Aalto University
School of Science

Master's Programme in Life Science Technologies

Study on projective feature selection on molecular fingerprint data

Mika Sorvoja

Master's Thesis
2024

Aalto University
School of Science

Author Mika Sorvoja

Title Study on projective feature selection on molecular fingerprint data

Degree programme Life Science Technologies

Major Complex Systems

Supervisor Prof. Juho Rousu

Advisor Dr Riikka Huusari, Dr Sandor Szedmak

Date 19 September 2024 **Number of pages** 53+6 **Language** English

Abstract

This thesis deals with molecular fingerprints and anti-cancer drug response data. It is studied whether features that contribute the most to the response the anti-cancer drugs elicit can be found with the help of feature selection. For this, mainly two different algorithms are used and compared: a basic linear Pearson correlation coefficient-based feature ranking algorithm and a novel kernel-based projective feature selection algorithm. The success of feature selection is evaluated using kernel alignment and regression. The former is used for examining how well feature selection maintains similarity between data points, while the latter is for predicting drug response values from the fingerprint data. In the regression problem, a few different learners are compared: the two mainly used are linear and random forest regression, but also support vector regression and two slightly different multilayer perceptrons are applied to the task. The results show that the feature selection algorithm relying on projection operators is superior to the simple linear one, finding features contributing both to higher kernel alignment and more accurate predictions of drug response values. Regressor-wise, the algorithm producing the most accurate predictions, on average, is the random forest regression while the linear regression yields the most inaccurate ones. However, none of the algorithms used achieves satisfactory prediction accuracy. When it comes to the different molecular fingerprints, it appears that the substructure keys-based produce slightly more accurate predictions than topological or circular fingerprints, on average. In conclusion, the results of this thesis show that predicting drug response values from molecular fingerprint data is a challenging task. However, with a suitable feature selection algorithm, it is possible to improve learning speed while maintaining almost as high prediction accuracy as with full features. Further studies might find more suitable learning algorithms producing more satisfactory results.

Keywords machine learning, feature selection, ProjSe, kernel methods, NCI-60 anti-cancer drug response data, molecular fingerprints

Aalto-yliopisto
Perustieteiden korkeakoulu

Tekijä Mika Sorvoja	
Työn nimi Tutkielma molekulaarisen sormenjälkidatan projektiivisesta muuttujanvalinnasta	
Koulutusohjelma Life Science Technologies	
Pääaine Complex Systems	
Työn valvoja Prof. Juho Rousu	
Työn ohjaaja Dr Riikka Huusari, Dr Sandor Szedmak	
Päivämäärä 19.9.2024	**Sivumäärä** 53+6 **Kieli** englanti

Tiivistelmä

Tässä työssä käsitellään molekulaarisia sormenjälkiä ja syöpälääkevastedataa. Tutkitaan voidaanko muuttujanvalinnan avulla löytää sellaisia ominaisuuksia, jotka myötävaikuttavat eniten syöpälääkkeiden elimistössä aiheuttamaan vasteeseen. Tähän tarkoitukseen käytetään ja verrataan keskenään pääasiallisesti kahta eri algoritmia: ensimmäinen näistä on yksinkertainen lineaarinen muuttujanvalinta-algoritmi, joka laittaa muuttujat niiden Pearsonin korrelaatiokertoimien mukaiseen paremmuusjärjestykseen, ja toinen on uusi kernel-pohjainen projektiivinen muuttujanvalinta-algoritmi. Muuttujanvalinnan onnistumista arvioidaan käyttäen kernel-kohdistusta ja regressiota. Ensimmäisellä menetelmällä tutkitaan kuinka hyvin muuttujanvalinta säilyttää datapisteiden välisen samankaltaisuuden, kun taas jälkimmäistä käytetään ennustamaan lääkevastearvoja sormenjälkidatasta. Regresio-ongelmassa verrataan keskenään muutamaa eri oppijaa, joista kaksi tärkeintä ovat lineaarinen ja satunnaismetsäregressio, joiden lisäksi tehtävässä käytetään tukivektoriregressiota ja kahta keskenään hieman erilaista monikerroksista perseptroniverkkoa. Tulokset osoittavat, että projektiooperaattoreihin perustuva muuttujanvalinta-algoritmi on ylivertainen yksinkertaiseen lineaariseen algoritmiin nähden, sillä sekä kernel-kohdistus että lääkevastearvojen ennustaminen antavat parempia tuloksia ensiksi mainittua muuttujanvalintamenetelmää käytettämällä. Regressiomalleja verratessa, kaikkein tarkimpia ennustuksia tuottaa satunnaismetsäalgoritmi. Mikään käytetyistä regressioalgoritmeista ei kuitenkaan onnistu tuottamaan tyydyttävän tarkkoja ennustuksia. Mitä tulee eri molekulaarisiin sormenjälkiin, tulosten perusteella näyttää siltä, että alarakenneavaimiin perustuvat (substructure keys-based) sormenjäljet tuottavat keskimäärin tarkempia ennustuksia, kuin topologiset tai säteeseen perustuvat (circular) sormenjäljet. Yhteenvetona voidaan todeta, että tulokset osoittavat lääkevastearvojen ennustamisen olevan haastava tehtävä. Sopivalla muuttujanvalinta-algoritmilla voidaan kuitenkin parantaa oppimisnopeutta säilyttäen samalla likimain yhtä korkea ennustustarkkuus kuin käytettäessä kaikkia muuttujia. Lisätutkimuksissa saatettaisiin löytää sopivampia regressiomalleja tuottamaan tyydyttävämpiä tuloksia.

Avainsanat koneoppiminen, muuttujanvalinta, ProjSe, kernel-menetelmät, NCI-60 syöpälääkevaste data, molekulaariset sormenjäljet

Preface

First of all, I would like to extend my deepest gratitude to Dr Riikka Huusari for providing me excellent guidance throughout the thesis process. I would also like to thank Dr Sandor Szedmak for many useful suggestions to improve the experiments and Professor Juho Rousu for helpful comments and suggestions while refining the thesis. Finally, I would like to thank my family for their support.

Otaniemi, 19 September 2024

Mika Sorvoja

Contents

Symbols and abbreviations

Symbols

x	Data point
$\mathbf{x}^i$	Data sample vector
$\mathbf{x}_i$	Feature vector
$\mathbf{X}$	Data sample matrix: samples on rows, features on columns
n	Number of rows in $\mathbf{X}$ i.e. length of a feature vector
m	Number of columns in $\mathbf{X}$ i.e. length of a sample vector
$\tilde{\mathbf{X}}_t$	Data sample matrix from which features selected at previous iterations of feature selection have been removed
y_i	Regression target
$\mathbf{y}$	Output vector, regression targets
$\mathbf{Y}$	Reference data matrix
m_Y	Number of columns in matrix $\mathbf{Y}$
$\mathbb{N}$	Set of natural numbers
$\mathbb{R}$	Set of real numbers
$O(\cdot)$	Computational (time) complexity
σ	Standard deviation
$\mathcal{X}$	Input space
$\mathcal{H}$	Reproducing kernel Hilbert space
$\mathcal{L}_Y$	Subspace of $\mathcal{H}$ spanned by the columns of $\mathbf{Y}$
L_1	$L1$-norm

Operators

$\in$	Belongs to
$\cap$	Intersection
$\cup$	Union
$\subseteq$	Subset
$\mathbf{P}_{\mathcal{L}_Y}$	Orthogonal projection into subspace $\mathcal{L}$ spanned by the columns of $\mathbf{Y}$
$\kappa(\cdot,\cdot)$	$\kappa : \mathcal{X} \times \mathcal{X} \to \mathcal{R}$
$\phi(\cdot)$	$\phi : \mathcal{X} \to \mathcal{H}$
$^\top$	Transpose
$^+$	Pseudoinverse

Abbreviations

CCA	Canonical correlation analysis
CDK	Chemistry development kit
LASSO	Least absolute shrinkage and selection operator
LR	Linear regression
MI	Mutual information
ML	Machine learning
MLP	Multilayer perceptron
MSE	Mean squared error
NCI	(The US) National Cancer Institute
PCA	Principal component analysis
PCC	Pearson's correlation coefficient
ProjSe	Projective selection algorithm
PTC	Percentage of Treated cell growth as a fraction of Control cell growth
RBF	Radial basis function
RFR	Random forest regression
ReLU	Rectified linear unit
RKHS	Reproducing kernel Hilbert space
SMILES	Simplified molecular-input line-entry system
SVD	Singular value decomposition
SVM	Support vector machine
SVR	Support vector regression

1 Introduction

Many devices are able to gather not just one but multiple types of data from the same source. One example of such device is nowadays widely popular: a smart watch can observe things like heart rate, blood oxygen level, GPS location, and the movement of its wearer from the wrist. Such data is called multi-view, or multi-modal. Its main advantage is that is can describe the measurement environment more comprehensively than single-view data (Zhao et al., 2017). Another example of data type that is often multi-view is biological data. For instance, with high-throughput screening, a method used, among others, in drug discovery, it is possible to conduct millions of tests on molecules in short time. The test measurements on the same molecule are its different views.

In machine learning (ML), there are a few different approaches to learning from such data: in so called early fusion, the views are combined into one large single-view data that is learned with a single-view algorithm. Another approach is to have separate algorithms trained for each view, and then the results are combined (late fusion). Third approach is to have one true multi-view algorithm that can learn from multiple views all at once. Additionally, there are algorithms such as canonical correlation analysis (CCA) (Hotelling, 1992) in which features from one view are learned with the help of another view.

