

SCHOOL OF ARCHITECTURE, COMPUTING AND ENGINEERING

Computer Science and Informatics Subject Area

**A Simple Guide to Biotechnology and Machine Learning**

**Gintaras Koncevicius**

**u1334444**

A report submitted in part fulfilment of the degree of

BSc (Hons) in *Software Engineering*

Supervisor**:** Ray Ruocco

CN6103

28 June 2016

C

ontents

[Abstract 4](#_Toc450325811)

[Chapter 1: Introduction 5](#_Toc450325812)

[1.1 Drug Development 5](#_Toc450325813)

[1.2 Cell Biology 6](#_Toc450325814)

[1.3 Machine Learning 7](#_Toc450325815)

[Chapter 2: Literature Review 8](#_Toc450325816)

[2.1 Ethical, legal and social issues 8](#_Toc450325817)

[2.2 Earlier and not machine learning techniques 8](#_Toc450325818)

[2.3 Techniques from biology examples 10](#_Toc450325819)

[2.4 Later techniques 12](#_Toc450325820)

[Chapter 3: Presentation 19](#_Toc450325821)

[3.1 The problem 19](#_Toc450325822)

[3.2 Proposed solution 20](#_Toc450325823)

[3.3 Analysis and Specification 21](#_Toc450325824)

[3.4 Methodology 22](#_Toc450325825)

[3.5 Trustworthiness 23](#_Toc450325826)

[3.6 Design 23](#_Toc450325827)

[3.7 Implementation 27](#_Toc450325828)

[3.8 Testing 31](#_Toc450325829)

[3.9 Results 37](#_Toc450325830)

[Chapter 4: Evaluation 39](#_Toc450325831)

[Chapter 5: Conclusion 40](#_Toc450325832)

[References 41](#_Toc450325833)

[Appendix A - Initial Project Proposal 44](#_Toc450325834)

[Appendix B - Final Project Proposal 45](#_Toc450325835)

[Appendix C – Further Reading 46](#_Toc450325836)

[5.1 Facets of Trustworthiness 46](#_Toc450325837)

[5.2 Ethical, legal and social issues 49](#_Toc450325838)

[5.3 The problem solution reasoning 50](#_Toc450325839)

[5.4 The web app implementation of bioassay experiment 51](#_Toc450325840)

[Appendix D– Project planning and PDP 60](#_Toc450325841)

[5.5 Personal Development Plan 60](#_Toc450325842)

[5.6 Strengths and weaknesses: 60](#_Toc450325843)

[5.7 My Objectives 60](#_Toc450325844)

[5.8 Project Aim 61](#_Toc450325845)

[5.9 Project Objectives 61](#_Toc450325846)

[5.10 Research and Development Objectives 61](#_Toc450325847)

[5.11 Planning 61](#_Toc450325848)

[5.12 Project milestones 61](#_Toc450325849)

[5.13 Project activities 61](#_Toc450325850)

Abstract

Advancement of science caused more cross-disciplinary and narrow fields to arise, so that more new topics could be covered and generate experts in various areas. One of these fields started from biology and with the rise of computers became bioinformatics. With the need to know about biotechnology and computers the knowledge gap between experts and beginners kept increasing. This project report offers a partial solution to the problem by educating about biotechnology and machine learning together with Java implementation of SVM interpretation.

Introduction describes bioinformatics topics which cover the problems like the complexity of finding the problem of disease because protein structure, biochemical pathways and even more has to be taken in to consideration and about how the big data gave rise to new techniques to manage it, because mathematical statistics started to become more and more complicated.

Literature review mentions several bioinformatics papers about biological data analysis with various machine learning techniques, while starting to explain older techniques, then ending with new ones like Bayesian networks and Support Vector Machines. This chapter touches on how the big data needs to be approximated, reduced, modelled, discretized, normalized or abstracted to some level and gives the examples on what has been taken from nature to create evolutionary algorithms and artificial neural networks. From the literature review it becomes clearer that SVM has been shown to be intuitive to use high precision algorithm. SVM is further explained with math and simple diagrams.

Chapter three further explains SVM as its distinct interpretation in Java with hopes that this way it would be easier to understand. The interpretation idea asks to look at SVM hyperplane in 2D and see that it is a line, which separates two sides, then imagine that a line can be two parallel lines very close to each other and that these lines could be just edges of very large circles with opposite centers very far away from each other. That is how Least Similar Spheres (LSS) idea was discovered. Further paragraphs explain how this machine interpretation was designed, that it was used on flower Iris dataset by Marshall M. (2016) and large bioassay dataset by A. Schierz (2009) and its results were compared with Weka SVM and R SVM.

In conclusions it is mentioned that although LSS had unexpectedly good results, the main point of it was to show that mathematical ideas could be interpreted differently, while still being used as they are and that machine learning might not be as difficult as it looks, because you can have your own interpretations. Some of the examples in LSS were the distance measure viewed as similarity, support vectors as the most similar and separated spaces with hyperplanes as spheres with radiuses. Further mentioned that literature review shown SVM is one of the best techniques. From implementation and explanation of SVM interpretation it could be concluded that it does not really matter whether the data is biological or not and that it can be useful even if precision is not high, because it can reduce the search space for further confirmation.

# Introduction

Curiosity and desire for discovering new things is one of human traits. These things could be called problems, which need solutions. The more problems are solved, the more can be achieved and technology can help saving time and effort. Preventing sickness and diseases would allow more minds to join the progress of discovery. *“Drugs are essential for the prevention and treatment of disease”* (S. Mandal et al., 2009, p90) and their development is one of the biggest time and resource consuming problems. More sciences involved in to this problem led to new mixed fields like biophysics, biochemistry, biotechnology and recently, bioinformatics, which emerged because *“the enormous amount of data gathered by biologists—and the need to interpret it— requires tools that are in the realm of computer science”* (J. Cohen, 2004, p123). Bioinformatics is one of the fields where different sciences collaborate to solve its problems and increasing amount of data about it. *“This area has arisen from the needs of biologists to utilize and help interpret the vast amounts of data that are constantly being gathered in genomic research—and its more recent counterparts, proteomics and functional genomics”* (J. Cohen, 2004, p122). This field uses computer science and technology to automate parts of it and improves the process of discovery by trying to reduce required time and resources needed to do it. The rise of this science and big data lead to more and more sophisticated statistics, which got more and more complicated. Data mining and machine learning emerged as solution to the difficulties in statistics with multi-dimensional spaces caused by large amount of attributes. Machine learning started from combining programming and different mathematical ideas ranging from regression (drawing a function, which can describe points on graph), clustering (finding distinct groups), to unique ideas taken from examples of the way humans think (decision trees, Apriori Algorithm) and how process of thinking works in biology (neural networks). Further improvements and their combinations lead to Support Vector Machines and Bayesian networks. The most recent ones use different combinations of multiple techniques. In machine learning *“each area involves one or more reasoning problems for which significant expertise exists in the AI community, such as simulation, planning, redesign, diagnosis, and learning”* (D. Karp et al., 1994, p8). Machine learning tools are used in bioinformatics, their use and improvement *“will have numerous benefits such as efficiency, cost effectiveness, time saving”* (S. Mandal et al., 2009, p90).

## Drug Development

Drug engineering just like software engineering has its methodologies. The top parts are called “*Discovery’, ‘Development’ and ‘Registration’ phases. The ‘Discovery’ phase, routinely three to four years, involves identification of new therapeutic targets, lead finding and prioritisation, lead optimisation and nomination of new chemical entities (NCEs)”* (J. Wang et al., 2004, p73). Discovering new drugs is very large and costly process and *“to address this issue, several multidisciplinary approaches are required for the process of drug development, including structural biology, computational chemistry, and information technology, which collectively form the basis of rational drug design”* (Y. Wang et al., 2015, p489). The discovery phase is where biotechnology and bioinformatics are mostly used. This phase has its own main parts called ADME, which is *"absorption, distribution, metabolism, and excretion"* (S. K. Balani et al., 2005, p1). Another technique in drug discovery focuses on 3D structures of biomolecules and is called structure based drug design or SBDD. “*SBDD provides insight in the interaction of a specific protein-ligand pair, allowing medicinal chemists to devise highly accurate chemical modifications around the ligand scaffold”* (V. Lounnas et al., 2013, p1). Together these rules allow better drug discovery, because it is not enough for a drug to be effective, it should also be less toxic (T in ADME/T stands for toxicity). In reality, a drug can be effective but not satisfy all the requirements of ADME/T and therefore not released for medical use. “*Investigation of terminated projects revealed that the primary cause for drug failure in the development phase was the poor pharmacokinetic and ADMET (ADME+Toxicity) properties rather than unsatisfactory efficacy”* (J. Wang., 2004, p73). There are various drug design techniques, some of them include structure based drug design, target based drug design and more recent rational drug design which involves multiple disciplines and techniques, it “*can be applied to develop drugs to treat a wide variety of diseases and can also be used for designing drugs for disease prevention”* (S. Mandal et al., 2009, p90). Drug discovery is the beginning of drug development and involves identifying problem or drug target, which *“is a biomolecule which is involved in signaling or metabolic pathways that are specific to a disease process”* (S. Mandal et al., 2009, p90). Identifying which molecules or ligands can bind to the receptors of the target is one of the most complicated parts of drug discovery.

## Cell Biology

Life forms are made of cells and *“cells are complex molecular machines contained within phospholipid membranes that isolate a unique chemical environment”* (E. Yoruk et al., 2011, p1). These machines are made from big protein, average peptide, smaller amino acid and very small molecules like H2O. Each big molecule is made from the atoms held by the atomic force. *”The atomic force field model describes physical systems as collections of atoms kept together by interatomic forces”* (J. Meller, 2001, p2). The proteins are also machines and *”are made of long chains of amino acids which in their natural environment (in solution) fold up into simple "secondary" structures, like helices, and then by further folding into higher-order structures”* (J. Fox et al., 1994, p290). While big molecules with more than 50 amino acids are called proteins,” *short strings of amino acids, called peptides”* (W. Noble, 2003, p23). *”Peptides can be considered to be up to 50 amino acids in length, with proteins being larger than this”* (C. Walle, 2011, p4). And *“there are twenty common amino acids”* (F. Altschul, 2011, p8). Cells can be divided in two communication parts, the inside and the outside. Both sides involve sequences of bio-reactions. *“The metabolism of a cell is the set of bioreactions that its enzymes can catalyse* and *such a sequence of reactions is called a path way”* (D. Karp et al., 1994, p7). These pathways are like a constantly moving and floating queues of large and small molecules interacting with each other, they are called biochemical pathways *“and, in reality, metabolic, signaling, and regulatory pathways interact and intersect in the course of cellular growth and activity”* (R. Gostner et al., 2014, p16:2). The inside is made of biochemical pathways, which allow communication between smaller inner organelles. *”Living cells are complex systems whose growth and existence depends on thousands of biochemical reactions”* (D. Karp et al., 1994, p2). A lot of biochemical reactions need help or catalysts for them to happen, these are called enzymes. Enzymes *“facilitate the association of several molecules to form a complex, and it lowers the energy barrier required for the bond rearrangements that constitute a reaction”* (D. Karp et al., 1994, p5). The outside on membrane’s surface receptors and ligands are used to communicate with other cells. How strongly ligand molecule reacts with receptor is called binding affinity. *”Binding affinity represents the strength of association between the ligand and its receptor protein”* (M. Ashtawy et al., 2012, p1301) that’s why*” the recognition of signal peptides is important for the development of new drugs”* (W. Noble, 2003, p12) and” *finding the structure and the fold of a protein is very important since it helps to understand the functions* (A. Chinnasamy et al., 2003, p1).

