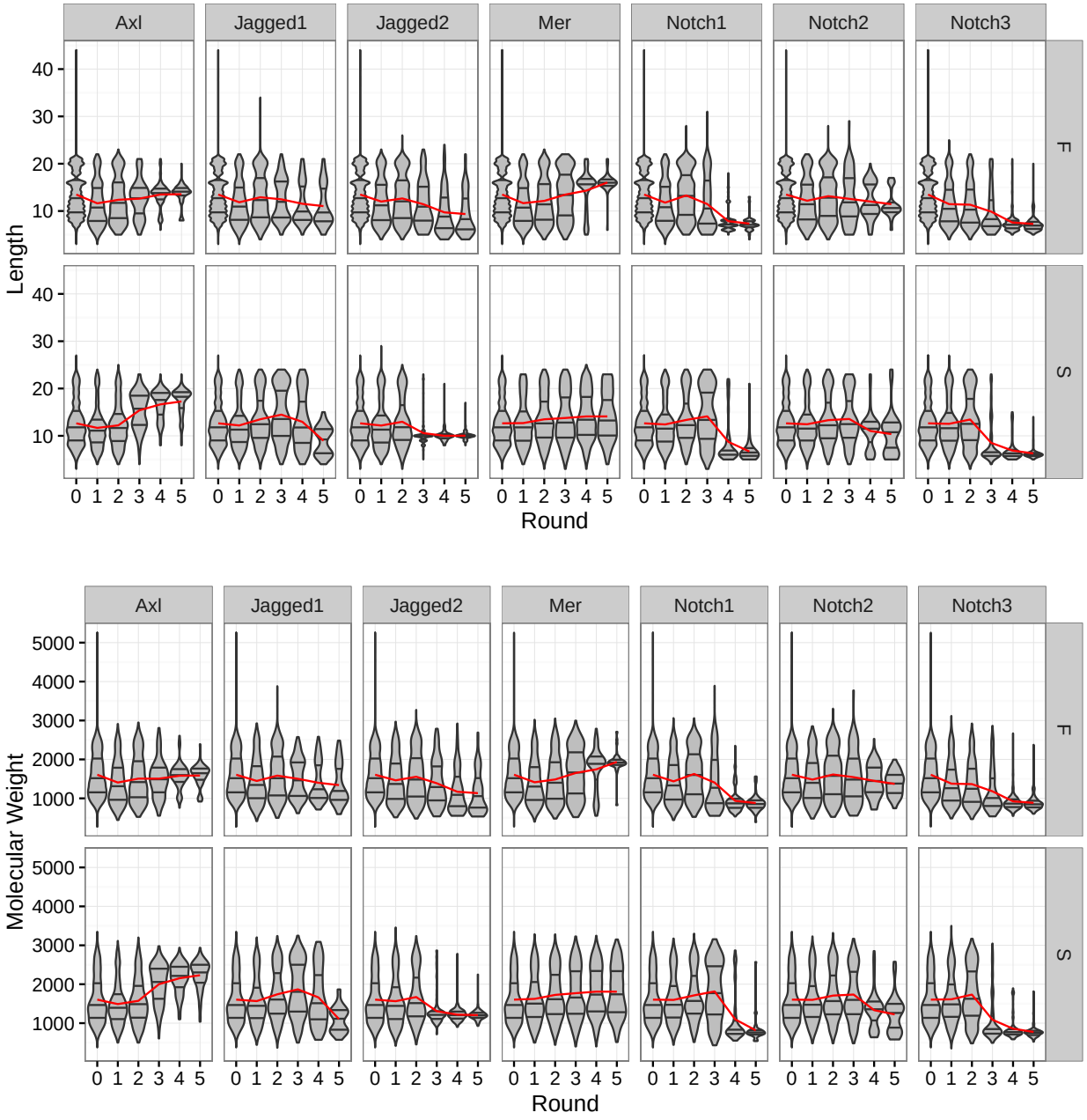
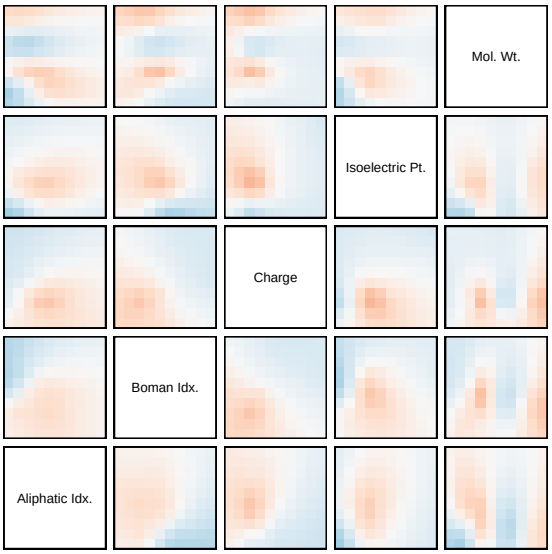
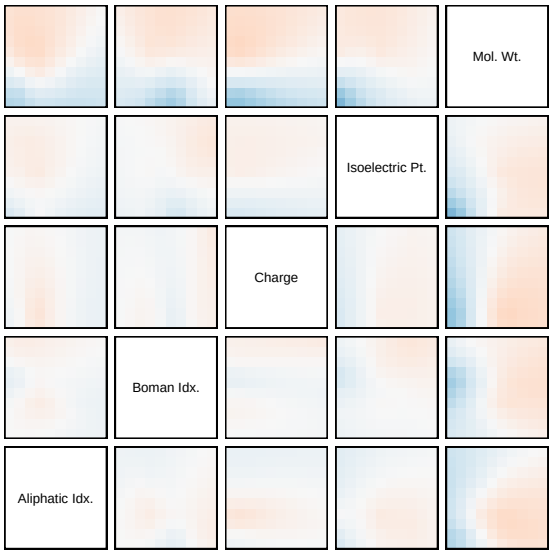