Nowadays, data (both single and multi-view) is often very large, i.e, the number of samples n is huge. In addition, data can be wide, in which case it is called fat. To be precise, fat data does not necessarily contain a huge number of samples, but it does have a high number of features m. A problem with fat data is that not every feature is relevant in every case. Often it is unknown which features are of interest. Furthermore, high dimensional data tends to be sparse: data points are far apart, leaving a lot of space between them. The higher the dimensionality the sparser the data (Li et al., 2017). This is called the curse of dimensionality. It can lead to algorithms designed for low-dimensional space to perform poorly. A way to address this issue is to use some dimensionality reduction technique. Perhaps the most commonly used dimensionality reduction technique is principal component analysis (PCA) (Pearson, 1901; Hotelling, 1933).

There are two types of dimensionality reduction techniques: feature extraction and feature selection (Li et al., 2017). In the former, high-dimensional features are projected into a new low-dimensional feature space while in the latter a subset of relevant features are selected. PCA is a widely popular example of a feature extraction method while Pearson correlation coefficient (PCC) (Pearson, 1895) and mutual information (MI) (Shannon, 1948) are examples of metrics that can be used in feature selection for ranking variables in so-called filter type feature selection methods. While both feature extraction and feature selection are valid techniques for dimensionality reduction, this thesis focuses on the latter. The reason for this choice is that unlike feature selection, feature extraction transforms the features into their linear or nonlinear combinations that have poor interpretability in general.

Although feature selection is a common tool for reducing the dimensionality of fat data, it can be used on data with less features as well: in general, feature selection is used to improve generalization capacity and learning speed of ML models (Kumar and Minz, 2014). Especially with more complicated learning algorithms, time complexity is often an issue. Reducing the dimensionality of the data may lead to improved performance. However, there is the danger that relevant features are removed in the process. Another possible pitfall, depending on the choice of feature selection algorithm, is that the dimensionality reduction itself can become the bottleneck as many methods tend not to scale very well.

An example of data with a large number of features, on which feature selection could be applied, are the molecular fingerprints. Each fingerprint is a unique identifier that contains information about the structure of the molecule it corresponds to. There is a multitude of algorithms for generating molecular fingerprints all of them producing identifiers with varying number of features: the ones used in the experiments of this thesis range from 166 to 2048. As a side note: while these fingerprints contain up to a good couple of thousand features, data consisting of them is still not fat in particular. In addition to the fingerprints, this thesis deals with the NCI-60 human tumor cell lines screen growth inhibition dataset (National Cancer Institute, 2024) which contains drug response measurements for anti-cancer drugs at different dose concentrations. For each drug, a molecular fingerprint can be generated. This raises a question: can response values be predicted from the molecular fingerprints? That will be explored in this thesis. Furthermore, it is studied if feature selection is able to find features having the greatest impact on drug response.

As in ML in general, feature selection can roughly be divided into unsupervised and supervised learning. There is also a third area called reinforcement learning, but it is ignored in this thesis as it is not as prevalent in feature selection as the other two. In general sense, unsupervised algorithms learn on their own, detecting underlying patterns in data. Typical examples of such methods are the different clustering algorithms, such as K-means (MacQueen et al., 1967) and spectral clustering (Ng et al., 2001). Supervised methods, on the other hand, require human supervision in the form of labels. The algorithms learn with training examples to map samples $\{\mathbf{x}^i\}_{i=1}^n$ to class labels or regression targets $\{y_i\}_{i=1}^n$. The labels can be, e.g. $y_i \in \{0, 1\}$ for binary classification and $y_i \in \mathbb{R}$ for regression. A widely popular example of a supervised algorithm suitable for both classification and regression is the support vector machine (SVM) (Cortes and Vapnik, 1995). When it comes to feature selection, traditional unsupervised algorithms work typically with single-view data, while traditional supervised algorithms can be suitable for multi-view settings, but are limited to data containing just one explanatory variable. Thus, there are few methods suitable for both two-view settings and vector-valued tasks (Szedmak et al., 2023).

One further issue with many feature selection algorithms is that they are linear. However, real-world data often contains nonlinearities which the linear methods cannot grasp. Therefore, nonlinear algorithms often offer a better option. Kernel-based methods are one possibility for nonlinear feature selection. They are functions that define an inner product in a high dimensional space (Mohri et al., 2018). Kernels can be thought as a similarity measure between elements of the original data space. Kernels allow mapping data into a high dimensional reproducing kernel Hilbert space (RKHS) $\mathcal{H}$, which enables linear computation on nonlinear features (Genton, 2001). Some important advantages of kernels are their computational efficiency and flexibility (Mohri et al., 2018). While kernel-based methods have their advantages, most algorithms have $O(n^3)$ time complexity which leads to significant computational limitations on large datasets.

A recent algorithm by Szedmak et al. (2023) offers a highly scalable kernel-based alternative to traditional feature selection methods. The algorithm, ProjSe, relying on projection operators, is suitable both for two-view settings and vector-valued supervised learning problems. ProjSe uses a kernel-based representation of variables which enables grasping nonlinearities in input data. It assumes that data samples $\mathbf{x}^i$ are on rows in the input matrix $\mathbf{X}$, and unlike traditional kernel-based methods, it considers features $\mathbf{x}_i$, i.e. the column vectors, instead of samples when computing the kernel. This enables the high scalability of the algorithm on large datasets, assuming the number of features m is not too large, i.e. in tens of thousands.

The goals of this thesis are to discover whether it is possible 1) to accurately predict the response values of different anti-cancer drugs from molecular fingerprint data using regression, and 2) with feature selection, to find features in the fingerprints that contribute the most to the response of said drugs. For feature selection, ProjSe is mainly compared to one very basic linear filter-type feature selection method that performs feature ranking using PCC as the ranking criterion. The success of feature selection is evaluated using both an unsupervised and a supervised method. While the prediction problem (supervised) is the main interest of this thesis, also kernel alignment (unsupervised) is used in the evaluation. In the regression tasks, several models are compared: linear, random forest (Breiman, 2001) and support vector regressor (SVR), and two slightly different single hidden layer multilayer perceptrons (MLP) (Rosenblatt, 1958), one of which performs feature selection by itself.

The structure of this thesis is the following: Section 2 discusses the NCI-60 data and the molecular fingerprints. In Section 3, a review of feature selection is given followed by discussion on kernels before the details of ProjSe are presented. Finally, the ways to evaluate the success of feature selection are discussed, and the methods used in the experiments, i.e. kernel alignment and the different regressors are presented. In Section 4, the workflow of the experiments is discussed, and Section 5 presents the results. Finally, Section 6 concludes the thesis and summarizes the findings.

2 Materials

In medicine, drug discovery is an example of a field where computers are used to help in finding promising new medicines for diseases for which less effective treatments already exist or for which there is no known cure. The problem in drug discovery often is that there are countless numbers of potential molecules from which to choose (Walters et al., 1998). Virtual screening offers a way to filter huge libraries of chemical compounds to find suitable molecules which to synthesize for drug testing.

Molecular fingerprints can be exploited in virtual screening in drug discovery (Cereto-Massagué et al., 2015). In computing in general, fingerprints are bit strings that act as unique identifiers. Molecular fingerprints contain information about the structure of chemical compounds. Fingerprints significantly simplify computation as they offer a simple vector representation to complex 2D, or even 3D, structures of chemical compounds. Molecular fingerprints are typically long, containing hundreds of features (see Table 1). Therefore, it may be beneficial to apply feature selection on such data. Furthermore, if there is some other data, e.g. measurements, available for the molecules the fingerprints represent (a multi-view setting), that data can be used as reference in feature selection to find the most significant features in that setting. So for example, if there are measurements that describe the strength of the response each drug molecule elicits, feature selection can be applied in order to try to find the features that contribute to it the most. This is what is done in this thesis.

In this section, the NCI-60 dataset and the different types of molecular fingerprints are discussed, and the fingerprints used in the experiments are presented.

2.1 NCI-60 data

The data used in this thesis is from the NCI-60 human tumor cell lines screen growth inhibition dataset (National Cancer Institute, 2024). It contains drug response data, more specifically PTC, or the Percentage of Treated cell growth as a fraction of Control cell growth, values for more than 50 000 anti-cancer drugs measured at different dose concentrations and tested on several tumor cell lines. In total, there are currently measurements for 73 different cell lines in the dataset which is subject to change as the dataset is being maintained. The PTC values are non-negative real numbers from zero onwards that can be interpreted as follows: values less than 100 indicate that the drug inhibited the tumor cells growth. The closer the value is to zero, the stronger is the inhibition. Value 100 suggests that the drug had no effect on the tumor cells, and values greater than 100 indicate that the drug promoted tumor cell growth compared to the control that did not receive the drug. The dataset contains also GIPRCNT values that have slightly different interpretation than PTC values, and are corrected for the count of cells at the time the drug was added. GIPRCNT values are in range $[-100, 100]$ where -100 refers to complete cell kill, 0 to complete cytostasis, i.e. complete inhibition of cell growth, and 100 refers to the control growth, i.e. growth without adding the drug. In this work, the PTC values are used both as reference data in feature selection, and as regression targets which are predicted from the molecular fingerprint data. In the NCI-60 dataset, each drug is identified by a numeric NSC code.