All these and even more has to be taken in to consideration when trying to find the disease or a problem with cells. Because of that much of complexity even todays computing power is not enough. Finding similar protein structures can also reduce the problem and machine learning can help recognize them.

## Machine Learning

Multi-disciplinary approach may increase reliability and effectiveness of problem solving, but it becomes more and more difficult to grasp it all as a whole. Increasing amount of data is pushing the limits of how much more should be done in the same amount of time to be able to expand the knowledge, which could possibly lead to the problem solutions. “*In the era of big data, a necessary goal is the ability to use rapidly accumulating data to pinpoint potential ADME/T issues before entering late-stage development.”* (Y. Wang et al., 2015, p508). The big data also gave rise to new techniques for managing it, because mathematical statistics started to become too complicated. That led to various machine learning approaches, which can use large amounts of data to learn and then predict the outcome of unknown new data. *”Computational analysis of biological data obtained in genome sequencing and other projects is essential for understanding cellular function and the discovery of new drugs and therapies”* (C. Ding et al., 2000, p349). The hopes are not lost because of how successful machine learning was and still is. As in the early years scientists trusted computers will help understand big problems saying that” *it is now generally accepted that modern molecular biology research needs many different types of software to support the management, analysis and interpretation of data* (J. Fox et al., 1994, p287) and in later years hoping that *“progress in reliable computational methods is greatly anticipated”* (M. Blaszczyk et al., 2015).

# Literature Review

Automating or simulating parts of drug discovery with machine learning involve its various branches and different techniques used for each one and it is because the problem is too big for one field to solve it. *“The interaction between drugs and the human body is a bidirectional process: drugs affect the human body, resulting in receptor inhibition, activation, and signal pathway blocking, and the human body disposes of drug by absorption, distribution, metabolism, and excretion”* (Y. Wang et al., 2015, p495). This chapter will review how different drug discovery problems like protein folding and genome sequencing are tackled with machine learning techniques from several papers by various authors.

## Ethical, legal and social issues

Genome sequencing has always been one of the biggest issues in biotechnology, because it is on the edge of how much it costs for private companies to develop it and that biological data has to be property of everyone in the world and therefore should be public. Public data could be stolen if not protected and then used as private product again. To prevent that *“in Europe legal protection is given to databases by the European Directive on The Legal Protection of Databases”* (C. McCubbin, 2003, p250). More research in to legal issues might be needed, but this project focus is on education. Further considerations on these issues are added to the Appendix C.

## Earlier and not machine learning techniques

Simplest and earliest form use of computing in biotechnology was knowledge based systems, which *“are being used to represent, manage and maintain, over time, knowledge obtained by interpreting the raw data stored in databases”* (J. Fox et al., 1994, p288), earlier interpreting was done by experts using statistics and rule based systems. Fox et al. (1994) mentions the use of regular expressions which extract sentences from long genetic sequences and can help identify the known parts. It is quite old technique widely used in scripting languages from Bash to JavaScript. From math and statistics arise regression, clustering and decision tree algorithms. Ashtawy et al. (2012) compared different machine learning techniques ranging from clustering K-Nearest Neighbour to regression, decision trees and SVM, which had best results. J. Cohen (2004) mentions the use of microarrays and says that *“a powerful new tool available in biology is microarrays. They allow determining simultaneously the amount of mRNA production of thousands of genes “(*J. Cohen, 2004, p129). In other words, microarrays are RNA slices which have known properties, then they are mixed with test substances and loaded in to computer using laser scanners to analyse the results. There are various ways to use micro arrays, one of them is when *“data from many separate microarray experiments are collected into a single matrix, indexed by gene (row) and experiment (column)”* (W. Noble, 2003, p15). Binding affinity can be used in *“scoring function SF, a mathematical or predictive model that produces a score representing the binding free energy of a binding pose”* (M. Ashtawy et al., 2012, p1302). Because biotechnologists need a way to evaluate or compare potential drug interactions with targets, there are various scoring functions and *“most SFs in use today can be categorized as either force-fieldbased, empirical, or knowledge-based SFs, but none were based on sophisticated machine-learning (ML) algorithms”* (M. Ashtawy et al., 2012, p1302). Nevertheless scoring functions are used in the latter algorithms, there are *”steady gains in performance of ML based SFs, in particular for those based on RF and BRT, as the training set size and type and number of features were increased”* (M. Ashtawy et al., 2012, p1311). Scoring functions are pretty close to the atomic level physics and chemistry, sometimes even quantum mechanics. Atomic level simulations are called molecular dynamics, *“a technique for computer simulation of complex systems, modelled at the atomic level”* (J. Meller, 2001, p1). Molecular dynamics usually used on macro molecules, because of very high complexity and need to understand how they work. *“Biologically important macromolecules and their environments are routinely studied using molecular dynamics simulations”* (J. Meller, 2001, p1). Ligand and receptor interaction can be measured by SFs, but the whole thing could also be simulated as molecular docking, which” *simulates the binding of drug molecules to their target protein molecules* (S. Smith et al., 2014, p119). Although there are non-computational *”experimental techniques, such as X-ray diffraction or nuclear magnetic resonance (NMR), allow determination of the structure and elucidation of the function of large molecules of biological interest”* (J. Meller, 2001, p1). They are not as popular as computational, because these allow some level of protein structure prediction without actually knowing it or costing as much resources. *”X-ray crystallography and Nuclear Magnetic Resonance (NMR) being expensive and not likely to cope with the growing amount of data and necessitates the development of novel computational techniques”* (M. Smitha et al., 2008, p2). Computational simulations or not, they are still very complex problems, because of how many laws and particles are involved in the process. *“Many problems of interpreting data from molecular biology experiments have combinatorial complexity”* (J. Fox et al., 1994, p294). Meaning that there are multiple related variables with each having their own different set of values. These could be solved by constraint-based system, which *“represents the dependencies among all the objects in a problem as constraints, and uses a problem solver that will prune illegal solutions and their consequents (constraint propagation) when a constraint is violated*” (J. Fox et al., 1994, p294). J. Fox et al. (1994) points the weaknesses of machine learning, but does not lose hope and says that *“AI has the potential to make a significant contribution to computational molecular biology”* (J. Fox et al., 1994, p298). Older techniques are still used but in combination with more advanced ones, which will be mentioned in further paragraphs.

### Abstracting the data to reduce the problem

The rise of big data gave rise to better results from machine learning, it allows *“to make fewer initial assumptions about the underlying model and let the data determine which model is the most appropriate”* (Y. Wang et al., 2015, p508). Because of how big the Big Data is, even the most efficient algorithms need it approximated, reduced, modelled, discretized, normalized or abstracted to some level, and that way allow taking limited amount of resources to produce the results. *“It is intuitively clear that less accurate approximations become inevitable with growing complexity”* (J. Meller, 2001, p2) and*” in order to develop efficient protein structure prediction algorithm, there is a need for restricting the conformational search space”* (M. Smitha et al., 2008, p3). Such simplification could be called simulation or *“reduction of complexities in living things for achieving much smaller systems that can express the whole or a part of living cellular properties”* (K. Tsumoto et al., 2007, p102). Mathematical abstractions arise, some of them rely on vector ideas, where each attribute of a protein is converted in to numeric value as a separate and unique vector dimension. These attributes can be called features and together represented as feature vectors. “*Much attention paid on statistical or machine learning techniques to classify the proteins using feature vector representations of available knowledge”* (A. Chinnasamy et al., 2003, p2). Noble (2003) mentions that you can make” *each protein characterized by a simple vector of letter frequencies”* (W. Noble, 2003, p5). Amino acids can be represented by letters. Proteins can also be represented as sequences of SF’s results, then “*resulting scores are used to map each protein into a 10,000-dimensional space”* (W. Noble, 2003, p6). Blaszczyk et al. (2015) described algorithm called CABS-dock which uses 3D structure of the receptor and peptide sequence to predict best fitting peptide for that receptor. 2D representations of biochemical pathways and their simulations*” can be interpreted as graphical programming languages, and compiler theory can be applied to compile/interpret the graphical language into another one that is supported by runtime simulators”* (R. Gostner et al., 2014, p16:2). Data can also be simplified by introducing limits such as *“neither the residues in the protein binding pocket nor the ligand are allowed to have organic elements other than C, N, O, P, S, F, Cl, Br, I, and H”* (M. Ashtawy et al., 2012, p1303). Even averaging can be done in some cases, because*” according to statistical mechanics, physical quantities are represented by averages over microscopic states”* (J. Meller, 2001, p4). Another way to identify protein is to compare its similarity to others, but*” it is important to keep in mind that sequence similarity does not always imply similarity in structure and vice-versa”* (J. Cohen, 2004, p131). With abstractions making the job easier for computers comes the side effect of reduced precision and*” in practice the quality of sampling and the accuracy of the interatomic potentials used in simulations are always limited”* (J. Meller, 2001, p4). Even though you have to lose precision by reducing the data for effectiveness, more powerful computers arise and less abstraction might be needed. Nevertheless with the rise of big data, precision becomes not such a big issue. *“With synchrotrons and fast computers, drug designers can visualize ligands bound to their target providing a wealth of details concerning the non-bonded interactions that control the binding process”* (V. Lounnas et al., 2013, p1).

## Techniques from biology examples

### Evolutionary algorithms

Even if evolution took so long it has been successful, that is why taking examples from nature has also been done in computer science. Some of these include evolutionary algorithms and artificial neural networks. Evolutionary algorithms use the ideas like natural selection and random mutation. *“The evolutionary algorithm is one of the major methods used to investigate protein folding”* (J. Tsay et al., 2012, p2). It is done by taking the parameters of molecule, its atoms and surrounding physics as mathematical representations, although these parameters or attributes might not necessarily be mathematical representations, because *“for every application of a genetic algorithm one has to decide the representation formalism for the genes”* (M. Smitha et al., 2008, p5). Each attribute randomly changed, then checked if new combination or fold is more successful than previous one. The process repeats for most successful candidate folds and it often does not lead to perfect solutions or might never finish searching, so it is stopped at some point and then tested. Protein folding is used to determine its functions, but there are different ways to do that. It can be compared to other known proteins, in other words, *“protein function can also be determined via sequence comparison with other species”* (W. Noble, 2003, p10), and it can also be determined by finding its 3D structure from a chain of amino acids, because *“it is the tertiary structure of a protein that determines its function”* (J. Tsay et al., 2012, p156).