Figure A.5: The distribution of the length and molecular weight across the set of distinct CDRH3 sequences identified in each sample. Refer to Figure 5.3 for a description.

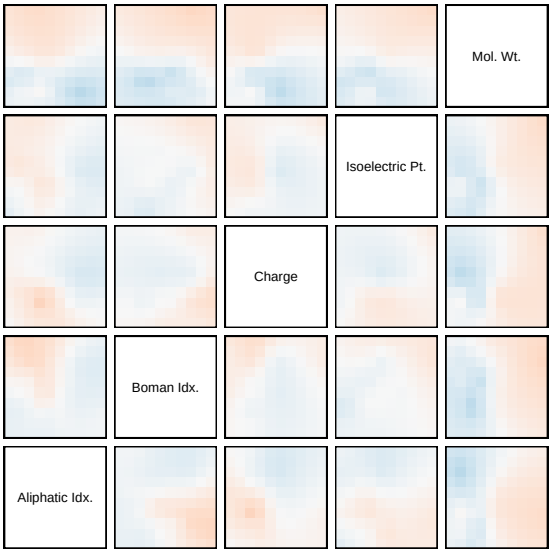
A.4 RBF-SVM Support Vector Projections



(a) Jagged1

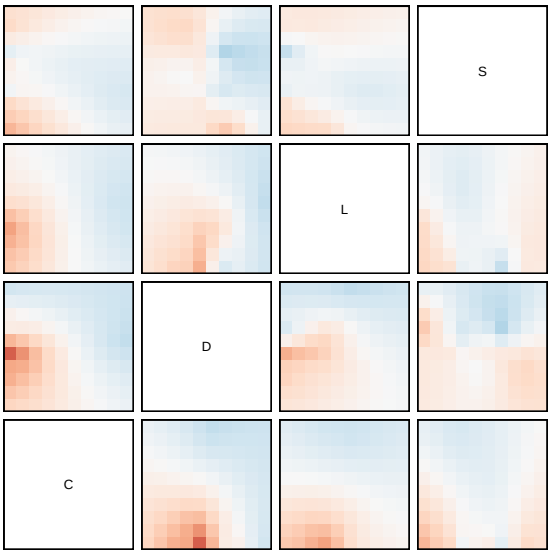


(b) Notch1

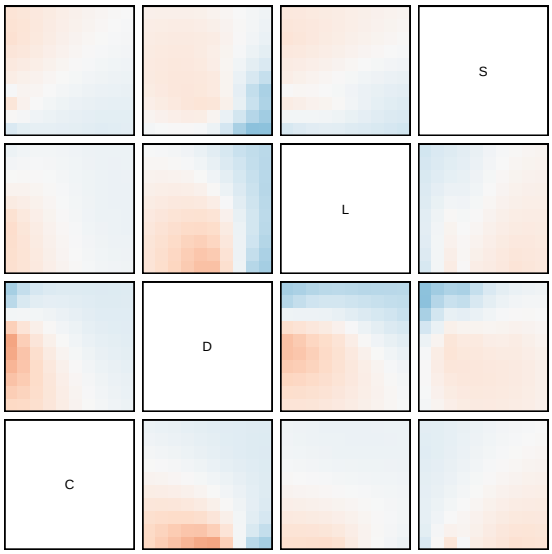


(c) Notch2

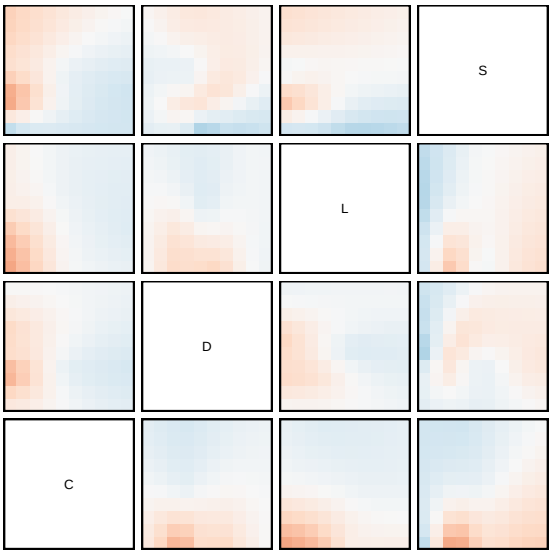
Figure A.6: RBF-SVM support vector projections for the physicochemical properties. Panels (a), (b), and (c) show the projections for the RBF-SVMs trained on datasets F-Jagged1, F-Notch1, and F-Notch2, respectively. In each facet, the dimension of the vertical axis is equal to the name of the feature in that row of facets. Likewise, the dimension of the horizontal