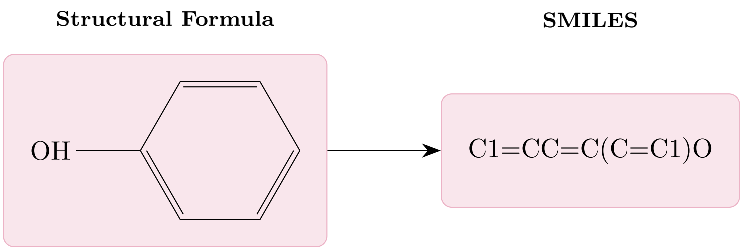


Figure 1: An example showing how the structural formula of phenol transforms into a SMILES string.

These codes can be used to access the SMILES (simplified molecular-input line-entry system) strings (see Figure 1) of the molecules to generate molecular fingerprints.

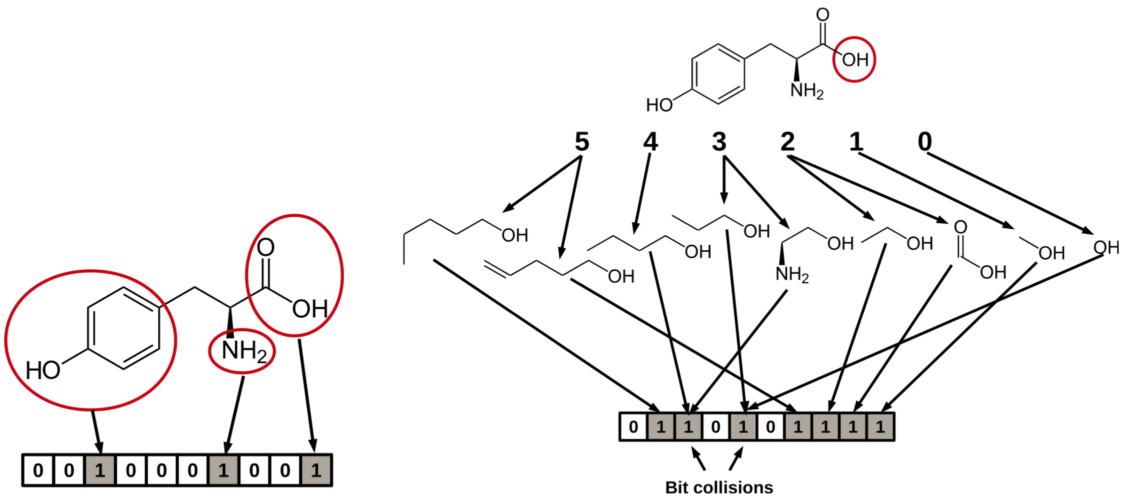
2.2 Molecular fingerprints

There are three main types of molecular fingerprints: substructure keys-based, topological or path-based, and circular fingerprints (Cereto-Massagué et al., 2015). Substructure keys-based fingerprints are the most straightforward type of molecular fingerprints. Each bit, or key, refers to a certain substructure also called as a molecular descriptor: value one indicates that the substructure is present in the molecule while zero indicates its absence. As the keys are predefined, substructure keys-based fingerprints are useful only with data that contains those substructures to a large extent. Otherwise the fingerprints are just bit vectors of zeros. Figure 2a illustrates substructure keys-based type of fingerprints.

Topological or path-based fingerprints showcased in Figure 2b work a bit differently. Instead of searching for certain substructures in a molecule, paths from each atom up to a certain number of bonds are analyzed (Cereto-Massagué et al., 2015). The paths are then hashed into bits creating the fingerprint. The bits in topological (and also in circular) fingerprints are not predefined like in substructure keys-based fingerprints but are evaluated during the fingerprint generation, i.e. they depend on the data at hand. Only the length of the fingerprints is fixed. The hashing has one major setback: it is not possible to revert the fingerprint creation, i.e., get the original features from the bits. This is because a bit may be set by several features causing a phenomenon called bit collision. Like the substructure keys-based fingerprints, topological fingerprints can be used for searching substructures.

Circular fingerprints are generated otherwise similarly as the topological ones, but instead of paths, the neighborhood of each atom is analyzed up to a predefined radius (Cereto-Massagué et al., 2015). As with topological fingerprints, the neighborhood is hashed into bits. An atom may appear in multiple neighborhoods, and hence the circular fingerprints are not suitable for substructure searching. However, they can be used for full structure similarity searching. Like topological fingerprints, circular fingerprints are susceptible to bit collisions as well.

In addition to the three main fingerprint types, Cereto-Massagué et al. (2015) describes two other classes of molecular fingerprints as well: hybrid and pharmacophore



(a) Substructure-keys based **(b)** Topological or path-based

Figure 2: Hypothetical fingerprints created from a molecule containing three substructures. Occurrences of two bit collisions are visible in the figure displaying topological fingerprint. The images are from Cereto-Massagué et al. (2015).

fingerprints. Hybrid fingerprints are such that consist both of structural keys and generated bits, while pharmacophore fingerprints are otherwise similar to substructure keys-based fingerprints but they take into account the 3D structure of molecules (while most fingerprints only consider the 2D structure). Additionally, there are a few molecular fingerprints that do not fall into any of the discussed categories (Cereto-Massagué et al., 2015).

In the experiments of this thesis, seven different fingerprints are used: MACCS (Durant et al., 2002), PubChem (National Health Institute, 2009), Morgan 2 (Morgan, 1965; Rogers and Hahn, 2010), ErG (Stiefl et al., 2006), RDKit (Landrum, 2012), Graph and Shortestpath. They are summarized in Table 1. The fingerprints are generated from the SMILES strings using three different libraries: the open-source cheminformatics and machine learning package RDKit (RDKit, 2013), the chemistry development kit CDK (Willighagen et al., 2017), and PubChemPy (Swain, 2014), a python wrapper for PubChem.

Heatmaps of Pearson correlation matrices of the features in the seven different fingerprints are shown in Figure 3. The heatmaps show that the similarity of features in the fingerprints varies. In the MACCS, PubChem, and ErG, there are several features that are similar to one another, some more strongly than others. Especially in the ErG heatmap, there are several large clusters visible. On the other hand, in the Morgan 2 and Graph fingerprints, hardly any of the features is similar to another one. While the Graph contains some feature pairs that are either weakly or strongly correlated, in the Morgan 2 fingerprints approximately all the pairwise correlations are zero. Another interesting pair of fingerprints is the RDKit and Shortestpath. While there are few strongly correlated feature pairs, almost all of the features are weakly correlated with each other. As a side note, there are many features in the PubChem fingerprints that do not appear in any of the approximately 55000 drugs which is why there are white stripes in the heatmap.

Fingerprint	Type	#Features	Data type
MACCS	substructure keys-based	166	[0,1]
PubChem	substructure keys-based	881	[0,1]
Morgan 2	circular	2048	[0,1]
ErG	topological	315	$\mathbb{R}_+$
RDKit	topological	2048	[0,1]
Shortestpath	topological	1024	[0,1]
Graph	topological	1024	[0,1]

Table 1: The fingerprints used in the experiments.

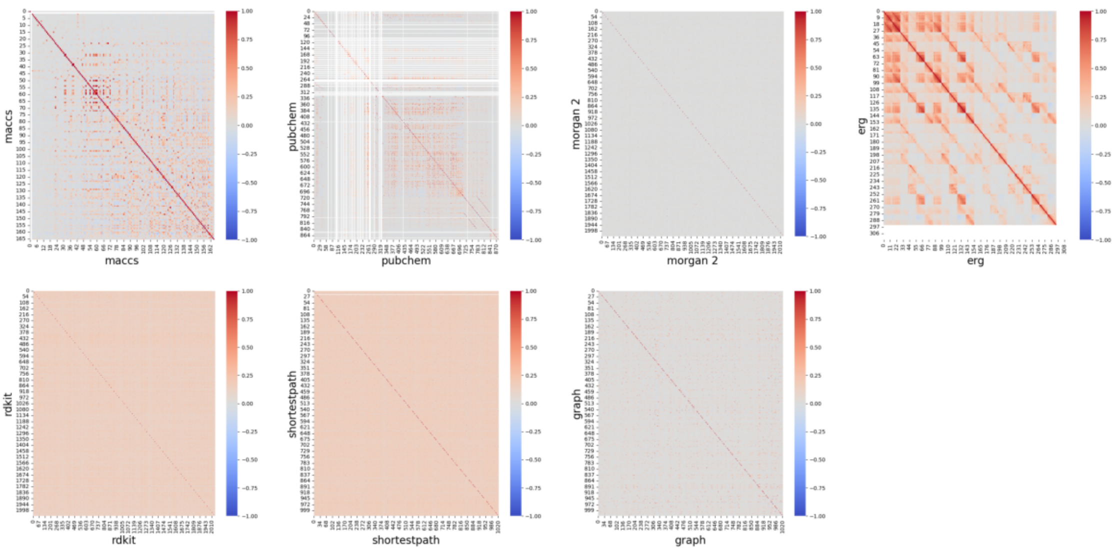


Figure 3: Heatmaps of Pearson correlations matrices of the features in the different fingerprints. White stripes in the heatmaps of PubChem and Shortestpath fingerprints represent missing values caused by features containing only zeros.

3 Methods

This thesis focuses on feature selection. It is applied on the molecular fingerprints of the anti-cancer drugs present in the NCI-60 dataset in order to study whether it is possible to find features that contribute most to the response the drugs elicit in human tumor cells. Particularly, it is examined if the kernel-based projective feature selection algorithm ProjSe is able to select relevant features from molecular fingerprints with the help of drug response data, and to compare it to a traditional feature selection algorithm. However, the selected features themselves are not interpreted in this thesis for two reasons: 1) transforming the features into actual molecular structures is, if not impossible, far from a straightforward process, and 2) feature selection and the evaluation of its success are the main interests.