### Representing proteins for evolutionary algorithm folding

Protein folding techniques may use laws of physics, chemistry or quantum mechanics. *“Physics-based folding is far from routine for general protein structure prediction of normal size proteins, mainly because of the prohibitive computing demand and the best current free-modeling results come from those which combine both knowledge-based and physics-based approaches”* (S. Wu et al., 2009, p234). Because of that, data has to be represented in abstracted and simplified ways. There has been various publications from different authors and their interpretations of how data could be abstracted.

Tsay et al. (2012) used evolutionary algorithms to predict protein structure and mentioned the use of 2D or 3D transformations as tuneable parameters for evolutionary algorithms to be randomly tweaked until best structure is found. Smith et al. (2014) used computation of each atom pair interaction energies to evaluate how well protein and ligand binds given certain pose or alignment. Representation can also be done by using *“fitness function based on a simple force field”* (M. Smitha et al., 2008, p2). Because of high computation needed, advanced optimization or minimal use of computer resources has been also considered, Smith et al. (2014) used genetic algorithms to determine best alignment of protein and ligand and tried to optimize it to the point where each atom description fits just in to 16 bytes, then compared different hardware computational power. If we consider ASCII character representation format, it would mean that each atom was represented by two short words, which is impressive, given that each atom has multiple properties like its mass, proton and electron count. 16 bytes is quite common size of memory addresses, it aligns *“much more efficiently with most hardware’s memory interfaces”* (S. Smith et al., 2014, p124).

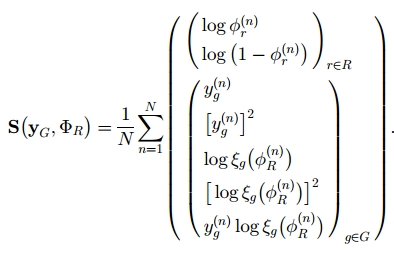
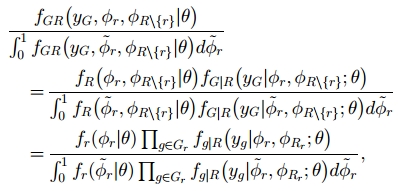
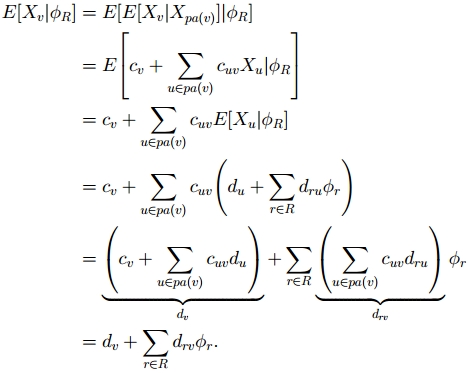
### Artificial neural networks

Another example taken from nature is the brain and its ability to recognize patterns, which for long time seemed like very complicated process until someone thought of our sensory data as inputs, brain as machine made from many repeated bits called neurons and the result of recognition as output. That machine takes inputs and is taught that these inputs represent certain value or meaning and when new similar situation happens, it can be recognized. Each repeated bit would be neuron and each neuron also has minimized version of input, machine and output, which are interconnected with others. How neuron learns relies on remembering previous input and comparing how different it is, then strengthening or weakening connections which interact with most informative data This new neural net then is used to evaluate how new data values are similar to strongest connections. Interconnections are mixed and crossed, inputs are summed and then ran through functions, which lead to some problems, because*” what they learn is difficult to understand”* (J. Fox et al., 1994, p292) and *“make many non-computational experts very wary and distrustful of the results”* (J. Fox et al., 1994, p292). But” *neural networks can learn very complicated relationships between their inputs and outputs”.* (N. Srivastava et al., 2014, p1929) and*” deep neural nets with a large number of parameters are very powerful machine learning system”.* (N. Srivastava et al., 2014, p1929). Ding et al. (2000) mentions that neural network accuracy drops significantly because of higher noise levels. Noise is a big problem to most machine learning algorithms, because it can skew the results towards some bad value. That is why noise reduction methods are usually added to the main machine learning algorithms.

## Later techniques

### Bayes Theory Probability Based Networks

From statistics we know that probability is a useful tool used for prediction, but it may become difficult when dealing with multiple related probabilities. Bayes theorem helped to solve this problem with famous equation:””. It takes in to account the probability of something that already happened and is related to something that might happen. This can be used as cascading tree of probabilities given conditions and treating each group of probabilities in a context of probability of A given the probability of the rest, until there is only one left. That way you can use it in machine learning on multidimensional data. Yoruk et al. (2011) used Bayesian networks to model cell protein signalling and said that*” biological data are inherently probabilistic and generally display hierarchical relationships”* (E. Yoruk et al., 2011, p592). You don’t usually have hundreds of thousands experiments on some biological target, which means that the amount of data for each unique event is not very large. Many machine learning algorithms need a lot of data for the accuracy to be higher. *”Bayesian neural nets are extremely useful at drug discovery, but are slow to train and difficult to scale to very large network sizes”* (N. Srivastava et al., 2014, p1941). But even if they are slow, they can be more precise because of the use of inter-related probabilities and because it is not very difficult to pre-calculate probabilities for smaller data sets. *”Bayesian classifier requires a prior knowledge of many probabilities”* (A. Chinnasamy et al., 2003, p2). This classifier is also more popular among biologists because the results are clear and easy to understand, slow training is not a problem when data sets are small and” *the biologists need to know the confident level of the resultant classes outputted by the classifiers for further analysis”* (A. Chinnasamy et al., 2003, p11). Chinnasamy et al. (2003) managed to produce better results with Bayesian networks than SVM. Even though the Bayes theorem seem simple, when used with multivariate attributes and many related probabilities, it may start to look like in the pictures below, making it not intuitive to understand, but to make it easier you could think of them simply as nested “P(A\The rest)”.



1

2

3

Screenshots 1, 2, 3: Possible Bayesian Network Formulas: (Source: E. Yoruk et al., 2011, p604)

### Geometry based Support Vector Machine

The way machine learning is used could be explained by saying that algorithms use data identified or labelled by experts to learn or find best separation point between different identities or class labels, so that when new unidentified or unlabelled data comes in, machine could tell on which side of separated space it is and therefore which label or identity that data has. There are many different algorithms which separate data in to various ways, but unlike SVM, these do not try to find the best way. The best separation in SVM is called widest margin. It leads to higher precision and that’s why *“support vector machine learning algorithm has been extensively applied within the field of computational biology”* (W. Noble, 2003, p1). SVM starts with very simple ideas from 2D/ 3D vector geometry. If we imagine data rows with 4 variables as points or vectors in 3D space and fourth value being the class label with one of two possible labels or colors, then we could draw a plane in between differently colored points, finally, try to modify the plane’s location and orientation so that it is as far away from each color as possible. When new uncoloured points come in, we can determine what color they should be, by measuring on which side of the plane they are, and because we modified the plane to be as far away as possible from each color, even the points on the very edge, difficult to identify, could simply be measured by how far and in which direction of the plane they are. “*There are three key ideas needed to understand SVM: maximizing margins, the dual formulation, and kernels”* (K. Bennett et al., 2000, p1). One of SVM problems is overfitting, but you can improve it by reducing *“the input dimensionality”* (N. Srivastava et al., 2014, p1942).

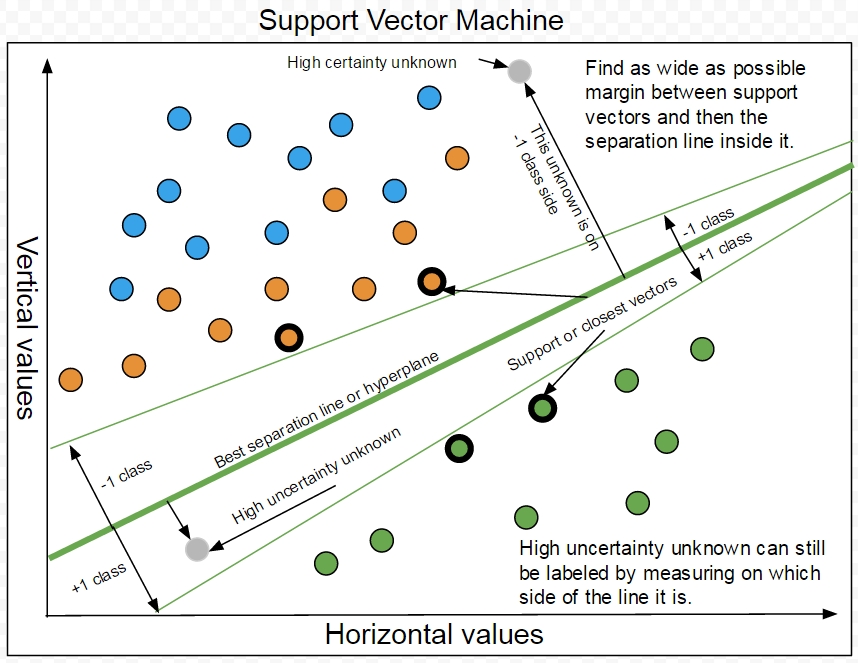


Figure 1. SVM and hyperplane separation (Source: Gintaras Koncevicius, 2016)

Data is rarely 2D or 3D, it is usually multi-dimensional, multi-class, noisy and non-linearly separable, which means that the different class label or color points are not separable by simply drawing a line or a plane. Instead it could be separated by some curve or curved hyper-surface in some dimension. Our data rows can be called as vectors or, as in some formulas, transposed vectors which are one row matrices. In higher dimensions the hyperplane is not anymore a nice hyper-flat geometrical thing, it is converted in to a formula or a function, which is created by measuring distances between vectors, taking in to account each vector’s color or class label. In SVM the geometric functions are transformed in to Lagrange multipliers, which needs two parts, main function determining the rule or the hyperplane and constraints relying on support vectors, which help determine how hyperplane is oriented. Lagrange multipliers allow finding the minimum or maximum values, and in this case, the function which represents furthest hyperplane from each class. In non-linearly separable case, function to measure the distance between points might not be the same as in linearly separable. In SVM distance formulas are called kernels. Kernels allow linear separation by plotting points in to higher dimensions. “*SVM kernel framework accommodates in a straightforward fashion many different types of data—vectors, strings, trees, graphs”* (W. Noble, 2003, p24). Higher dimension kernels could simply be called measurement s of how similar are the two vectors.