axis is equal to the name of the feature in that column of facets. The range of each axis is from the minimum support vector to the maximum support vector. Values have been centered; therefore, colour is only an indication of relative magnitude.



(a) Jagged1

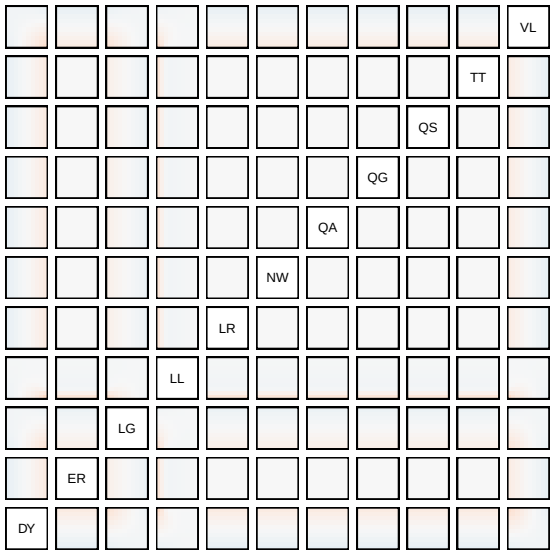


(b) Notch1

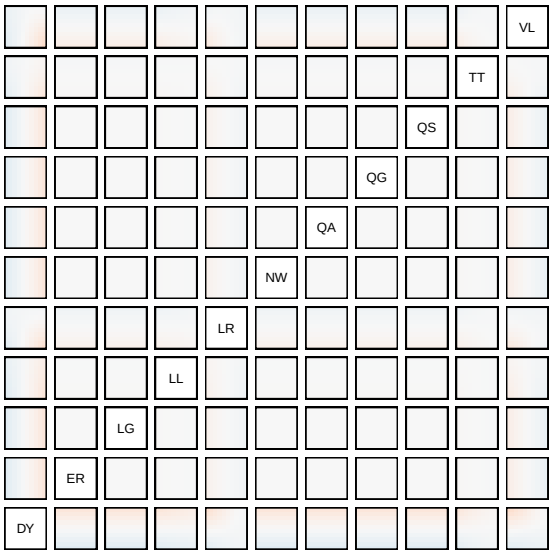


(c) Notch2

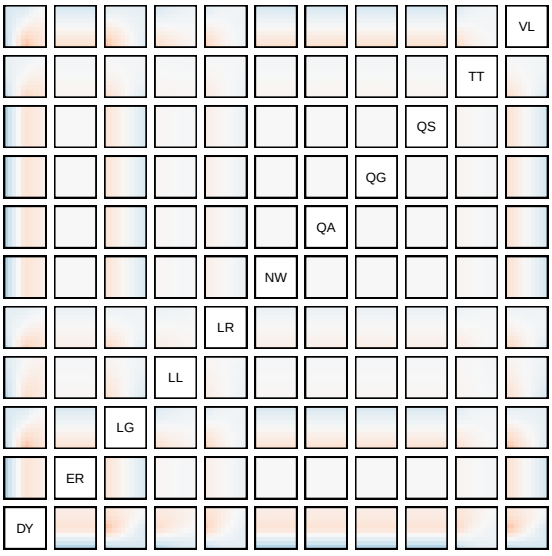
Figure A.7: RBF-SVM support vector projections for the amino acid compositions. Refer to Figure 5.14 for a description.



(a) Jagged1



(b) Notch1



(c) Notch2

Figure A.8: RBF-SVM support vector projections for the dipeptide counts. Refer to Figure 5.14 for a description.

A.5 SSK-SVM Dipeptide Contributions

Table A.6: Dipeptide contributions of the 7 SSK-SVM models.