In this section, feature selection is first discussed in general and the main types of feature selection algorithms are reviewed. Then, the basics of kernels and kernel-based feature selection are discussed and the details of ProjSe (Szedmak et al., 2023) are presented. Finally, the methods used for unsupervised and supervised evaluation of the success of feature selection are reviewed.

3.1 Feature selection

Unwanted features can be removed from data with feature selection, ideally without considerable loss of information (Li et al., 2017). Having less features may help with learning performance since real-world data is often noisy and contains redundancies. Feature redundancy is traditionally defined as two features having a perfect correlation to each other (Huang, 2015). On the other hand, Cover (1974) noted that two independent features can be redundant as well: "the best two independent measurements (features) are not the two best". It can be showed that the most informative subset of two or more features, i.e. a feature subset yielding the lowest probability of error, is not always the union of the two individually most informative. Guyon and Elisseeff (2003) showed with two examples that while reduction in noise can be achieved by using independent features, two highly correlated features can also improve prediction accuracy when combined, unlike when using either feature alone. This indicates that feature correlation is not equal to feature redundancy. Therefore, defining feature redundancy is not a simple task.

An important concept in feature selection related to redundancy is relevancy. Feature selection is about finding an optimal subset, i.e. such subset that contains only relevant features. Definition of feature relevancy is dependent on the context: "relevant to what?" (Blum and Langley, 1997) Therefore, there are many definitions of feature relevancy: features can be relevant either to a target, to the sample, or to some distribution. Other definitions exist as well, see e.g. Blum and Langley (1997). However, finding the optimal feature subset, is often NP-hard (Kohavi and John, 1997) making it computationally impractical: for data with m features, $2^m - 1$ need to be evaluated (Kumar and Minz, 2014). Therefore, the best strategy often is to find a subset that is not optimal but sufficiently good nonetheless.

In feature selection, finding the feature subset commonly involves the following

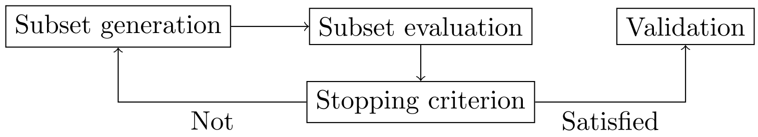


Figure 4: The general outline of feature selection algorithms.

stages: subset generation, subset evaluation, and validation of the results (Jović et al., 2015), see Figure 4. The first two steps are repeated using some stopping criterion until the best subset is found, which is then validated, for example, on a separate dataset.

In subset generation, different search strategies can be used in search for feature subset (Kumar and Minz, 2014). Search can be sequential, exponential or random. Sequential search performs search iteratively. It is simple to implement but does not yield an optimal subset like the more complex exponential search guarantees to find. Exponential search can be either exhaustive or use some heuristic function. Random search can find an optimal subset as well, but it includes randomness in the process to avoid ending up at local optima. The order of the search space of each type of search strategy is $O(m^2)$.

In the second step, i.e.subset evaluation, feature subsets are evaluated using either independent or dependent evaluation criteria. Kumar and Minz (2014) describes six different types of measures as independent criteria: distance or divergence, information or uncertainty, probability of error, dependency, consistency, and interclass distance. Independent criteria are used with so-called filter type feature selection methods and are independent of any learning algorithm used in the following steps of the analysis. On the other hand, dependent evaluation criteria are such that are used together with a specific pre-selected learning algorithm. The algorithm is used to evaluate different feature subsets and select the best. Thus, the selected subset is highly dependent on the specific algorithm that is used in the evaluation. The feature selection methods that use dependent criteria with some learning algorithm are called wrappers.

Stopping criteria ends the search for the best feature subset. Kumar and Minz (2014) lists some common stopping criteria: maximum number of iterations, minimum number of features, minimum classification error, or lack of further major improvement or deterioration when adding or removing features to the subset. Furthermore, simply the completion of the search can act as stopping criterion.

Above, the general procedure for feature selection was presented. Since the existing feature selection algorithms are diverse, some form of classification is in order. Feature selection models can be categorized in a few different ways: first of all, algorithms can be either supervised or unsupervised depending on whether they need human input as labels or not. Secondly, feature selection algorithms can be categorized based on how they perform the feature selection itself: there are three main types that are called wrappers, filters, and embedded methods (Li et al., 2017). Additionally, models can be categorized as either linear or nonlinear.

Selecting an algorithm for feature selection is highly dependent on the data. If the data is simple, a basic linear model often suffices. However, data is often complex and therefore a nonlinear model will most likely yield better results (Guyon and Elisseeff, 2003). If there is some additional reference or target data available, a supervised model may be used. Otherwise, one needs to use an unsupervised model. Many models contain a number of different parameters, some more important than others. Without domain knowledge, (hyper)parameter tuning is often appropriate. It involves running the model alternating values given to parameters to find the ones yielding the best performance. Additionally, for the evaluation of the success of feature selection, either unsupervised or supervised methods can be used. They are discussed in Section 3.3.

3.1.1 Main types of feature selection methods

A wrapper method iteratively selects a subset of features and evaluates their quality until, e.g., the highest learning performance is achieved. Wrapper methods rely on a separate learning algorithm to handle the evaluation, the choice of which depends on the task (Jović et al., 2015). In classification tasks, the wrapper includes a classifier such as a support vector machine (SVM), while in clustering tasks the learning algorithm is a clustering algorithm such as the K-means. In theory, any algorithm can be used in a wrapper, but because of the computational limitations, only efficient algorithms such as linear SVM are used in practice. However, wrapper methods are quite inefficient due to their search space being 2^m, where m is the number of features in the search space (Li et al., 2017). Additionally, wrappers produce good results only when all the following conditions are fulfilled: 1) there are few relevant features all of them being available, 2) appropriate learning algorithm is used, and 3) the data is mostly noise-free (Huang, 2015).

Unlike in wrappers, filter methods do not involve any learning algorithm. Instead, they rely on characteristics of data in order to determine the importance of a feature (Li et al., 2017). In filter methods, features are first ranked by importance and then the ones with low importance are filtered out. Filter methods are generally more efficient than wrapper methods but the selected features are not necessarily ideal for target learning algorithms because of the unguided feature selection. They can miss features that are useful with others but not on their own (Kumar and Minz, 2014).

There are two types of feature ranking algorithms. Univariate filters analyze each feature at a time determining their dependence on the target, while multivariate filters consider all features simultaneously (Huang, 2015). Univariate filters have low computational complexity, only $O(m)$, while some multivariate filters use an exhaustive search strategy, thus being computationally inefficient.

Filter methods can use either linear or nonlinear ranking criteria. A widely popular example of a linear ranking criterion used in feature selection is the Pearson's correlation coefficient (PCC) (Pearson, 1895). For each feature vector $\mathbf{x}_i$ of length n, its linear correlation with the target vector $\mathbf{y}$ is computed. PCC is defined by the covariance of the two variables divided by their standard deviations $\sigma_{\mathbf{x}_i}$ and $\sigma_{\mathbf{y}}$, i.e.,

$$R_{Pearson}(\mathbf{x}_i, \mathbf{y}) = \frac{cov(\mathbf{x}_i, \mathbf{y})}{\sigma_{\mathbf{x}_i}\sigma_{\mathbf{y}}}. \tag{1}$$

In the equation, covariance is defined by

$$cov(\mathbf{x}_i, \mathbf{y}) = \sum_{k=1}^{n}(x_k - \bar{\mathbf{x}}_i)(y_k - \bar{\mathbf{y}}), \tag{2}$$

where x_k and y_k are the kth elements in feature vector $\mathbf{x}_i$ and target vector $\mathbf{y}$, respectively, and $\bar{\mathbf{x}}_i$ and $\bar{\mathbf{y}}$ are the means of $\mathbf{x}_i$ and $\mathbf{y}$, respectively. The computed correlation coefficients can be used to simply select, for example, the l best features from the data.

Sometimes, linear feature selection methods result in poor performance. This is especially true with complex data as the variables may contain nonlinear dependencies that the linear methods are not able to grasp. There are some nonlinear ranking criteria that can be used in filter-type feature selection. Mutual information (MI) is an example of such criterion. It measures the mutual dependence between two variables using the probability densities $p(\mathbf{x}_i)$ and $p(\mathbf{y})$ and the joint density $p(\mathbf{x}_i, \mathbf{y})$ and is defined as

$$I(i) = \int_{\mathbf{x}_i}\int_{\mathbf{y}} p(\mathbf{x}_i, \mathbf{y}) log \frac{p(\mathbf{x}_i, \mathbf{y})}{p(\mathbf{x}_i)p(\mathbf{y})} dxdy. \tag{3}$$

However, the MI has one major downside: the probability densities are unknown. They need to be estimated from the data which is rather difficult (Guyon and Elisseeff, 2003).

While in wrappers, generating each feature subset is followed by its evaluation by a separate learning algorithm, in embedded methods, feature selection is incorporated in the evaluation algorithm, i.e., feature subset selection and evaluation are performed concurrently (Jović et al., 2015). Embedded methods combine the good qualities of the two: a learning algorithm is involved in feature selection but the evaluation is not performed iteratively. According to Li et al. (2017), the most popular embedded methods are the regularization models. They fit a learning model by minimizing training errors using a regularization term of which the L_1 norm is one of the most commonly used. An example of a model that uses the L_1 norm for regularization with a squared loss is the least absolute shrinkage and selection operator (LASSO) (Tibshirani, 1996) algorithm. Other examples of algorithms that are used as embedded methods in feature selection are, for example, decision tree algorithms, such as random forest, and multinomial logistic regression.