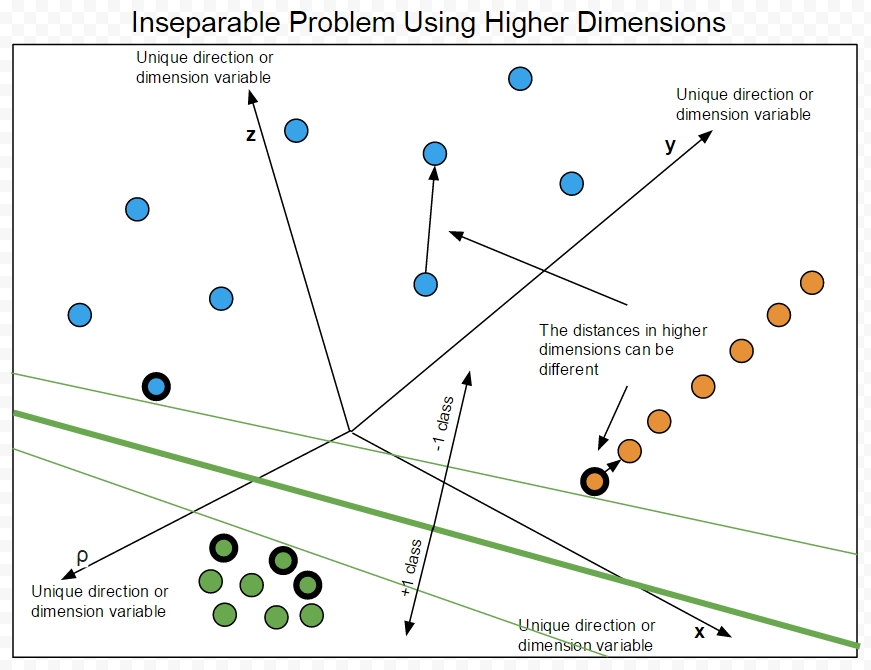


Figure 2. Higher dimension separation (Source: Gintaras Koncevicius, 2016)

Finding the hyperplane could be called searching for the best formula which uses the best vectors or support vectors to determine its constants, so that when new unknown color or label vector comes in, formula outputs a number, which tells the color or class label of that vector. Support vectors in SVM are vectors which are closest to the opposite class. Because of the noise, similarity formulas might not need to be very strict, allowing to avoid noise determining the value. In SVM it is called slack variable and can also be used to separate non-linearly separable cases, by reducing precision. Slack variable is simply a value which makes comparison less precise or restrictive to allow some noise tolerance.

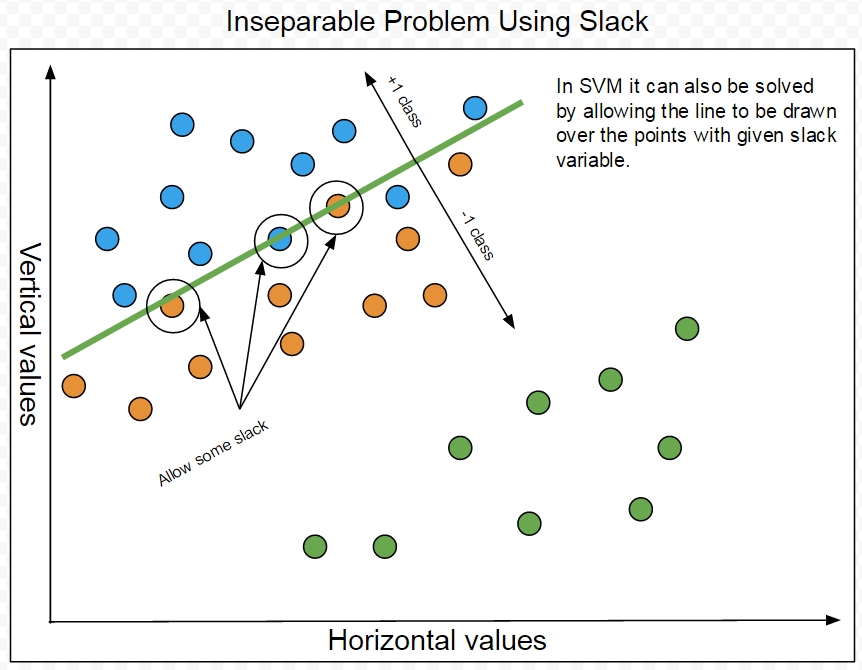


Figure 3. Slack variable (Source: Gintaras Koncevicius, 2016)

Multi class solution is called “one vs the rest”, when positive class is marked as +1 and the other classes as -1, then repeated multiple times by taking away already found positive class. The *“most common means of training an SVM for an n-class problem is the one-vs-others method”* (W. Noble, 2003, p5).

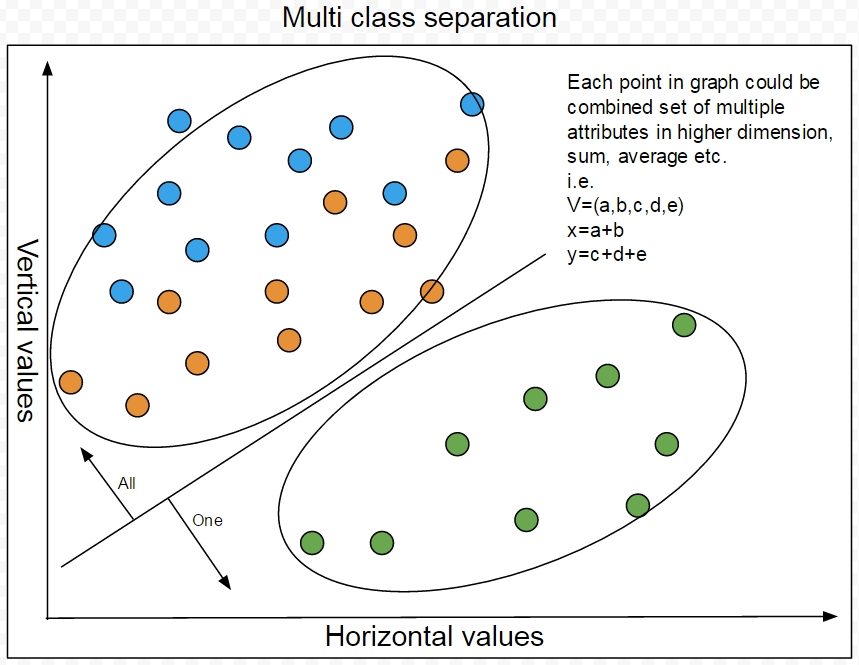


Figure 4. One vs. others/rest/all (Source: Gintaras Koncevicius, 2016)

In SVM the Lagrange multiplier formula is further transformed and simplified for the easier use different kernels and slack variables. This formula is then solved by software called quadratic optimizers. At this point SVM idea or problem becomes mathematical optimization problem with multiple unknowns of one type, called alphas and constraints, which define the rules for where and how oriented the hyperplane, function or a matrix, should be. The answer then can be used to determine the class label of new unlabelled vectors. Programmer who wants to write SVM, would either have to use existing QP solver software to be able to solve SVM problem, or know how to write software which solves QP problems, which requires advanced math knowledge. In the following tables author shows some of the transformations of Lagrange to SVM formula.

|  |  |
| --- | --- |
| Line equation: | Line or hyperplane can also be described as |
|  |  |
| Or | This hyperplane is the center hyperplane. |
|  | To find its best orientation you need 2 more parallel hyperplanes with slightly different equalities. |
| Imagine it is made from two vectors | Positive class: |
|  |  |
| Then it can be described as dot product | Negative class: |
|  |  |
| Modified for easy use hyperplane function for Lagrange multiplier: | Function defining constraints of the hyperplane, its shift or distance from origin or zero, value , associated class labels and support vectors |
|  |  |

Lambda is an unknown variable which helps finding the unknowns for the rest of the functions. When you have multiple constraints , you have to use the sum of them.

|  |  |
| --- | --- |
| Lagrange multiplier formula. | Multi-constraint |
|  |  |

|  |
| --- |
| Plugged in formulas. |
|  |

|  |
| --- |
| Extract the -1 |
|  |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |
| **Produce partial derivatives to simplify formula:** |
| With respect to |
|  |
|  |
|  |
| With respect to b |
|  |
|  |
|  |
| **Plug the derivatives back in** |
|  |
|  |
| **Possible further reconfigurations** |
|  |
|  |
|  |
|  |

Further simplification leads to SVM formula which allow easy use of kernels and slack variables.

|  |  |
| --- | --- |
|  |  |
| Dot product distance or similarity measurement formula. | Kernel looks version |
|  |  |
| Constraints from derivatives |  |
|  |  |

You could replace with any formula or kernel which measures distance between vectors in various dimensions. The big formula now has same amount of as vectors. You could view s as the hyperplane which separates vectors of class labels by using K kernel to measure distances. You have to find best s which satisfy the constraints and . It is difficult to find so many unknowns which rely on known variables and constraints. That is why Lagrange multipliers are used and that is why its formula is converted in to quadratic optimization problem for QP solver software. QP software usually accepts these formulas as matrix of vector values and matrix of constraints represented as constants.

### SVM results by various authors

Studies have been done comparing multiple machine learning techniques with biological data and SVM has been shown to be intuitive to use high precision algorithm. Chinnasamy et al. (2003) talked about other studies where SVM performs better, but has a high number of false positives, he said that*” from all these studies it is evident that among all the prediction methods, SVM performs better* (A. Chinnasamy et al., 2003, p3) and explained that *“in SVM, as the number of classifiers is high, reading the distances between hyperplane and the classes are very difficult”* (A. Chinnasamy et al., 2003, p11). Noble (2003) said that SVM can be used*” to learn to differentiate true from false positives”* (W. Noble, 2003, p23). Furlanello et al. (2005) used SVM for molecular profiling, outlier detection and classification of samples to reduce computation required. Ding et al. (2000) compared SVM and neural networks when predicting protein fold and the results indicate much higher accuracy and faster processing with SVM than NN. SVM is one of the most precise techniques and it is*” very robust methodology for inference with minimal parameter choice"* (K. Bennett et al., 2000, p1) and that is because you just have to choose the Kernel and slack variable value. Choosing correct kernel might not be easy, because it is not always clear what dimension or measurement of similarity is the most appropriate for the data.

# Presentation

## The problem

Discovering drugs still involves a lot of real experiments on live cells. Automated and simulated models of cells could allow speeding up this process. Simulations require designing models, which represent real structures to predict their behaviour without doing real experiments and requiring more resources.