Dip.	Axl	Jag1	Jag2	Mer	Not1	Not2	Not3
AA	0.14	0.10	0.56	-0.01	0.45	0.17	0.34
AC	-0.39		-0.08		-0.12	-0.22	
AD	-0.29	0.53	0.32	0.20	-0.19	0.66	1.12
AE	-0.11	-0.01				0.02	
AF	0.34	0.97	0.60	-0.17	-0.06	-0.32	0.37
AG	-1.34	1.19	0.68	0.31	0.59	-0.26	0.85
AH	0.59	-0.06	0.06	0.22	-0.35	-0.44	0.84
AI	-0.82	0.75	1.12	0.10	0.19	0.25	0.88
AL	-0.75	0.50	0.45	0.16	-0.36	0.57	0.65
AM	0.05	-0.85	0.38	-0.10	0.67	0.65	-0.13
AN				-0.22	0.07		
AP	0.09	-0.47	-1.56	0.02	0.61	0.29	0.43
AR	-0.01		-0.25	-0.01		-0.14	
AS	0.79	-0.71	0.61	-0.31	0.07	-0.50	2.32
AT	-0.19	-0.25	-0.07	-0.06	-0.03	-0.23	
AV	-0.32	0.58	-0.24	0.33	0.19	-0.22	0.42
AW	-0.67	2.40	-0.64	0.35	0.27	-0.34	-0.52
AY	-1.41	-0.57	0.95	0.35	0.36	0.72	0.74
CA	-0.03		-0.33	-0.02	-0.01	-0.04	
CC				-0.11		-0.03	
CD	-0.65	-0.45	-0.11	-0.01	-0.05	-0.00	-0.00
CF	0.02			-0.18	-0.05	-0.32	-0.02
CG	0.11		-0.00	-0.00		-0.18	-0.01
CH	0.01		-0.03			-0.02	
CI	-0.00		-0.00				
CL	-0.06	-0.09		-0.05	-0.06	-0.14	-0.01
CM	-0.59	-0.06	-0.19	-0.01	-0.00		
CP	-0.03						-0.03
CR	-0.00			-0.01			
CS	-0.06		-0.05			-0.31	-0.04
CT					-0.02		
CV	-0.05	-0.11	-0.00		-0.01	-0.08	-0.00
CW	0.03		-0.15		-0.03	-0.05	
CY	-0.39	-0.42	-0.41	-0.17	-0.33	-0.06	-0.07
DA	0.08	-0.22	-0.94				
DC			-0.02				
DD	0.02	-0.08	-1.50		-0.01	-0.02	
DF	-0.03		0.10		-0.10	-0.03	
DG	-0.14		-0.58	-0.11	-0.03	-0.05	
DH			-0.49				
DI	0.05		-0.43				
DL			-0.57	-0.06			
DM	-0.01	-0.13	-0.35				
DN		0.36					
DS		-0.36	-0.08		-0.05	-0.22	
DW	0.01		-0.05			-0.13	
DY	0.60	0.49	1.31	0.50	0.47	0.42	0.95
EL		-0.02					
EY	-0.16					0.03	
FA	-0.47	0.47	0.68	0.12	0.03	0.32	1.49