A challenge with many feature subset selection methods is that they are sensitive to minor changes in the conditions (Guyon and Elisseeff, 2003): having redundant variables in the data may lead to different subsets being selected if the initial conditions of the algorithm are slightly altered or data is manipulated by adding noise, or adding or removing a small number of features. Variable selection can be stabilized using bootstraps, i.e. performing feature selection on sub-samples of data until the final subset is found.

3.1.2 Unsupervised feature selection

Everything discussed in this section so far has been about supervised feature selection as supervised methods as much more common. However, there are some unsupervised methods as well. There are at least two advantages in unsupervised feature selection compared to its supervised counterpart: the methods are unbiased and have good performance without prior knowledge, and they have smaller risk of overfitting (Solorio-Fernández et al., 2020). However, according to Dy and Brodley (2004), unsupervised feature selection is significantly harder problem than supervised since defining the relevancy of features is not an easy task. Solorio-Fernández et al. (2020) lists three criteria that are used for determining the relevancy: selecting features that best preserve the manifold structure of the original data, finding cluster indicators with clustering algorithms and then converting the problem into a supervised one, or determining the dependencies between the features and minimizing or maximizing the correlation in the final feature subset.

Similarly to supervised feature selection, unsupervised feature selection algorithms can be divided into three categories: filters, wrappers, and hybrid methods (Solorio-Fernández et al., 2020). The classification corresponds to filters, wrappers, and embedded methods of supervised feature selection for the most part. Like in supervised methods, filters can be either uni- or multivariate. Univariate filters rank features by evaluating them one by one using some criterion. Univariate filter methods can filter out irrelevant, but not redundant features. Multivariate filters, on the other hand, can remove both irrelevant and redundant features as they take into account any dependencies between the features. Solorio-Fernández et al. (2020) divide univariate filters into two and multivariate into three different groups in their taxonomy of unsupervised feature selection methods. Univariate filters are either information or spectral-similarity based methods. In the former, the degree of dispersion of the data is evaluated using measures like entropy divergence and MI, while in the latter, the local or global data structure is modeled using Laplacian matrices.

Multivariate filters can be statistical/information based, bio-inspired, or spectral/sparse learning methods (Solorio-Fernández et al., 2020). Filters of the first group use measures like variance-covariance, linear correlation, or MI in feature selection. Bio-inspired methods use stochastic search strategies that are based on the swarm intelligence paradigm by Beni and Wang (1993) for finding a good feature subset. The third group, i.e. spectral/sparse learning methods, are based on, as the name suggests, spectral analysis and sparse learning.

Solorio-Fernández et al. (2020) divides unsupervised feature selection wrapper methods into three groups: sequential, bio-inspired, and iterative. In sequential wrappers, features are added or removed sequentially. Bio-inspired wrappers use randomness in the search to avoid local optima while iterative wrappers instead of combinatorial search, use estimation. Hybrid wrappers consist of a filter and a wrapper stage. In the former, features are ranked or selected, while in the latter, the feature subset is evaluated using a clustering algorithm. For a throughout walkthrough and many examples of unsupervised feature selection algorithms, refer to Solorio-Fernández et al. (2020).

3.2 Kernel-based feature selection

A kernel is a function $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ that equals to an inner product of vectors $\phi(x)$ and $\phi(x')$ (Mohri et al., 2018):

$$\forall x, x' \in \mathcal{X}, \kappa(x, x') = \langle \phi(x), \phi(x') \rangle \tag{4}$$

where $\phi : \mathcal{X} \to \mathcal{H}$ is a mapping of points x and x' from the input space $\mathcal{X}$ to a (usually) higher-dimensional space $\mathcal{H}$ called reproducing kernel Hilbert space (RKHS), informally feature space. Kernels are often used in computing for their efficiency which is due to the fact that it is often considerably more efficient to evaluate the kernel function in the input space than to compute the inner product directly in the feature space, which, in the worst case scenario, could be even an infinite-dimensional. This is the so-called kernel trick. Kernels can be used when there is an inner product defined between the input variables, typically data points, which makes their use so convenient. Another advantage kernels is their flexibility as κ can be arbitrarily chosen. Kernels are also used for including nonlinearity to algorithms through the use of nonlinear kernels.

While the kernel methods are not prevalent in feature selection, a few algorithms have been developed recently (Brouard et al., 2022; Szedmak et al., 2023). Advantages of kernel methods in feature selection is that they 1) enable nonlinear feature selection, 2) can handle multiple types of data from the same individuals, and 3) allow to work with very high dimensional data, at least in theory. Brouard et al. (2022) proposed two algorithms: one suitable for unsupervised (unsupervised kernel feature selection UKFS) and other for supervised feature selection in multi-output case (kernel output kernel feature selection KOKFS). They can be considered to belong to embedded feature selection methods for their use of regularization. For an input data matrix $\mathbf{X} \in \mathbb{R}^{n \times m}$, where n is the number of rows, or samples, and m the number of columns, or features, the kernel is computed on samples. While the computation itself is performed in the feature space, the selected features are from the original input space. A major drawback of these two algorithms, and others that compute kernels on samples, is that with big data, which here refers to data with a huge n, the algorithms do not scale well as the computed kernels are huge having dimensions $\mathbb{R}^{n \times n}$. In addition, the two algorithms operate so that they over and over again first perform calculation in the feature space, then return to the input space to select the feature corresponding to the result before continuing computation again in the feature space. This jumping back and forth naturally causes computational limitations to the number of samples in the input data.

There are many kernel functions that can be used in kernel-based feature selection. The following ones, also used in this thesis, are some of the most commonly used: linear, polynomial, radial basis function (RBF) and Tanimoto kernels. In feature selection setting, the first of them is, as the name suggests, suitable for linear selection while polynomial and RBF kernels are appropriate for nonlinear feature selection. In the following equations, $\mathbf{x}, \mathbf{x}'$ can be either data sample ($\mathbf{x}, \mathbf{x}' \in \mathbb{R}^m$) or feature vectors ($\mathbf{x}, \mathbf{x}' \in \mathbb{R}^n$).

Linear kernel is defined simply by the dot product of $\mathbf{x}$ and $\mathbf{x}'$, i.e.,

$$\kappa_{linear}(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T \cdot \mathbf{x}'. \tag{5}$$

Linear kernel is different from other kernels in the sense that data points are not mapped to a higher-dimensional feature space. Instead, the dimensionality of the feature space equals to the dimension of the input space. Thus, using linear kernel in a kernel-based feature selection method does likely not yield any better subset of features than some other non-kernel-based linear feature selection algorithm would.

Polynomial kernel extends linear kernel by adding a constant $c > 0$ and a degree $d_{poly} \in \mathbb{N}$ and is defined over $\mathbb{R}^d$, where $d \in \{m, n\}$ depending on whether the inputs are data sample or feature vectors, by

$$\kappa_{polynomial}(\mathbf{x}, \mathbf{x}') = (\mathbf{x} \cdot \mathbf{x}' + c)^{d_{poly}}. \tag{6}$$

Polynomial kernel maps the input space to a feature space of dimension $\binom{d+d_{poly}}{d_{poly}}$. Commonly used values for the two parameters are $c = 1$ and $d_{poly} = 3$, which are also used in the experiments of this work.

RBF, or a Gaussian kernel is another nonlinear kernel and is defined over $\mathbb{R}^d$ by

$$\kappa_{RBF}(\mathbf{x}, \mathbf{x}') = exp(-\gamma \|\mathbf{x} - \mathbf{x}'\|^2), \tag{7}$$

where $\|\mathbf{x} - \mathbf{x}'\|^2$ is the squared Euclidean distance between $\mathbf{x}$ and $\mathbf{x}'$, and $\gamma = \frac{1}{2\sigma^2}$ is a parameter that controls the width of the RBF kernel. As noted by Mohri et al. (2018), Gaussian kernels are positive linear combinations of polynomial kernels.

Tanimoto kernel, first proposed by Ralaivola et al. (2005), is specifically developed for analyzing molecular fingerprints as already discussed in Section 2. The kernel is defined by

$$\kappa_{Tanimoto}(\mathbf{u}, \mathbf{v}) = \frac{|\mathbf{u} \cap \mathbf{v}|}{|\mathbf{u} \cup \mathbf{v}|}, \tag{8}$$

where $|\mathbf{u} \cap \mathbf{v}|$ and $|\mathbf{u} \cup \mathbf{v}|$ are the number of elements in the intersection and union of the fingerprint vectors $\mathbf{u}$ and $\mathbf{v}$, respectively. Although Tanimoto kernel is originally designed for fingerprint data, it can be generalized to any type of vectors, see Szedmak and Bach (2020).

3.2.1 Projective selection

Although kernels are generally considered efficient in introducing nonlinearities to algorithms, when dealing with large data matrices, computing kernels can be extremely time and resource consuming, as mentioned above. Traditionally, for any data matrix $\mathbf{X} \in \mathbb{R}^{n \times m}$ where samples are on rows and features on columns, kernels are computed row-wise, i.e., the similarity between each pair of rows is calculated. Thus, a kernel matrix for two data matrices $\mathbf{X}_1 \in \mathbb{R}^{n_{\mathbf{X}_1} \times m}$ and $\mathbf{X}_2 \in \mathbb{R}^{n_{\mathbf{X}_2} \times m}$ has the dimension $\mathbb{R}^{n_{\mathbf{X}_1} \times n_{\mathbf{X}_2}}$, where $n_{\mathbf{X}_1}$ and $n_{\mathbf{X}_2}$ are the number of rows in matrices $\mathbf{X}_1$ and $\mathbf{X}_2$, respectively, and m is the number of columns in both matrices. As typically $n_{\mathbf{X}_1}, n_{\mathbf{X}_2} \gg m$, the resulting kernel matrix is huge.