If we consider a cell as an object which we are trying to simulate as precisely as possible, we might need to include physics and chemistry and even quantum mechanics of each atom. Full simulation even of just one cell is still beyond what computers can do effectively today and that’s why biology, physics, chemistry and computing experts try to find the best balance of how far the problem can be abstracted, but still keep predictions in satisfactory precision and still return results within resource constraints. Simulating just certain function would reduce the problem, but could lose precision, because of undiscovered relations to other functions. There are some examples of quite abstract solutions in machine learning, like neural networks, which simulate brain function of recognizing patterns, without the need to simulate brains to the atomic level and even taking more space than just couple paragraphs of code. But even if this example lets us know that it is possible to make quite big abstractions to work, in drug development it might not be the case. Drugs are also made of molecules, which have to be exactly defined to be able to produce them. That causes some difficulties when trying to abstract molecular processes to get answers about precise molecules. It is difficult to abstract something and get out precise answers, but it is possible to increase precision by gradually abstracting less and less.

“*Computational modeling of the structure of protein-peptide interactions is usually divided into two stages: prediction of the binding site at a protein receptor surface, and then docking (and modeling) the peptide structure into the known binding site”* (M. Blaszczyk et al., 2015, p1). The *“gap between identified hits and the many criteria that must be fulfilled, can be bridged by investigating the interactions between the ligands and their receptors“(*V. Lounnas et al., 2013, p1). Finding the point of abstraction at which simulated cell or cell’s function behaves acceptably close to reality is another interesting and complicated problem. Experimenting at different levels of abstraction could allow finding the optimal solution for further development of these simulations. *“With faster and more powerful computers larger and more complex systems may be explored using computer modelling or computer simulations”* (J. Meller, 2001, p1). Work with cell receptor and ligand interactions have examples both in real biochemistry experiments and machine learning. The most unambiguous and precise language for modelling is mathematics and *“mathematical modelling typically requires deep mathematical or computing knowledge, and this limits the spread of modelling tools among biologists”* (R. Gostner et al., 2014, p16).

The knowledge gap between experts and beginners keeps increasing, because their time on actual research is more valuable than time spent explaining their work. Advancing science caused more narrow fields to arise and as a result it keeps increasing the knowledge gap even between experts. The need for experts and guides on their created technologies arise and it could be solved by gradually educating about the general topic or parts of it and its problems. There are experts who are doing actual experiments and search for real solutions, but there are not that many intermediate tools or solutions which introduce to the bioinformatics by showing how it works and what are its tools.

## Proposed solution

Partial solution could be to advance research about these kind of problems and educate more people about bioinformatics. Proposed solution offers to educate about the main topics of bioinformatics and one of the best machine learning techniques- SVM together with interpreted implementation of it in Java.

## Analysis and Specification

Given that SVM math is difficult to fully implement from scratch, because it uses quadratic optimizers, it was assumed that to create its interpreted version of code, idea, math and logic would be easier to understand. The solution to this difficulty was called plan B and supposed to be used in case author does not manage to implement SVM in Java. It lead to discovery of different interpretation of SVM, which looks easier to understand even at programming the math level.

### The interpretation and Least Similar Sphere

If we look at SVM hyperplane in 2D, it is a line, which separates two sides. We can imagine that a line can be two parallel lines very close to each other. These lines could be just edges of very large circles, which means that opposite centers have to be very far away from each other. In SVM we use support vectors or closest opposite vectors to support the finding of separating hyperplane, but in interpretation these vectors are used to find anti-support or furthest vectors. In geometry we can measure distances with various formulas, which in SVM are called kernels. The distance between two different vectors is a value. That value can have slightly different name than a measurement of distance, it could be called measurement of how similar or different are the vectors. Author uses these names to further simplify mathematical problem in to more intuitive logical one. That way multidimensional vector space, distances, hyperplanes, hyperspheres can become easier to imagine. Another difficult problem might be to try to imagine hyperplane separating high dimensional space. In previous paragraphs it was mentioned that this high dimensional hyperplane becomes a complex formula with no easy way to imagine its shape. That is where another simplification might help- spheres and hyperspheres. If a circle is simply a set of points which surround a center point always in the same distance, just different direction, so is the sphere. The sphere just has one more dimension, but the idea of the same distance to all directions from center is the same.

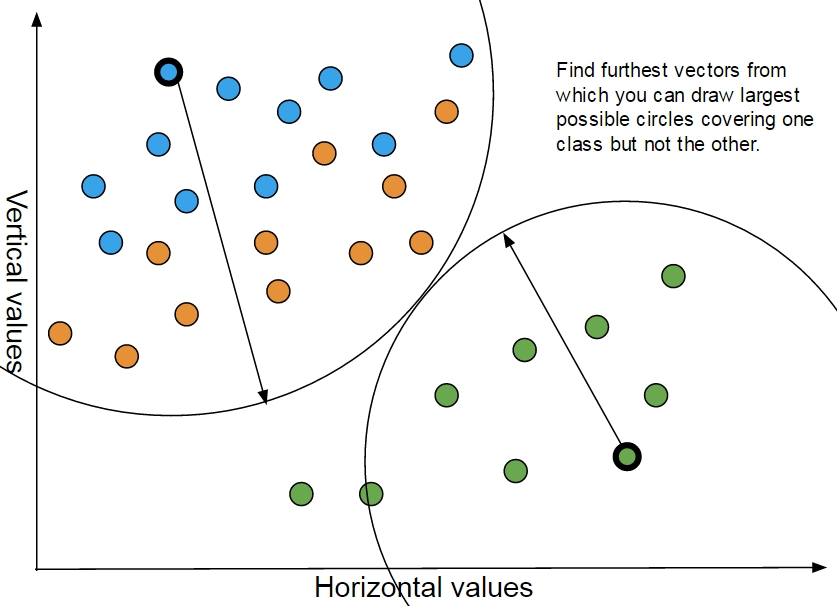


Figure 5. Least Similar Sphere LSS or SVM interpretation (Source: Gintaras Koncevicius, 2016)

At first it might seem difficult to imagine hypersphere, but if you look at it just as picking the point in high dimension space and spray the paint same distance in all directions of that space, then that would seem like a sphere. Now if we join all ideas it starts to sound like finding least similar vectors of opposite classes and drawing spheres up to the most similar opposite vectors and then using these spheres to determine how different or how close the new unknown vectors are from these spheres. That is how Least Similar Sphere idea arisen. Author hopes that someone else, somewhere else, some time ago did not already invent this idea and claims that given limited amount of knowledge about the topic and desire to invent, tried to independently and uniquely create something new and unify existing ideas learned from SVM, ANN and clustering.

## Methodology

### Programming Language

Questions arise about what type of language to use for implementation of Least Similar Sphere or LSS when knowing about *”AI programming languages such as Lisp, Prolog”* (J. Fox et al., 1994, p288) and more mathematical MatLab. Because these languages focus on effectiveness and optimal solutions, these usually result in very difficult to read functional or procedural programming code. Author focuses on educating about machine learning and therefore needs the code to be easy to read and understand. That leads to the choice between object oriented languages like C++ and Java. Java was chosen because of previous author’s experience with it and lack of time for learning C++.

### Data for machine

Since the main topic is about ligand and receptor interaction or bioassay experiments, the data chosen had to be as relevant as possible. Searching online lead to finding one large bioassay dataset by A. Schierz (2009). Could not find more possibly because most of the data is still used in research and is not easily available as a bunch. Further attempts to gain more biological data led to discussions with specialists in this area, but led to conclusion that author has to collect his own data manually from databases like PubMed using protein docking software like Pyrx. Author attempted to use Pyrx, but had bad results and decided to go with plan B for dataset. That meant searching for few proteins or receptors on PubMed, which have a lot known ligands that bind to them. After few hundred compounds reviewed, author found proteins with up to 10 known ligands, but did not use all of the ligands as data, because some of them were just common ions like magnesium. The protein id was then chosen to be a class label, while different unique ligands were represented as vectors. Author focus on idea of information theory and interprets ligand molecular formula as information which already contains its 3D structure without being folded. From author’s point of view, if protein can fold from a sequence, then that sequence contains information how it should fold and therefore indirectly describes its 3D structure. That led author to naïve way of representing molecules using their mass in g/mol, hydrogen count, carbon count, carbon count upwards, carbon count downwards, oxygen count, oxygen bond counts, oxygen count upwards, oxygen count downwards, nitrogen counts, nitrogen bond counts, nitrogen count upwards, nitrogen count downwards, sulphur count, sulphur bonds, phosphorus count, phosphorus bond count, which in the dataset are named as *“MassGmol, HCount, CCount, CUp, CDown, OCount, OBonds, OUp, ODown, NCount, NBonds, NUp, NDown, SCount, SBonds, PCount, PBonds, BondsToProtein”*. Given that these biological datasets might not necessarily produce results with good description of how well the machine works, another dataset of flower Iris by Marshall M. (2016) was used to further test for precision of the machine.

### Prediction, results and comparison

While the machine could produce some results, they might be difficult to evaluate without comparing it to something. That is why author chose popular open source machine learning software like Weka and R to compare how real SVM works with collected data and how it compares to interpreted implementation.

## Trustworthiness

Given that machine implementation does not deal with private data and is not critical software, web implementation will be used to explain all facets of trustworthiness. The reliability is partially solved in machine learning implementation by using Java exception handling, which prevents some events, like file not found, from crashing whole software and allowing user to re-enter another file path. The safety, reliability, availability, resilience and security facets for web implementation are further described in Appendix C together with its design and implementation.

## Design

LSS machine design requires data, which should be loaded in to memory as lists of feature vectors with their class labels. It needs two types of data, one with assigned class labels for learning phase, another with unknown class labels to predict them. Then for each class label there should be a pair of lists representing each “one to all”. To evaluate prediction precision it should use same data for learning as for predicting, but reset class labels to unknowns and then predict them. Once it is done predicting it can then compare it with learning data. The learning phase needs to find closest vectors to opposite class and then use these to find the furthest vectors. Predicting phase has to use these vectors and calculate distances which represent each class multiple radius spheres of each class and then check if new unknown value is within the radius. The inseparable problem is then attempted to be solved by using different formulas for measuring the distances and using a slack value to reduce or increase each radius size. LSS could also try to learn and predict multiple times to find best prediction, by changing the settings. Additional ideas like creating the averaged vector clusters, finding and removing outliers could be used to try to improve prediction precision. The outlier could be furthest vector which is far from its closest vectors but close to opposite class closest vectors. The averaging could allow finding smaller averaged cluster which has higher abstraction than original one, but still has information describing it. The formula for finding closest vectors could be different from finding furthest and different from the prediction formula. Its success is mentioned further. The design of LSS idea will be shown in following pictures and comments.

### Learning phase, finding furthest and closest vectors.

Once closest points are found, by measuring each vector of one class and checking how far away from other class it is, then choosing closest one, these points are then used to find furthest ones.