Dip.	Axl	Jag1	Jag2	Mer	Not1	Not2	Not3
FC		-0.13	-0.33	-0.21	-0.05		-0.04
FD	0.92	0.53	1.23	-0.30	-0.51	-0.51	-0.07
FF	-0.85	0.61	-0.43	-0.12	0.29	0.62	0.22
FG	0.12	0.04	-0.01	0.45	0.25	0.64	0.33
FH	0.65	-0.30	-0.47	0.13		-0.42	0.04
FI	0.03	-0.20	-0.27		0.02	-0.05	-0.21
FL	0.03	-0.29	-0.39	-0.18	0.03	0.17	0.60
FM	-0.00	-0.02	0.06	0.15	0.25	0.24	0.59
FN		0.00	-0.10		-0.04		
FP	0.49	-0.80	0.04	0.04	0.22	0.17	0.16
FR	-0.15			-0.01			
FS	1.00	0.57	0.14	-0.08	-0.76	0.42	0.41
FT			-0.03	-0.02	-0.05	-0.10	-0.10
FV	-0.17	0.25	-1.22	-0.01	-0.18	-0.13	0.06
FW	0.14	-1.02	-0.44	0.20	0.83	0.06	0.33
FY	0.40	1.62	0.15	-0.07	0.50	0.43	0.92
GA	0.80	-0.18	0.48	-0.56	-0.54	-0.27	-1.44
GC	-0.02	-0.08	-0.22	-0.02	-0.24	-0.37	-0.07
GD	0.81	1.06	0.95	0.59	0.92	0.47	0.03
GE						0.00	
GF	0.19	0.50	0.64	0.08	-0.47	0.21	0.02
GG	-0.34	0.19	0.24	0.26	0.97	0.08	0.45
GH	0.55	-1.35	-1.52	0.14	0.17	-0.16	0.86
GI	1.10	0.85	0.23	0.21	0.54	0.09	0.05
GK		-0.25					
GL	-0.34	0.68	0.12	0.15	-0.13	0.00	-0.60
GM	0.50	0.44	0.01	0.52	0.72	0.08	0.47
GN		0.15	-0.06		0.17	-0.22	
GP	-1.12	0.65	0.10	-0.19	0.24	0.19	0.31
GQ			-0.40				
GR	-0.11	-0.06	-0.52	-0.00		-0.11	
GS	0.11	1.04	0.43	-0.58	-0.02	-0.24	1.54
GT	-0.04	-0.47	-0.11	-0.02		-0.27	-0.26
GV	-0.04	-0.06	0.56	0.10	0.19	-0.26	0.93
GW	-1.09	-0.08	0.88	0.42	0.17	0.18	0.46
GY	0.43	0.39	-0.07	0.28	-0.20	-0.52	0.94
HA	0.05	1.45	0.34	0.24	0.40	0.18	-0.34
HC	-0.02	-0.09					
HD	0.21	0.21	0.06	-0.01	-0.04	0.12	0.07
HF	0.04	-0.19	0.11	0.06	-0.28	-0.06	-0.09
HG	-1.35	0.07	0.83	0.04	-0.14	1.50	0.18
HH	-0.11	0.43	0.50	0.05	0.15	0.19	-0.65
HI	0.10	0.15	0.67			0.04	0.43
HL	0.01	-0.02	-0.38	0.01	-0.10	0.20	0.13
HM	0.49	0.58	-0.06	0.02	0.13	0.01	-0.28
HP	-0.17	-0.55	0.53	-0.24	0.24	0.19	0.91
HR			-0.09				
HS	-0.59	-0.88	-0.89	0.44	0.06	-0.50	-0.37
HT	-0.00		-0.01				
HV	-0.11	0.56	0.74	0.07	-0.13	0.15	0.07
HW	-0.02	-0.43	-0.52	-0.06	-0.34	0.76	-0.45

Table A.6 (cont.)**:** Dipeptide contributions of the 7 SSK-SVM models.

Dip.	Axl	Jag1	Jag2	Mer	Not1	Not2	Not3	Dip.	Axl	Jag1	Jag2	Mer	Not1	Not2	Not3
HY	-0.15	0.98	1.46	0.04	0.33	0.03	0.08	MG	-0.18	-0.28	-0.23				-0.10
IA	-0.52		0.00	-0.02	0.22	0.00	0.01	MI	-0.01						-0.36
IC	-0.04							ML	-0.51	-0.01	-0.36		-0.14		
ID	0.15	1.09	1.34	0.24	0.49	0.57	0.99	MM	-0.01						
IE						0.00		MN				-0.36		-0.36	
IF							0.03	MS			-0.00				
IG	0.17		-0.27	-0.02		-0.07		MT	-0.04				-0.05		