However, a recent idea is to kernelize the variables so that instead of rows, a similarity is computed between each pair of column vectors (Szedmak et al., 2023). Then, for two input data matrices $\mathbf{X}_1$ and $\mathbf{X}_2$, the dimension of the resulting kernel matrix is just $\mathbb{R}^{m\times m}$ which is significantly more efficient to compute. Since the kernels are computed on columns, the features can be directly selected in the feature space without the need to return to the input space. These two things enable fast and highly scalable feature selection. In their paper, Szedmak et al. (2023) present a supervised kernel-based feature selection algorithm ProjSe that uses the kernelization of features this way.

In ProjSe, features are selected iteratively in the feature space using projection operators that guarantee the following: at iteration t, a feature vector $\mathbf{x}_i \in \mathbb{R}^n$ from $\tilde{\mathbf{X}}_t \in \mathbb{R}^{n\times(m_x-t+1)}$, where n and $m_x - t + 1$ are the number of row and column vectors in $\tilde{\mathbf{X}}_t$, respectively, and $\tilde{\mathbf{X}}_t$ is the input matrix from which already selected features are removed, is selected so it has as high correlation as possible to the features in the reference data matrix $\mathbf{Y} \in \mathbb{R}^{n\times m_y}$, where m_y is the number of features in $\mathbf{Y}$, and has as small correlation to previously selected features as possible. The reference features can be other data-views, output labels, or regression targets.

The logic in the algorithm is approximately as follows: correlation between an input feature vector and the output variables is calculated, i.e.

$$corr(\mathbf{x}_i, \mathbf{Y}) = \left\| \mathbf{P}_{\mathcal{L}_Y} \frac{\mathbf{x}_i}{\|\mathbf{x}_i\|} \right\| = \left\langle \mathbf{P}_{\mathcal{L}_Y} \frac{\mathbf{x}_i}{\|\mathbf{x}_i\|}, \mathbf{P}_{\mathcal{L}_Y} \frac{\mathbf{x}_i}{\|\mathbf{x}_i\|} \right\rangle^{\frac{1}{2}} = \left\langle \frac{\mathbf{x}_i}{\|\mathbf{x}_i\|}, \mathbf{P}_{\mathcal{L}_Y} \frac{\mathbf{x}_i}{\|\mathbf{x}_i\|} \right\rangle^{\frac{1}{2}}, \tag{9}$$

where $\mathbf{P}_{\mathcal{L}_Y}$ is the orthogonal projection operator into subspace spanned by the columns of $\mathbf{Y}$, $\mathcal{L}_Y$. This correlation is computed for each feature vector $\mathbf{x}_i$. $\mathbf{P}_{\mathcal{L}_Y}$ can be written as

$$\mathbf{P}_{\mathcal{L}_Y} = \mathbf{Y}(\mathbf{Y}^\top\mathbf{Y})^+\mathbf{Y}^\top, \tag{10}$$

where $\mathbf{Y}^\top$ is the transpose of matrix $\mathbf{Y}$ and $^+$ is the pseudoinverse operator. Applying singular value decomposition (SVD) on $\mathbf{Y}$ yields

$$\mathbf{P}_{\mathcal{L}_Y} = \mathbf{Y}(\mathbf{Y}^\top\mathbf{Y})^+\mathbf{Y}^\top = \mathbf{Y}\mathbf{Y}^+ = \mathbf{U}_Y\mathbf{S}_Y\mathbf{V}_Y^\top\mathbf{V}_Y\mathbf{S}_Y^+\mathbf{U}_Y^\top = \mathbf{U}_Y\mathbf{U}_Y^\top, \tag{11}$$

where $\mathbf{U}_Y$ is the $n \times n$ left unitary matrix, $\mathbf{S}_Y$ is the $n \times m$ diagonal matrix, and $\mathbf{V}_Y$ is the complex conjugate of the $m \times m$ right unitary matrix. Inserting this result into Eq. 9 gives

$$corr(\mathbf{x}_i, \mathbf{Y}) = \left\langle \frac{\mathbf{x}_i}{\|\mathbf{x}_i\|}, \mathbf{U}_Y\mathbf{U}_Y^\top \frac{\mathbf{x}_i}{\|\mathbf{x}_i\|} \right\rangle^{\frac{1}{2}} = \left\langle \mathbf{U}_Y^\top \frac{\mathbf{x}_i}{\|\mathbf{x}_i\|}, \mathbf{U}_Y^\top \frac{\mathbf{x}_i}{\|\mathbf{x}_i\|} \right\rangle^{\frac{1}{2}} = \left\| \mathbf{U}_Y^\top \frac{\mathbf{x}_i}{\|\mathbf{x}_i\|} \right\|. \tag{12}$$

By replacing $\mathbf{x}_i$ with its feature representation $\phi(\mathbf{x}_i)$, where ϕ is the feature mapping associated with a kernel, in Eq. 12, at every iteration t, the algorithm computes $\left\| \mathbf{U}_Y^\top \frac{\phi(\mathbf{x}_i)}{\|\phi(\mathbf{x}_i)\|} \right\|$. After careful manipulation, it turns out that

$$\mathbf{U}^\top\phi(\mathbf{x}_i) = \mathbf{S}^+\mathbf{V}^\top\kappa(\mathbf{Y}, \mathbf{x}_i), \tag{13}$$

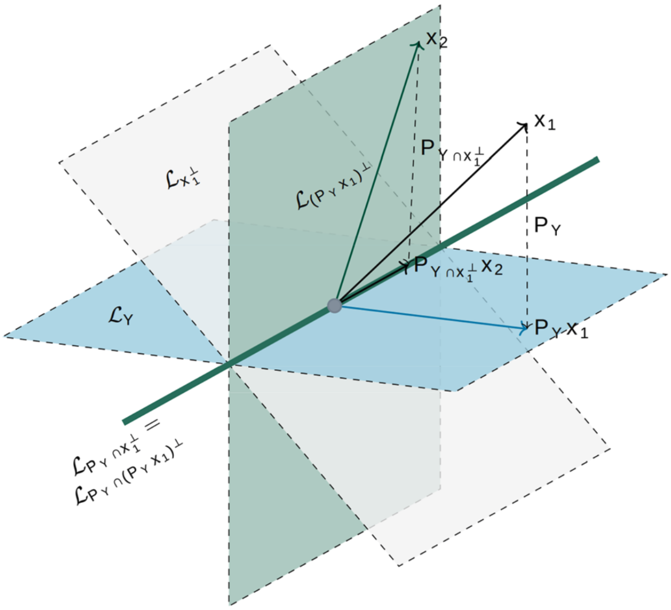


Figure 5: Projection steps in ProjSe algorithm as illustrated in Szedmak et al. (2023), Fig. 1.

where $\kappa(\mathbf{Y}, \mathbf{x}_i)$ is a kernel evaluated between the features in $\mathbf{Y}$ and $\mathbf{x}_i$. In ProjSe algorithm, any kernel can be used in practice. Depending on the choice of kernel, ProjSe can perform either linear or nonlinear feature selection. In the experiments section, the four kernels presented in Section 3.2 are used.

According to Szedmak et al. (2023), ProjSe scales to very large datasets, by which they refer to datasets where number of samples n exceed one million. This is possible due to the way the variables are kernelized as discussed above. However, ProjSe is not suitable for very fat data, i.e. data where $n \ll m$. When m is very large, the very same problem as with traditional kernel methods arises, i.e. the resulting kernel matrix is huge and the computation becomes practically unfeasible in a sensible time and with reasonable resources.

The main steps of the algorithm are illustrated in Figure 5. First, a feature $\mathbf{x}_1$ is selected so that it has the largest correlation with the output features $\mathbf{Y}$ when projected as $\mathbf{P}_Y\mathbf{x}_1$ onto the subspace spanned by columns of $\mathbf{Y}$, i.e., $\mathcal{L}_Y$. Then, a second feature $\mathbf{x}_2$ is selected so that it has the largest correlation possible when projected onto the intersection of $\mathcal{L}_Y$ and the orthogonal complement of previously chosen variables $\mathbf{x}_1$, i.e., $\mathbf{P}_{\mathcal{L}_Y \cap \mathcal{L}_{\mathbf{x}_1^\perp}}$. It is noteworthy that the projected feature $\mathbf{P}_{\mathbf{Y} \cap \mathbf{x}_1^\perp}\mathbf{x}_2$ also lies in the intersection of $\mathcal{L}_Y$ and $\mathcal{L}_{(\mathbf{P}_Y\mathbf{x}_1)^\perp}$ which enables the efficiency of the algorithm. Details can be found in Szedmak et al. (2023).

3.3 Evaluating feature selection

When using feature selection for dimensionality reduction, it is essential to assess whether it has been successful or not. Unsuccessful feature selection means that significant feature subset is not found and relevant information is lost. Then, it is very likely to lead to poor performance in the succeeding steps of the workflow. Ways to evaluate feature selection can roughly be divided into unsupervised and supervised methods.