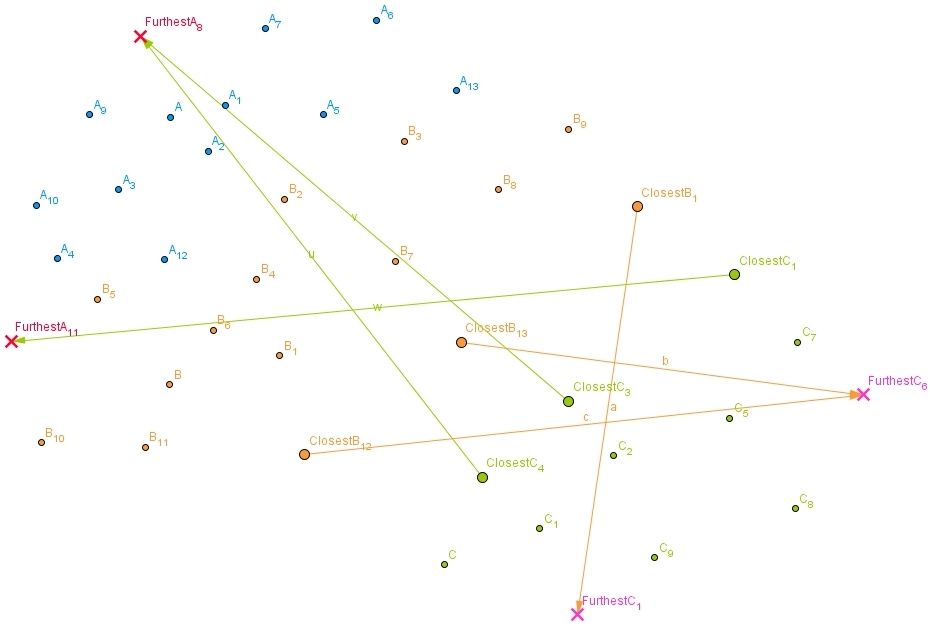


Figure 6. Finding closest and furthest vectors. (Source: Gintaras Koncevicius, 2016)

### Predict phase

For each “one to all” class label feature vector pairs iterate through furthest vectors. For each furthest vector find closest opposite class vector from already found closest ones. Find closest opposite class vector, which this time is its own closest vector. Find mid-point and then calculate the distance from furthest vector to mid-point, then use this distance as sphere radius which represents this class and check if unknown vector distance from this furthest vector is less or equal to that radius, indicating that it is inside or outside this class. In case closest vectors themselves are out of range of the radius, they also have their own radius. It is calculated by finding closest opposite class vector and then using radius to midpoint between them.

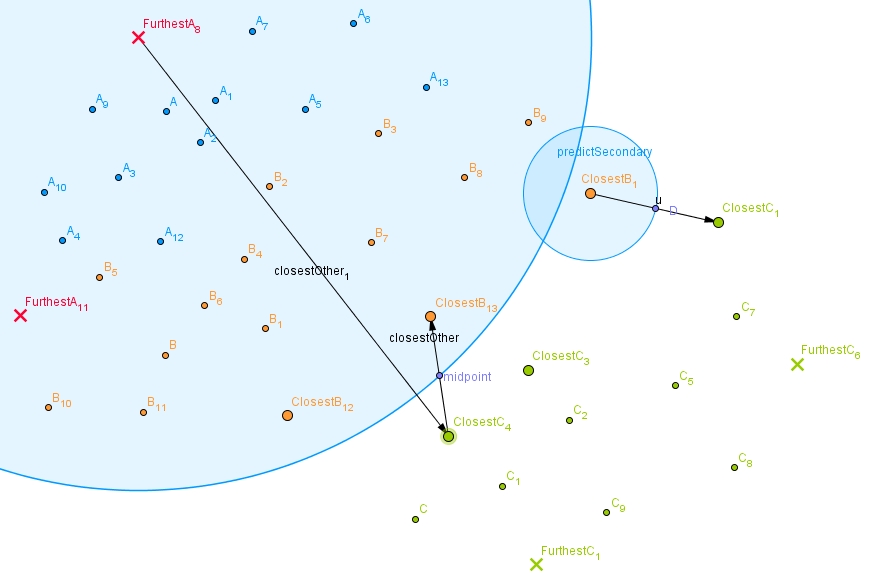


Figure 7. How spheres and their radiuses are built. (Source: Gintaras Koncevicius, 2016)

### The whole picture of LSS

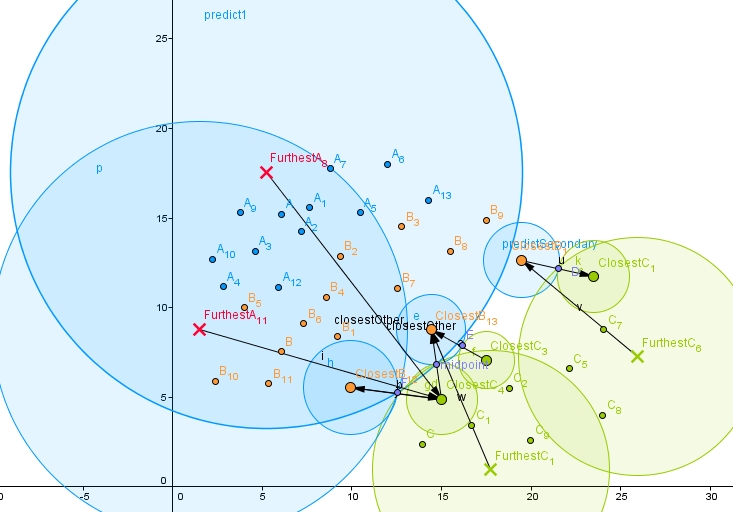


Figure 8. How LSS predicts. (Source: Gintaras Koncevicius, 2016)

## Implementation

The implementation has been done in Java and expanded to have graphical user interface for easier use. Although the design section describes working version of learn and predict functions, these have been discovered through trial and error while trying to write implementation and trying to find a best way to use the correct vectors to represent closest and furthest, together with finding correct prediction radius calculation. The problems ranged from furthest vectors found being too close to opposite class, because of how the search method caused the prediction to be too low or too high. These issues indicated that maybe LSS just knows the amounts of each class label and predicts them from top to bottom. There were problems in finding the correct technique of calculating the furthest vector sphere radius, because it either included opposite class or did not include enough vectors of its own class. It was also solved by drawing various diagrams and by introducing hash maps and shuffling. To improve fairness of calculating the prediction precision, learn and predict data are shuffled and corresponding vectors are found using hash maps. In further code snippets and screenshots it is explained how main idea of the LSS design was implemented.

The main machine is “FastMachine”. It only prepares “one to all” class pairs, finds closest and furthest vectors, and then uses them to predict unlabelled data. “SlowMachine” extends “FastMachine” and allows using it multiple times to find best settings resulting in best prediction.

### Hash map use

Learn data is put in to hash map so that later it could be used to calculate prediction precision when all the data is shuffled.



### Multi class solution

One to all label pairs are created to allow multiclass solution. More effective solution could use the list of hash codes instead of whole copies of data, but has been left for future improvements.Each label pair contains the data which separates them in to one and all pairs for each class label.



### Learning

It finds closest and furthest vectors for each label pair.



Find closest and furthest uses find with Boolean which sets what to find.



It uses settings which contain the pointers to formulas which measure the distances to try and solve inseparable problem.



### Predicting

Various modifications have been done to prevent high precision because of how data is sorted and how class labels are counted. Shuffling and hash maps gave the most reliable results.



For each label pair, check which class the unknown value might be.



The “assignLabels” method iterates through furthest vectors and measures radiuses in the way it is described in the design section.



The “inSphere” method checks if new unknown value is in the class representing sphere. It iterates through each furthest vector, then finds closest opposite class vector. Using this vector it finds closest opposite class vector again, but this time it is its own closest vector. These are used to find midpoint and then radius which represents class points inside the radius. The next iteration checks each closest vector and their own circle radius in case they are not in range of furthest vector radius.



Inseparable problem solution by using slack variable can be seen inside the above code snippet as “safeDist\*=currentSettings.getPredictionSearchSensitivity()”. It modifies the sphere radius by multiplying it with given value.

### Slow Machine

To improve learning the option to search for best learning subset was also implemented. It randomly creates subsets of learn data, learns it, predicts, reset original data to unknowns, then compares the prediction precision and tracks which was the best subset. Due to difficulties and complexity for calculating precision when using subsets, these were not used for results even though it had better results. One of those difficulties arise when subset becomes vectors of one class label, which results to 100% precision because there is no other class.



Find best settings and find best closest similarity settings use randomly generated settings and “FastMachine” multiple times by comparing the prediction precision.

## Testing

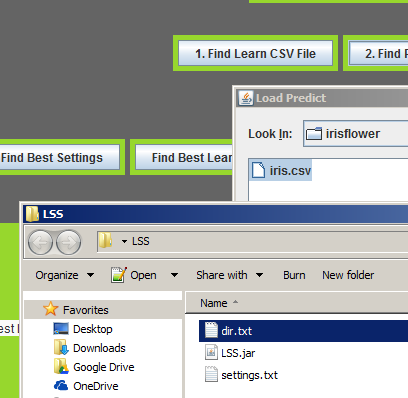
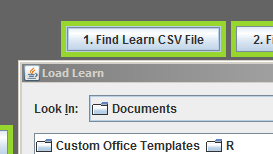
LSS machine accepts CSV format and can be used through simple GUI created for it. It has few additional features for easier use. Once you load file, it remembers last location you used and allows either loading previous files or open previous directory. It can also save settings, so that next time you wouldn’t need to search for them. It has 2 main phases- learning and predicting. If the error or exception is caught, its output can be seen in the information text area. Further testing of GUI and machine will be explained in the following screenshots and comments.

### Possible functions

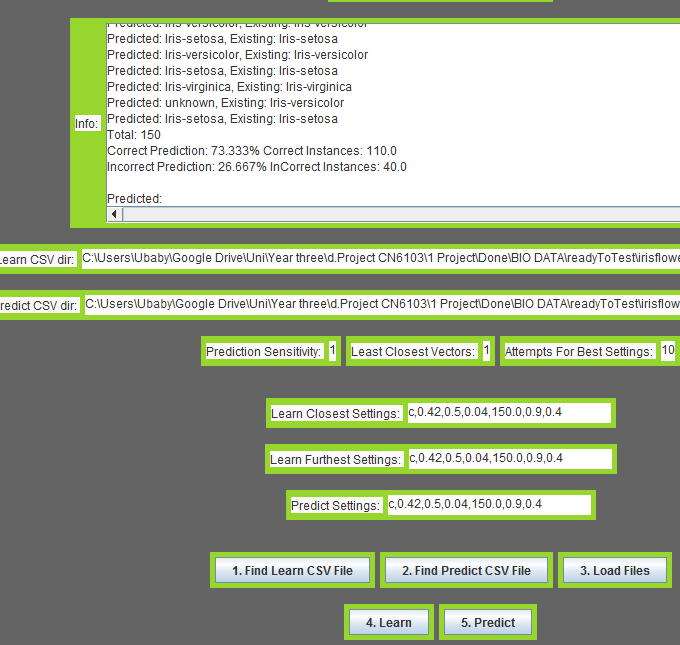
The launcher is in jar format, it needs Java 7/ 8 installed on your computer.