IH	-0.00		-0.11					MV	-0.03						
II			-0.01					MW	-0.18						
IL	-0.12		-0.07	-0.01		-0.05	0.00	MY	-0.30	-0.22	0.08	-0.13	0.61	0.29	0.03
IM	-0.30		0.02	-0.01	0.13	0.00		NA	0.36		-0.07				
IN					0.27			ND	0.13		-0.00				
IP						0.04		NF			-0.18				
IR	-0.01					-0.22		NL	0.22						
IS	-0.01		-0.18					NW			-0.34				
IT	-0.06					-0.36		NY	0.08		-0.11	-0.36	0.23	-0.36	
IV	-0.35				0.36	-0.13		PA	1.13	-0.41	-0.24	0.54	0.51	0.23	1.08
IW	0.01						0.18	PC	-0.05			-0.01			
IY	0.08	0.66	0.49	-0.07	0.25	0.38	0.22	PD	-0.08	0.33	-0.07	0.07	0.26	0.20	0.27
KA		-0.02						PE						0.01	
KD		-0.01						PF	-1.12	-0.31	0.12	-0.06	0.55	0.16	0.36
KF		-0.15						PG	-0.58	0.31	0.32	-0.78	0.56	0.67	0.68
KG		-0.05						PH	1.14	0.27	-0.65	-0.02		0.30	-0.04
KK		-0.26						PI	0.09	0.03	0.42	-0.00		0.01	0.08
KM		-0.01						PL	-0.61	0.62	-0.12	-0.03	0.22	0.18	0.14
KP		-0.41						PM	0.13	-0.43	0.39	0.09	0.14	0.03	-0.03
KS		-0.09						PN		0.01					
KW		-0.03						PP	0.69	0.18	-0.65	0.03		0.01	0.41
KY		-0.25						PR		-0.03	-0.03	-0.00			
LA	-0.33	-0.08	-0.39	-0.07	-0.04	-0.01	-0.00	PS	0.08	0.19	0.33	-1.16	0.66	0.57	1.39
LC	-0.30	-0.63			-0.03	-0.05		PT	-0.25	-0.04		-0.01			
LD	0.66	0.96	0.24	0.35	0.01	0.31	0.25	PV	1.01	-0.52	0.56	-0.09		0.26	0.64
LF	-0.06		-0.02	-0.08	-0.10	-0.14	-0.52	PW	-0.77	0.13	0.96	0.12	0.65	0.06	-0.03
LG	-1.15	-0.12	-0.58	-0.21	-0.12	-0.04	-0.13	PY	1.33	0.60	1.17	0.36	0.20	0.30	0.59
LH	-0.01	-0.15	-0.24		-0.06		-0.00	QA			-0.40	-0.02			
LI	-0.03		-0.34	-0.04	-0.21		-0.01	QC					-0.04		
LL	-0.57	-0.58	-0.26	-0.54	-0.12	-0.02	-0.02	QD			-0.14	-0.00	-0.01		
LM	-0.20	-0.06	-0.15	-0.11	-0.00			QG				-0.13			
LN		0.22						QL			-0.02		-0.02		
LP	-0.54		-0.22	-0.05				QM			-0.22	-0.00			
LR	-0.15	-0.03	-0.36	-0.22	-0.02	-0.21		QS			-0.12	-0.13			
LS	-0.75	-0.13	-0.26	-0.25		-0.05	-0.06	QV				-0.03			
LT	-0.49		-0.01	-0.27	-0.01	-0.01		QW			-0.07		-0.06		
LV	-0.19	-0.52	-0.44	-0.05	-0.02	-0.21	-0.12	QY			-0.09	-0.01	-0.01		
LW	-1.07	-0.71	-0.26		-0.00	-0.08	-0.04	RA	-0.31	-0.00	-0.55	-0.03			
LY	-1.01	-0.10	-0.44	0.10	-0.12	0.17	-0.15	RC				-0.05			
MA	-0.30						-0.22	RD	-0.22	-0.03	-0.59	-0.06	-0.01	-0.21	
MC	-0.11		-0.13					RF	-0.01	-0.07	-0.20	-0.16	-0.01	-0.22	
MD	-0.31	-0.09	0.07	0.15	1.02	0.81	0.40	RG	-0.19	-0.27	-0.36	-0.02		-0.22	
ME	-0.16	-0.02				0.03		RH	-0.04		-0.05				
MF	-0.04		-0.36				-0.04	RI	-0.06		-0.13				