3.3.1 Unsupervised methods

As appropriate labeled data is not always necessarily available, one must sometimes get by with the features only: when evaluating the success of feature selection in unsupervised manner, only the features themselves are used. While it is possible to perform the evaluation independently of any subsequent task, it is much more common to pass the selected feature subset to some learning algorithm. For example, in their paper, Dy and Brodley (2004) perform Expectation-Maximization clustering on the selected feature subset and then use scatter separability and maximum likelihood as two performance criteria to evaluate the selected features.

However, there are some methods for performing "independent" evaluation. One way is to determine how redundant the selected features are (Solorio-Fernández et al., 2020). This can be done in two ways: either determine the redundancy of each selected feature in respect to other features, or determine how redundant the selected features are in some subsequent task. There are various criteria that can be used for measuring the redundancy of selected features: degree of dependence, similarity, association and correlation. Furthermore, the stability of the selected features can be inspected. This refers to observing how often same features are selected when the selection algorithm is run multiple times, while each time using a different subsample of data (Meinshausen and Bühlmann, 2010). If the same features are selected often, they are probably at least somewhat significant.

In this thesis, kernel alignment is used as an unsupervised method to evaluate the success of feature selection. While using kernel alignment for this purpose is not common, the logic behind this choice is the following: kernel alignment describes similarity between data points in kernels. If feature selection produces such feature subset that yields high kernel alignment when compared to a kernel evaluated with full features, it signifies that the feature selection has found features that maintain this similarity. Then, it is reasonable to assume that the selected feature subset describes the full data set, at least to some extent, as it has managed to find somehow relevant features and not just random noise. Therefore, the feature selection can be considered successful.

The first step in kernel alignment is to compute kernels both of the data matrix with the m selected features and of the original data matrix with all features included. Kernel can be chosen freely: since this thesis deals with molecular fingerprint data, the Tanimoto kernel is used. Kernel alignment is defined by

$$ka(\kappa, \kappa') = \frac{\langle \kappa_c, \kappa'_c \rangle_F}{\|\kappa_c\|_F \, \|\kappa'_c\|_F}, \tag{14}$$

where $\kappa \in \mathbb{R}^{n \times n}$ and $\kappa' \in \mathbb{R}^{n \times n}$ are two kernel matrices, $\langle \kappa_c, \kappa'_c \rangle_F$ is their Frobenius inner product and $\|\kappa_c\|_F$ and $\left\|\kappa'_c\right\|_F$ are their Frobenius norms (Cortes et al., 2010). Subscript c refers to the kernel matrices being centered. The kernel alignment outputs a value in range $[0, 1]$. The closer the value is to one, the more similar the two kernels are. Computing kernel alignment has time complexity $O(n^2)$, where n is the number of rows in both kernel matrices. Kernel alignment is suitable for evaluating the similarity of smaller kernel matrices, but the quadratic time complexity makes it unfeasible with huge kernels.

3.3.2 Supervised methods

It may be that supervised methods are better for evaluating feature selection since they incorporate more data into the task in form of labels. They can be used as prediction targets: a model is taught to map input data to the labels. Depending on the labels, two types of algorithms can be used: classifiers if the labels are discrete and regressors if continuous. The evaluation is performed so that a supervised learning algorithm is trained both with full data and data that contains only the features chosen in feature selection. If the chosen evaluation metric yields almost as high value for the feature subset as for the full data, feature selection can be considered successful.

Since the prediction targets used in the experiments in this thesis are continuous, regression is used for supervised evaluation of feature selection. In short, regression is a task where real-valued labels are predicted from data. There is a plethora of different models for this kind of task. In the experiments, a few of them are used to evaluate the success of feature selection by using drug response data as regression targets. Simultaneously, it is investigated whether it is possible to predict accurately how much different anticancer drugs inhibit tumor cell growth using molecular fingerprints as the input data.

The simplest regression model is the linear regression which is defined as

$$y_i = a\mathbf{x}^i + b, \tag{15}$$

where $\mathbf{x}^i$ is an input data sample vector, y_i is a regression target, and a and b are the slope and intercept. Linear regression estimates the linear relationship between the inputs and the output. The model is typically fitted using the least squares method. Linear regression is an efficient algorithm but it does not generalize well, as it simply minimizes the empirical error without using any sort of regularization (Mohri et al., 2018).

Since it is expected that the linear regression struggles with the (mostly) binary molecular fingerprint data, a nonlinear regression model is used as well for comparison. The model is random forest (Breiman, 2001), which is an ensemble learning method based on decision trees. It creates a controllable number of trees and trains them with data created by bootstrapping, i.e. taking random subsets of the original data with replacement. Each tree is trained with different subset which makes the random forest algorithm less sensitive to the training data than a single decision tree. The results from each decision tree are aggregated and averaged to generate a prediction output. The algorithm is illustrated in Figure 6.

The two above mentioned regressors are used throughout the prediction experiments, but additionally, some other models are experimented with: a support vector regressor (SVR) and multilayer perceptrons (MLP). SVR is based on the SVM

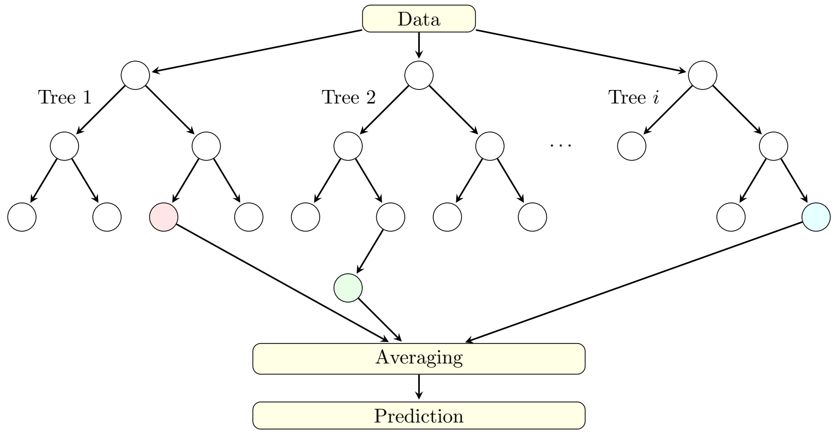


Figure 6: An illustration of the random forest algorithm used for regression with i trees in the forest.

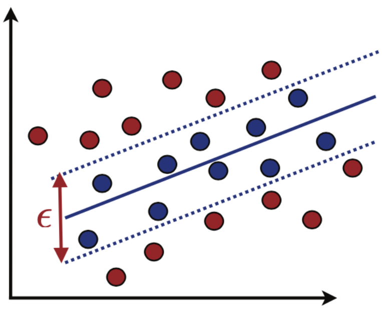


Figure 7: SVR fits two (dotted) lines at ϵ apart to the data. Data points (blue) inside the area between the two lines are not penalized while the outliers (red) are. The image is adapted from Mohri et al. (2018).

algorithm, and the idea in the algorithm is to fit two lines at ϵ apart to the data, see Figure 7 (Mohri et al., 2018). The data points that fall outside the area between the lines are penalized as a function of distance. The choice of parameter ϵ is a trade-off: a large ϵ includes more data points inside the so-called "ϵ-tube" but may decrease the prediction accuracy. Using epsilon-insensitive loss, or its squared counterpart, as a loss function in the SVR helps to prevent the model from overfitting to small noise around the ϵ-tube.

MLP, on the other hand, is a feedforward artificial neural network (ANN). Feedforward refers to the architecture of the ANN having no cycles or loops. MLPs consist of an input layer, at least one fully connected hidden layer, and an output layer. A fully connected layer is a layer in which every neuron is connected to every neuron in the next layer with a certain weight. The hidden layers have nonlinear activation functions to enable modeling complex data. Some typical activation functions are sigmoid, hyperbolic tangent (tanh) and rectified linear unit (ReLU) (Fukushima, 1969). The

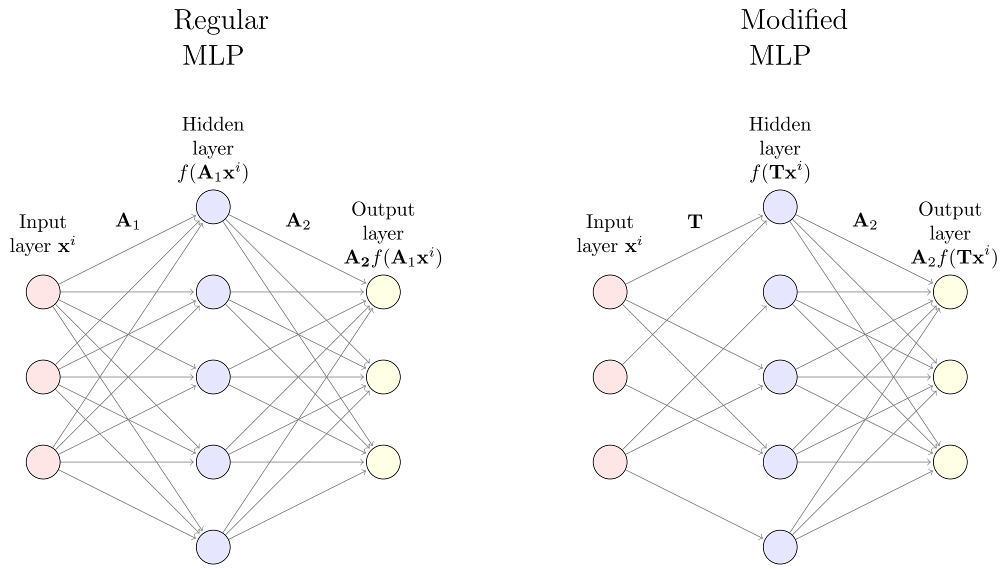


Figure 8: Two MLPs used in the experiments, both with a single large hidden layer with $5m$ neurons, where the number of features m depends on the input data, followed by a ReLU activation function f.

one used in MLPs in the experiments of this thesis is the ReLU, and it is defined as

$$f(x_a) = \max(0, x_a) = \begin{cases} x_a & \text{if } x_a > 0 \\ 0 & \text{if } x_a \leq 0, \end{cases} \tag{16}$$

where x_a is the pre-activation value.