C:\Users\Ubaby\Desktop\1.JPG

The buttons are numbered to allow easier understanding on what to click first. Once you press find learn csv file, it opens documents folder on first run, but next time it opens your previous folder. It creates a file which is used to remember your used directories.



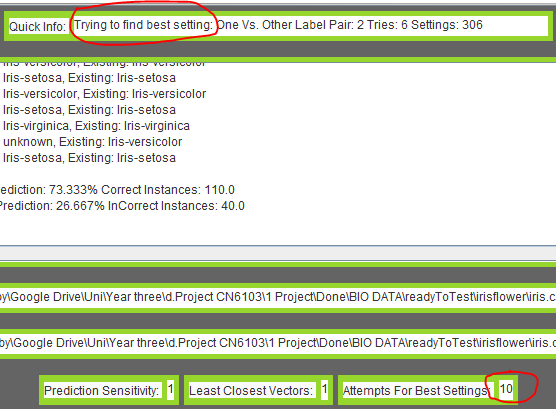
Once files are found, their paths will be shown in text areas, which can be edited for easier loading other similar files without searching for them. If the choice has been successful, then pressing load files, loads them in to memory and make them ready for learning. After pressing learn, last option is to click predict. The results can be seen in the top info text area.



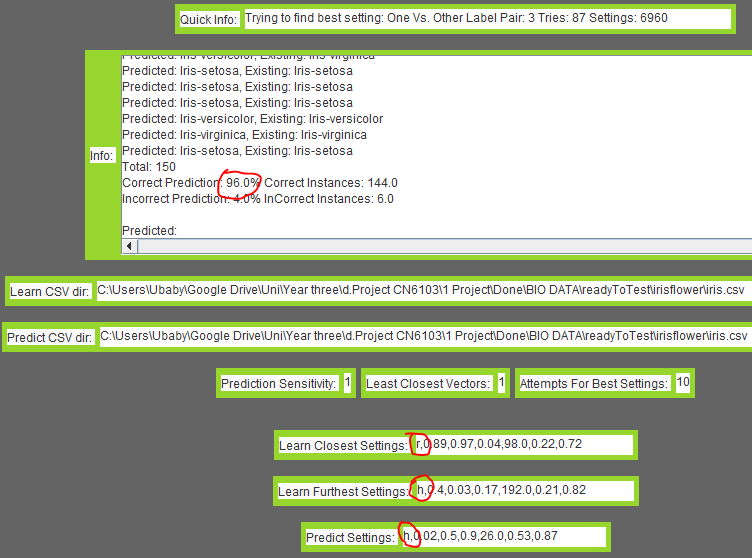
If the result is not acceptable, there are few more options. Find best settings run the machine multiple times guessing the best settings for highest precision. As seen in screenshot, the default settings are Euclidean distance measurement formula and prediction sensitivity or slack multiplier 1. The formula titles are their first letter. Available formulas for distance or similarity measurements can be seen in the following code snippet.



Finding best settings progress can be seen in the top, it can have custom amount of tries, but the default is 10. It searches for each different setting to find closest vector, try different setting to find furthest vector, then try to predict using different setting and measure the prediction precision.

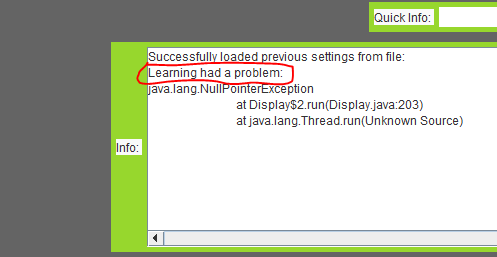


Once new best settings are found, they potentially have chance to have higher prediction precision and if not, various mix of trying to find best settings and prediction sensitivity can be used to improve results. And later, by trying to find the best learning subset. The new settings can be seen in settings text area and can be saved. Next time LSS is launched it will remember previous files and previous settings.



### Possible errors

One of the examples when trying to learn with files not loaded.



### Possible bugs

Because vectors are loaded and then shuffled multiple times through learning and predicting phases, the same data loaded at different times can produce different predictions.

## Results

LSS machine has been tested on various data and compared with real SVM machines in Weka and R. LSS used various distance measurement formulas, while Weka and R were left with default settings. How it was tested in R can be seen in the code snipped below.



### Prediction precision table of LSS, Weka SVM and R SVM on various biological data.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Type** | **Name** | **Size KB** | **LSS** | **R-SVM-RBF** | **Weka-SVM-RBF** |
| Atomic data | Ligand Protein | 2 | 69.44 | 69.44 | 91.66 |
| Flower | Iris | 5 | 95.33 | 97.33 | 98.66 |
| Bioassay | AID439red\_train | 14 | 82.14 | 82.14 | 94.64 |
| Bioassay | AID721red\_train | 18 | 100.00 | 82.89 | 94.73 |
| Bioassay | AID644red\_train | 38 | 92.72 | 75.15 | 93.33 |
| Bioassay | AID1284red\_train | 67 | 97.58 | 84.48 | 93.10 |
| Bioassay | AID1608red\_train | 168 | 98.18 | 93.35 | 96.61 |
| Bioassay | AID1608Morered\_train | 193 | 98.30 | 93.35 | 96.61 |
| Bioassay | AID362red\_train | 724 | 100 | 98.60 | 98.71 |
| Bioassay | AID456red\_train | 1627 | 100 | 99.72 | 99.72 |
| Bioassay | AID688red\_train | 4428 | 100 | 99.09 | 99.08 |
| Bioassay | AID687red\_train | 6025 | 100 | 99.71 | 99.71 |
| Bioassay | AID604AID644\_AllRed\_train | 7506 | 100 | 99.89 | 99.88 |
| Bioassay | AID604red\_train | 10893 | 100 | 99.64 | 99.64 |
| Bioassay | AID746red\_train | 10893 | 100 | 99.39 | 99.38 |
| Bioassay | AID373red\_train | 11308 | 100 | 99.90 | 99.89 |
| **Average** |  | **3369.3125** | **95.85563** | **92.13** | **97.21** |

### Precision increase in used algorithms when data size gets bigger.

# Evaluation

Given that this project background research scope is not very large, the assumption that there is knowledge gap between experts and beginners in machine learning and biotechnology might not be as true as proposed, because there are already plenty of books and journals to read about the topic and if desire to learn is high, then lack of knowledge is not a big problem, because the gaps can be filled by carefully reading these books. Although more intermediate guides to general topic allow shorter introductions and might bring more interest in the topic. While original idea was to implement existing machine learning technique- SVM in java, so that it could be easier understood at programming level, the discovery of distinct interpretation arisen together with unexpectedly good results. The failure to implement SVM in Java might have led to successful and new way of explaining it. From another point of view, implementing something what already exists is a waste of time, but this project focus is to educate about the problem and explain it with simple examples. From author’s perspective, different views and interpretations about problems can at least be useful by expanding on the ways we look at them and sometimes even help discover the solutions.

As seen in results section, Weka-SVM had best results on all datasets. Surprisingly LSS had better overall results than R-SVM, but that might not be because it works. On the other hand, LSS had 2 exactly the same predictions on same data as R-SVM which hopefully indicates that it actually works. Looking back at the design of diagrams and ideas proposed for LSS, these seem to show they should work, but because of how each time data is loaded and differently predicted it might indicate that LSS implementation in Java is probably not learning independently from how data is sorted.

# Conclusion

The report started with basic biology and machine learning introduction, then further expanded on machine learning techniques and tools used for them in biotechnology. Various papers and their results with different algorithms have been mentioned. In most of these papers SVM results were the best. Further explanation of SVM gradually reduced to author’s interpretation, then tested with bio data and compared with already existing SVM implementations in Weka and R. The results were unexpectedly good, but unclear exactly why. Either idea is good or implementation is buggy and diagrams would not work as expected in higher dimensions.

One of the biggest problems when discovering drugs is the lack of time. Some of it could be compensated if more people could work on it. More people could work on it if they understood it. Proposed solution was to educate about biotechnology and machine learning. One of the ways to do it is by introducing to the topic, guiding through main ideas, showing examples and possible interpretations of already existing solutions. Maybe in future interpretations of previous work could become full solutions for unsolved problems.

Because LSS results were unexpectedly good, possibly because of how data is loaded and how learning depends on the way it is sorted, different implementation of LSS idea could produce different results.

From the effectiveness point of view it could be improved by being re-written in more suitable machine learning language, use hash maps of hash codes instead of full copies of array lists and some for loops. LSS could be improved if it could run in command line interface. From the easier understanding, the code could be further simplified by making methods more descriptive and have more comments, it could also be written in JavaScript for it to be more web based and simpler to learn, because it is the language which is known even by web developers who don’t know a lot about programming.

Another point or discovery of LSS was to show that mathematical ideas could be interpreted differently, while still being used as they are. Some of the examples in LSS could be distance measure renamed to similarity, support vectors renamed to closest and spheres with radiuses instead of separated spaces with hyperplanes.

Literature review shown that SVM is one of the best techniques, especially when datasets are not very large, but it also shown that biologists still want higher precision and therefore even the best algorithms seem to be not as useful at first sight. Even though the precision might be as low as 80-90%, it can still help reduce the research space of, worth doing experiment for, candidates. The reduction of search space could further be improved by using it as data and some of its attributes could become how precise the machine was compared to real experiment, how useful was the reduced search space. Reduced search space would mean that at the beginning scientist had thousands of experiments to do, but he used previously done experiments with known results to train the machine and then machine can tell the scientist which of these thousand experiments are worth investigating, later he can pick random samples and test how useful the machine suggestion was.

From implementation and explanation of SVM interpretation it could be concluded that it does not really matter whether the data is biological or not. All the algorithm might care about is just how big, how noisy, how complex or simple is the data, but not where it came from or what its meaning is for those who use it.

The purpose of this project is to educate about how machine learning works and how it could help in biotechnology. It could be further applied even as introductory online guide. The purpose of the LSS could become more than educational tool about how machine learning techniques could be interpreted and used, because it maybe has the potential to become useful machine learning technique.

References

Altschul, F. (2011). Global and Local Sequence Alignment. *National Center for Biotechnology Information,* pp.1-35. Available at: https://www.cs.umd.edu/class/fall2011/cmsc858s/Alignment.pdf [Accessed 12 Oct. 2015].

Ashtawy, H. and Mahapatra, N. (2012). A Comparative Assessment of Ranking Accuracies of Conventional and Machine-Learning-Based Scoring Functions for Protein-Ligand Binding Affinity Prediction. *IEEE/ACM Trans. Comput. Biol. and Bioinf.* 9(5), pp.1301-1313. Available at: http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=6171157&url=http%3A%2F%2Fieeexplore.ieee.org%2Fiel5%2F8857%2F4359833%2F06171157.pdf%3Farnumber%3D6171157 [Accessed 5 Oct. 2015].