Table A.6 (cont.)**:** Dipeptide contributions of the 7 SSK-SVM models.

Dip.	Axl	Jag1	Jag2	Mer	Not1	Not2	Not3	Dip.	Axl	Jag1	Jag2	Mer	Not1	Not2	Not3
RK		-0.15						VH	0.55	0.21	1.23	0.02	0.10	0.25	-0.51
RL	-0.42	-0.04	-0.06	-0.05		-0.13		VI	-0.22	0.24	-0.82	0.04	0.06	-0.15	-0.15
RM	-0.30	-0.04	-0.16	-0.10				VK		-0.09					
RP	-0.17	-0.03						VL	-0.35	-0.18	0.51	-0.30	-0.04	-0.26	-0.18
RR	-0.01	-0.02						VM	0.13	0.72	0.10	0.23	0.14	-0.39	0.18
RS	-0.04	-0.19	-0.43	-0.15		-0.02		VN			-0.28		0.04	-0.13	
RT	-0.09					-0.01		VP	-0.13	-0.39	-0.06	0.01	0.20	0.21	0.00
RV			-0.15	-0.07	-0.02	-0.36		VR	-0.19	-0.26		-0.01	-0.01	-0.14	
RW	-0.21	-0.02	-0.50			-0.00		VS	0.38	0.76	-0.18	-0.13	0.34	0.23	1.62
RY	-0.20	-0.04	-0.17	-0.10	-0.00	-0.12		VT	-0.01		-0.01	-0.06	-0.25	-0.26	
SA	1.00	1.07	0.77	0.16	0.01	0.87	0.84	VV	0.19	0.28	-0.83	-0.02	0.25	0.58	0.64
SC	-0.64	-0.07	-0.14	-0.08	-0.08		-0.01	VW	-0.55	1.06	0.68	0.22	-0.13	0.34	0.16
SD	0.58	-0.84	0.12	0.21	-0.36	0.32	0.21	VY	0.58	0.91	-0.89	0.40	0.26	-0.78	0.07
SE	-0.04	-0.01						WA	1.24	0.75	-0.61	0.61	0.56	0.64	0.97
SF	0.02	-0.34	1.04	0.22	-0.06	-0.73	0.45	WC	-0.13	-0.07	-0.19		-0.08	-0.14	-0.02
SG	-0.30	-0.21	0.61	0.36	1.13	0.75	0.62	WD	0.04	-0.04	0.35	-0.03	0.14	-0.03	-0.08
SH	0.72	-0.61	-1.22	0.17	0.58	0.24	1.04	WE		-0.00					
SI	0.24	0.74	0.40	0.01	0.24	0.22	0.20	WF	-0.03	-0.64	-0.47	0.24	0.38	-0.15	0.58
SK		-0.41						WG	-0.10	0.30	0.53	-0.31	-0.07	-0.07	0.54
SL	0.24	-0.63	-0.01	-0.08	-0.20	0.16	0.39	WH	0.31	-0.33	-0.22	0.20	-0.23	-0.43	0.13
SM	-0.60	-1.17	-0.63	0.05	-0.07	0.42	-0.05	WI	-0.29	0.52	0.66		0.09	0.23	0.33
SN	0.36	0.08	-0.17		0.08			WL	0.00	-0.06	-0.74	-0.01	-0.20	-0.16	-0.68
SP	-1.55	0.10	0.65	-0.05	0.12	0.26	-0.45	WM	-0.04	0.12	0.73	0.16	0.27	0.10	-0.19
SR	-0.33	-0.02	-0.40	-0.03		-0.02		WN		0.05					
SS	0.61	-0.04	0.87	0.07	0.90	0.31	0.08	WP	0.68	-0.84	-0.83	-0.16		0.73	0.74
ST	-0.02	-0.38	-0.06	-0.19	-0.00	-0.02		WQ					-0.06		
SV	0.32	-0.25	-0.21	-0.71	0.57	0.76	0.28	WR	-0.30	-0.02	-0.08			-0.05	
SW	0.00	-0.74	0.19	0.06	-0.55	0.03	0.25	WS	-0.05	0.09	-0.73	-0.10	0.09	0.29	-0.08
SY	-1.01	-0.05	-0.30	-0.11	0.38	-0.44	-0.20	WT	0.15	-0.15	-0.19			-0.02	
TA	-0.33	-0.70	-0.35	-0.00	-0.25	-0.02	-0.13	WV	-0.04	-0.93	0.56	0.63	0.02	0.02	1.18
TC	-0.08		-0.01					WW	-0.44	0.80	-0.23	0.31	0.27	-0.36	0.34
TD	-0.27	-0.50	-0.22	-0.12	-0.04	-0.12	-0.01	WY	0.57	0.40	1.40	0.27	0.32	-0.31	0.02
TF		-0.00	-0.13		-0.05	-0.01	-0.02	YA	-0.57	0.35	-0.45	-0.17	0.09	0.22	0.44
TG	-0.27	-0.09	-0.21	-0.02		-0.15	-0.06	YC	-0.26	-0.19		-0.06	-0.32		-0.01
TH	-0.03		-0.04					YD	-1.14	-0.33	1.27	-0.45	-0.10	-0.52	0.39
TI	-0.26	-0.33	-0.02			-0.16		YE	-0.08	-0.00				0.02	
TK		-0.05						YF	0.77	0.10	0.08	-0.63	0.33	-0.11	0.84
TL	-0.51		-0.21	-0.17	-0.05	-0.18		YG	0.49	-1.11	0.03	0.20	1.05	0.45	0.41
TM	-0.17	-0.09	-0.02	-0.13	-0.06		-0.16	YH	0.84	0.17	0.31	-0.16	-0.00	0.56	0.08
TP		-0.01	-0.16					YI	-0.47	0.02	0.37	-0.01	0.60	0.22	0.13
TR	-0.09	-0.15				-0.36		YL	-0.08	0.37	-0.14	-0.07	0.33	-0.28	0.38
TS	-0.10	-0.06		-0.24		-0.02		YM	-1.26	0.79	2.15	-0.93	-0.09	0.32	-0.14
TT	-0.06			-0.28	-0.04	-0.01		YN		0.04		-0.21	-0.01	-0.08	
TV	-0.02	-0.26	-0.06	-0.00	-0.12	-0.22		YP	-0.09	-0.98	0.11	-0.27	0.66	0.03	-0.31
TW	-0.19	-0.26	-0.03			-0.22		YQ			-0.18	-0.21	-0.04		
TY	0.07	-0.31	-0.14	-0.07	-0.03	-0.20	-0.01	YR		-0.08	-0.33	-0.38		-0.03	
VA	1.03	0.55	1.47	-0.33	0.46	0.45	0.96	YS	0.35	-0.05	-0.01	0.39	0.37	0.31	0.41
VC	-0.10	-0.15	-0.05			-0.08		YT	-0.11	-1.01	-0.26	-0.21	-0.00	-0.04	
VD	-0.07	0.10	-0.77	0.19	0.03	-0.23	-0.18	YV	0.03	-0.51	0.22	0.10	0.79	0.42	0.99
VF	0.36	0.13	-0.62	0.15	0.24	0.70	0.51	YW	0.69	0.02	-0.60	-0.35	0.43	-0.17	0.47
VG	-1.05	1.61	0.75	-0.01	0.05	0.05	0.48	YY	0.93	0.23	0.37	-0.65	0.14	0.29	1.12

A.6 Average Classification Accuracies

Table A.7: The average classification accuracy of the compared machine learning models. The accuracy was measured using 10-fold cross-validation.

Library	Target	Classifier	Accuracy	Library	Target	Classifier	Accuracy
F	Axl	NB	76.05	S	Axl	NB	71.04
F	Axl	LM	65.79	S	Axl	LM	77.76
F	Axl	ANN	78.68	S	Axl	ANN	79.53
F	Axl	RBF-SVM	81.32	S	Axl	RBF-SVM	82.15
F	Axl	RF	83.68	S	Axl	RF	84.21
F	Axl	RBF-SVM (t)	75.00	S	Axl	RBF-SVM (t)	79.24
F	Axl	SSK-SVM (t)	87.11	S	Axl	SSK-SVM (t)	88.31
F	Jagged1	NB	71.02	S	Jagged1	NB	58.64
F	Jagged1	LM	51.58	S	Jagged1	LM	55.73
F	Jagged1	ANN	70.98	S	Jagged1	ANN	79.64
F	Jagged1	RBF-SVM	69.80	S	Jagged1	RBF-SVM	73.82
F	Jagged1	RF	74.58	S	Jagged1	RF	74.82
F	Jagged1	RBF-SVM (t)	71.37	S	Jagged1	RBF-SVM (t)	68.36
F	Jagged1	SSK-SVM (t)	72.63	S	Jagged1	SSK-SVM (t)	76.82
F	Jagged2	NB	78.28	S	Jagged2	NB	76.43
F	Jagged2	LM	71.68	S	Jagged2	LM	64.45
F	Jagged2	ANN	81.38	S	Jagged2	ANN	83.02
F	Jagged2	RBF-SVM	78.30	S	Jagged2	RBF-SVM	85.51
F	Jagged2	RF	83.67	S	Jagged2	RF	89.95
F	Jagged2	RBF-SVM (t)	80.15	S	Jagged2	RBF-SVM (t)	77.96
F	Jagged2	SSK-SVM (t)	80.99	S	Jagged2	SSK-SVM (t)	91.17
F	Mer	NB	78.53	S	Mer	NB	61.20
F	Mer	LM	61.29	S	Mer	LM	56.00
F	Mer	ANN	89.30	S	Mer	ANN	62.40
F	Mer	RBF-SVM	85.15	S	Mer	RBF-SVM	58.40
F	Mer	RF	91.69	S	Mer	RF	60.80
F	Mer	RBF-SVM (t)	88.71	S	Mer	RBF-SVM (t)	60.60
F	Mer	SSK-SVM (t)	92.83	S	Mer	SSK-SVM (t)	55.60
F	Notch1	NB	85.44	S	Notch1	NB	79.03
F	Notch1	LM	60.82	S	Notch1	LM	62.78
F	Notch1	ANN	93.13	S	Notch1	ANN	84.44
F	Notch1	RBF-SVM	92.36	S	Notch1	RBF-SVM	91.67
F	Notch1	RF	93.08	S	Notch1	RF	90.69
F	Notch1	RBF-SVM (t)	93.08	S	Notch1	RBF-SVM (t)	86.81
F	Notch1	SSK-SVM (t)	89.40	S	Notch1	SSK-SVM (t)	87.92
F	Notch2	NB	78.68	S	Notch2	NB	66.96
F	Notch2	LM	51.98	S	Notch2	LM	54.38
F	Notch2	ANN	76.37	S	Notch2	ANN	75.07
F	Notch2	RBF-SVM	67.25	S	Notch2	RBF-SVM	72.75
F	Notch2	RF	83.30	S	Notch2	RF	82.91
F	Notch2	RBF-SVM (t)	71.81	S	Notch2	RBF-SVM (t)	71.60
F	Notch2	SSK-SVM (t)	87.97	S	Notch2	SSK-SVM (t)	84.12
F	Notch3	NB	82.27	S	Notch3	NB	77.72
F	Notch3	LM	65.45	S	Notch3	LM	52.11
F	Notch3	ANN	88.86	S	Notch3	ANN	90.96
F	Notch3	RBF-SVM	87.27	S	Notch3	RBF-SVM	94.15
F	Notch3	RF	90.23	S	Notch3	RF	91.55
F	Notch3	RBF-SVM (t)	89.32	S	Notch3	RBF-SVM (t)	90.96
F	Notch3	SSK-SVM (t)	89.55	S	Notch3	SSK-SVM (t)	88.86