Balani S.K., Devishree V.S., Miwa G.T., Gan L.S., Wu J.T., Lee F.W. (2005). "Strategy of utilizing in vitro and in vivo ADME tools for lead optimization and drug candidate selection". *Curr Top Med Chem., 5 (11)*, pp.1033–8. Available at: http://www.ncbi.nlm.nih.gov/pubmed/16181128 [Accessed 29 Oct. 2015].

Blaszczyk, M., Kurcinski, M., Kouza, M., Wieteska, L., Debinski, A., Kolinski, A. and Kmiecik, S. (2015). Modeling of protein–peptide interactions using the CABS-dock web server for binding site search and flexible docking. *Methods*. Available at: http://arxiv.org/ftp/arxiv/papers/1505/1505.01138.pdf [Accessed 06 Dec. 2015].

Bredensteiner, E. and Bennett, K. (2000). Duality and Geometry in SVM Classifiers. *Department of Mathematics*, 1, pp.1-8. Available at: http://www.robots.ox.ac.uk/~cvrg/bennett00duality.pdf [Accessed 14 Jan. 2016].

Chinnasamy, A., Sung, W. And Mittal, A. (2005). Protein Structure and Fold Prediction Using Tree-Augmented Naïve Bayesian Classifier. *Journal of Bioinformatics and Computational Biology*, 03(04), pp.803-819. Available at: http://psb.stanford.edu/psb-online/proceedings/psb04/chinnasamy.pdf [Accessed 20 Oct. 2015].

Cohen, J. (2004). Bioinformatics—an Introduction for Computer Scientists. *ACM Computing Surveys.*, 36(2), pp.123-158. Available at: http://www.cs.indiana.edu/~predrag/files/cohen\_2004.pdf [Accessed 14 Oct. 2015].

Ding, C. and Dubchak, I. (2001). Multi-class protein fold recognition using support vector machines and neural networks. *Bioinformatics*, 17(4), pp.349-358. Available at: http://ranger.uta.edu/~chqding/papers/protein.pdf [Accessed 06 Dec. 2015].

Fox, J. and Rawlings, C. (1994). Artificial intellugence and knowledge based systems in molecular biology\*. *The Knowledge Engineering Review*, 9(03), p.287. Available at: http://www.academia.edu/17944921/artificial-intelligence\_and\_knowledge-based\_systems\_in\_molecular-biology [Accessed 7 Oct. 2015].

Furlanello, C., Serafini, M., Merler, S. and Jurman, G. (2005). Semisupervised Learning for Molecular Profiling. *IEEE/ACM Trans. Comput. Biol. and Bioinf.* 2(2), pp.110-118. Available at: http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=1438348&url=http%3A%2F%2Fieeexplore.ieee.org%2Fiel5%2F8857%2F30971%2F01438348 [Accessed 6 Oct. 2015].

Gostner, R., Baldacci, B., Morine, M. and Priami, C. (2014). Graphical Modeling Tools for Systems Biology. *CSUR*, 47(2), pp.1-21. Available at: https://www.cs.umd.edu/class/fall2011/cmsc858s/Alignment.pdf [Accessed 12 Oct. 2015].

Karp, P. and Mavrovouniotis, M. (1994). Representing, analyzing, and synthesizing biochemical pathways. *IEEE Expert*, 9(2), pp.11-21. Available at: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.45.352&rep=rep1&type=pdf [Accessed 12 Oct. 2015].

Lounnas, V., Ritschel, T., Kelder, J., McGuire, R., Bywater, R. and Foloppe, N. (2013). Current Progress In Structure-Based Rational Drug Design Marks A New Mindset In Drug Discovery. *Computational and Structural Biotechnology Journal*, 5(6), pp.1-14. Available at: http://www.sciencedirect.com/science/article/pii/S2001037014600398 [Accessed 23 Oct. 2015].

Mandal, S., Moudgil, M. and Mandal, S. (2009). Rational drug design. *European Journal of Pharmacology*, 625(1-3), pp.90-100. Available at: http://www.udel.edu/chem/bahnson/chem645/Rational\_drug\_design\_Abhijit.pdf [Accessed 22 Oct. 2015].

Marshall M. (2016). Iris Dataset. *UCI Machine Learning Repository.* Irvine, CA: University of California, School of Information and Computer Science. Available at: http://archive.ics.uci.edu/ml/datasets/Iris [Accessed 21 Oct. 2015].

McCubbin C. (2003). Legal issues in bioinformatics. *Journal of commercial biotechnology*, 9(3), pp.249-265. Available at: http://commercialbiotechnology.com/article/viewFile/34/30 [Accessed 31 Jan. 2016].

Meller J. (2001). Molecular Dynamics. *Encyclopaedia of life sciences nature publishing group*, 1, pp.1-8. Available at: https://folding.cchmc.org/publications/md\_els.pdf [Accessed 14 Jan. 2016].

McIntosh-Smith, S., Price, J., Sessions, R. and Ibarra, A. (2014). High performance in silico virtual drug screening on many-core processors. *International Journal of High Performance Computing Applications*, 29(2), pp.119-134. Available at: http://hpc.sagepub.com/content/early/2014/05/13/1094342014528252.full.pdf [Accessed 4 Oct. 2015].

Noble, S. (2003). Support vector machine applications in computational biology. *Washington University Department of Genome Sciences*, pp.1-31. Available at: https://www.ibisc.univ-evry.fr/~dalche/cours/noble\_support.pdf [Accessed 10 Oct. 2015].

Rigand, N. (2008). Biotechnology: Ethical and social debates. *OECD* *International futures programme*, pp.1-89. Available at: http://www.oecd.org/futures/long-termtechnologicalsocietalchallenges/40926844.pdf [Accessed 30 Jan. 2016].

Schierz A. (2009). Bioassay Data Set. *UCI Machine Learning Repository*. Irvine, CA: University of California, School of Information and Computer Science. Available at: http://archive.ics.uci.edu/ml/datasets/PubChem+Bioassay+Data [Accessed 20 Mar. 2016]

Smitha, M., Mitra, A., Singh, H. (2008). Real Valued Genetic Algorithm Based Approach for Protein Structure Prediction - Role of Biophysical Filters for reduction of Conformational Search Space. *Third IAPR International Conference on Pattern Recognition in Bioinformatics*, pp.1-20. Available at: http://www.researchgate.net/publication/237550701\_Real\_Valued\_Genetic\_Algorithm\_Based\_Approach\_for\_Protein\_Structure\_Prediction\_-\_Role\_of\_Biophysical\_Filters\_for\_Reduction\_of\_Conformational\_Search\_Space [Accessed 11 Oct. 2015].

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R. (2014). A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research 15.* pp.1931-1958. Available at: https://www.cs.toronto.edu/~hinton/absps/JMLRdropout.pdf [Accessed 7 Oct. 2015].

Tsay, J. and Su, S. (2013). An effective evolutionary algorithm for protein folding on 3D FCC HP model by lattice rotation and generalized move sets. *Proteome Sci*, 11(Suppl 1), p.S19. Available at: http://www.proteomesci.com/content/11/S1/S19 [Accessed 12 Oct. 2015].

Tsumoto, K., Kamiya, K., Yoshimura, T. (2007). Display of Recombinant Membrane Receptors on Giant Liposomes: Attempt to Construct a Cell Model with Integrated Membrane Protein Systems. *978- 1-4244-] 858-9/07/$25.00 ©C2007 IEEE.*pp.102-107. Available at: http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=4420834&url=http%3A%2F%2Fieeexplore.ieee.org%2Fiel5%2F4420810%2F4420811%2F04420834.pdf%3Farnumber%3D4420834 [Accessed 15 Oct. 2015].

Van der Walle, C. (2011). Peptide and protein delivery. London: Academic Press. Available at: http://cst.ur.ac.rw/library/Food%20Science%20books/batch1/Peptides%20and%20protein%20delivery.pdf [Accessed 14 Jan. 2016].

Wang, J., Urban, L. (2004). The impact of early ADME profiling on drug discovery and development strategy. *Drug Discovery World Fall 2004.*pp.73-86. Available at: http://www.ddw-online.com/media/32/2460/04.fal.the-impact-of-early-adme-profiling-on-drug-discovery-and-development-strategy.pdf [Accessed 5 Oct. 2015].

Wang, Y., Xing, J., Xu, Y., Zhou, N., Peng, J., Xiong, Z., Liu, X., Luo, X., Luo, C., Chen, K., Zheng, M. and Jiang, H. (2015). In silico ADME/T modelling for rational drug design. *Quarterly Reviews of Biophysics*, 48(04), pp.488-515. Available at: http://www.ncbi.nlm.nih.gov/pubmed/26328949 [Accessed 21 Oct. 2015].

Wu, S., Zhang, Y. (2009). Protein Structure Prediction. *Bioinformatics: Tools and Applications, DOI 10.1007/978-0-387-92738-1\_11.*, pp.225-242. Available at: http://zhanglab.ccmb.med.umich.edu/papers/2009\_8.pdf [Accessed 15 Oct. 2015].

Yoruk, E., Ochs, M., Geman, D. and Younes, L. (2011). A Comprehensive Statistical Model for Cell Signaling. *IEEE/ACM Trans. Comput. Biol. and Bioinf.* 8(3), pp.592-606. Available at: http://cis.jhu.edu/people/faculty/geman/publications/pdf/Geman\_IEEETransCBB\_2011.pdf [Accessed 5 Oct. 2015].

Appendix C – Further Reading

## Ethical, legal and social issues

Large private projects on sequencing DNA and creating new molecules aim to patent it, *“many patent applications have been filed in the field of genomics/functional genomics claiming the output from largescale DNA sequencing projects“*(C. McCubbin, 2003, p252). Software which is created for biotechnology reasons can also be patentable, because it is usually unique and not easily discoverable in the same way without actually copying or knowing something about it. From private company owners point of view it seems unfair that drug they developed and sequences they discovered through many costly years are not patentable and therefore it causes difficulties to have a profit. But from another point of view it would be unethical if one company discovers cure to cancer, but government discovers it later, people could not use the free cure, because they would have to buy previously patented one. That also leads to questions about if there was most efficient molecule found, then others are not allowed to use it or reproduce it even in their own way without paying to first inventor? Unique software, patented without problems, could possibly produce the results which are not supposed to be private. What if that software creates the cure, while being patented itself, would its results also be patented? There are also socially not acceptable issue examples like genetically modified foods and experiments with animals. Other problems include *“social views on such issues as whether knowledge is being commoditised, whether it is acceptable to patent living organisms, innovations derived from traditional local knowledge and active ingredients from plants considered sacred, and whether the industry should share benefits with local communities”* (N. Rigand, 2008, p20).