The two slightly different architectures that are in the experiments are illustrated in Figure 8. The regular MLP is fully connected. The matrices $\mathbf{A}_1$ and $\mathbf{A}_2$ apply weights to the features passing through the network. The weights are learned during the training of the MLP via a process called backpropagation (Linnainmaa, 1970). In backpropagation, the gradient of the network is computed with respect to a loss function such as the mean squared error (MSE).

In the modified MLP, there is no parameter learning between the input and hidden layers as the weight matrix $\mathbf{A}_1$ is replaced with a transformation matrix $\mathbf{T}$. It is a sparse matrix containing only values $\{-1, 0, +1\}$ which are given uniformly at random as weights to the features in the input data vector $\mathbf{x}^i$ passing through to the hidden layer. Since some features will get the weight zero, they do not pass through to a neuron in the hidden layer. Thus, the MLP is not fully connected and, in fact, it performs feature subset selection. After applying the ReLU activation function f, the matrix $\mathbf{A}_2$ gives a weight to each of the outputs of the hidden layer. These weight are learned in the training process. In the experiments, the vector $\mathbf{A}_2$ is regularized with L_1 norm (LASSO) in order to decrease the number of inputs to the output layer to avoid overfitting and improve generalization of the MLP.

4 Experiments

In this thesis, the experiments are divided into two pipelines: one evaluating the success of feature selection using an unsupervised method, and other using a supervised one. As discussed before, the unsupervised method used is the kernel alignment, which describes how well the similarity between data points is preserved when a subset is selected from the full set of features. Regression, as the supervised method, is used both for predicting the PTC values from molecular fingerprint data with full set of features, and examining how close to those results can be reached by using a feature subset.

In this section, the workflow of the experiments conducted is explained. The methods used for preprocessing the drug response data are discussed, as well as the two feature selection methods, ProjSe and Pearson correlation coefficient-based variable ranking, and how their performance is evaluated.

4.1 Outline

The workflow as illustrated in Figure 9 is as follows: spectral clustering is performed on a similarity matrix containing normalized pairwise counts of tumor cell lines, i.e. the number of common drugs for each cell line pair in the NCI-60 data in order to find as many tumor cell lines as possible on which a sufficiently large number of drugs is tested. Using the number of clusters $n_{cluster} = 20$ results in 22 tumor cell lines on which $n = 31549$ drugs has been tested, see Figure 10. The cell lines are described in Table 2. To have labels $\mathbf{y}$ for regression, PTC values for drugs tested on tumor cells in A549/ATCC cell line at the highest concentration available are taken aside. The remaining drug response data, i.e., PTC values measured at the highest dose concentration for each drug tested on tumor cells in the other 21 cell lines act as the reference data $\mathbf{Y}$ in feature selection. The justification for taking PTC values measured particularly at the highest concentration available is based on the assumption that the dose-response relation follows a S-shaped (sigmoidal) curve in which the larger the dosage, the stronger the response (Coleman and Marks, 1998). It is assumed that the

Cell line	Cancer type
A549/ATCC	Non-Small Cell Lung
NCI-H23	Non-Small Cell Lung
NCI-H460	Non-Small Cell Lung
COLO 205	Colon
HCT-116	Colon
HCT-15	Colon
HT29	Colon
SW-620	Colon
KM12	Colon
IGROV1	Ovarian
OVCAR-4	Ovarian

Cell line	Cancer type
OVCAR-8	Ovarian
K-562	Leukemia
MOLT-4	Leukemia
SK-MEL-28	Melanoma
SK-MEL-5	Melanoma
UACC-257	Melanoma
SNB-19	CNS
SF-268	CNS
U251	CNS
SN12C	Renal
UO-31	Renal

Table 2: Human tumor cell lines from the NCI-60 dataset used in the experiments and their cancer types. CNS refers to central nervous system and non-small cell lung cancer is the more common of two main lung cancer types.

largest dosage for each drug is approximately at the same location on the dose-response curve, i.e. the response to the drug is at its maximum, which may not be the case. Other way to select drug dosage could be, for example, to select the half maximal effective concentration (EC_{50}) for each drug, but the problem is just that the location of the curves may wary. So for some drug, the response may already be at maximum while for some other drug a higher dosage is needed for any response at all. It is noteworthy that the highest concentration at which the different drugs in the NCI-60 dataset are tested varies. This is natural as appropriate dosage is not the same for every drug. However, since the dosage itself is not of interest in this thesis, the differences in drug concentrations are disregarded and the PTC values are used as reference data in feature selection as discussed. As the number of features in the reference data $\mathbf{Y}$ is quite low, two different methods to increase its size are used: either take PTC values measured at the several different concentrations, or put the PTC values into 10 categories and then transform them into, for example, 10-bit vectors. More of this in Section 4.2.

After preprocessing the reference data $\mathbf{Y}$, a 75%-25% train-test split is performed on the fingerprint data $\mathbf{X}$, the reference data $\mathbf{Y}$ and the regression targets $\mathbf{y}$. Using the reference training data $\mathbf{Y}_{train} \in \mathbb{R}^{n_{train} \times m_y} \subseteq \mathbf{Y}$ in both the feature ranking filter-type method and ProjSe, $m = \{21, \ldots, 105\}$ features, depending on the pipeline, are selected from the fingerprint training and testing data sets $\mathbf{X}_{train} \in \mathbb{R}^{n_{train} \times m}$ and $\mathbf{X}_{test} \in \mathbb{R}^{n_{test} \times m}$ where $\mathbf{X}_{train}, \mathbf{X}_{test} \subseteq \mathbf{X}$. Since ProjSe can take in the whole matrix $\mathbf{Y}_{train}$ as reference data, the algorithm is run only once per kernel, but for fair comparison, the other method requires multiple runs: it is a regression based algorithm that takes in a 1D vector as regression targets. In this feature selection method, both the input fingerprint data and the regression targets centered. The algorithm is used so that it performs feature selection using PTC values from tests on one of the 21 tumor cell at a time as regression targets. The algorithm yields a Pearson correlation coefficient (PCC) for each feature: the mean of 21 runs is computed to average out the differences between the responses in different tumor cell lines. Based on the mean PCC values for each feature, the l ($\leq m$) best can be selected. ProjSe is run with the four (linear, RBF, polynomial and Tanimoto) kernels presented in Section 3.2, with both the input data $\mathbf{X}_{train}$ and reference data $\mathbf{Y}_{train}$ centered and normalized. After performing feature selection on the training data, the corresponding features are then selected from the test data $\mathbf{X}_{test}$. To evaluate the success of the two feature selection methods, both the kernel alignment and different regression models are used.

In unsupervised evaluation of feature selection, to assess how well the selected subset of variables match to the full fingerprints, kernels both on samples of the fingerprint data with full features, and samples with m features are computed. The choice of kernel is straightforward as the Tanimoto kernel is development specifically for analyzing molecular fingerprint data. It should be noted that as one of the fingerprints, ErG, consists of non-binary data, a generalized version of the Tanimoto kernel must be used with these fingerprints. As a supervised method, different regression models are used in predicting PTC values $\mathbf{y}$ from the various fingerprint data $\mathbf{X}$. The models are trained using the $\mathbf{X}_{train}$ and $\mathbf{y}_{train}$, and tested with $\mathbf{X}_{test}$ and $\mathbf{y}_{test}$. The success of feature selection can be evaluated by comparing the resulting (Spearman) correlations between predicted values and groudtruth labels produced by fingerprint feature subsets vs. full features.

4.2 Manipulating reference data $\mathbf{Y}$

As mentioned above, when using only the PTC values measured at the highest dose concentration for each drug, the resulting reference data matrix $\mathbf{Y}$ has only 21 features whereas the fingerprints contain hundreds of features. Such few features are likely not enough to yield good performance in feature selection. Therefore, two different methods for increasing the size of $\mathbf{Y}$ are used. The first one is very straightforward: since the NCI-60 dataset contains measurements for each drug at multiple different concentrations, the number of features can be increased to k-fold simply by taking measurements at k concentrations. There are limits to this however: the number of different dose concentrations at which each drug is tested varies and furthermore, the dosages are not the same for every drug. The number of drugs for which there is response values measured at least at five different dosages is sufficient (29 900), but drops drastically after that. Therefore, including PTC values for drugs tested at the five different dose concentrations, the resulting reference matrix $\mathbf{Y}$ has $n = 29000$ rows (drugs) and $m_y = 105$ columns (features).

As was discussed in Section 4.1, the assumption is that the dose-response relation of the anti-cancer drugs follows a sigmoidal curve. Since the location of the curve for most of the drugs varies, the responses at the five different concentrations are unlikely to correspond very well between different drugs. However, since the number of different dosages the drugs are tested at is five for most drugs, it is assumed here that the measurements cover the dose-response curve to large extent. And, since the response values are only used to increase the size of the reference data matrix $\mathbf{Y}$, their exact correspondence should not be essential.

The other method to grow the size of $\mathbf{Y}$ is slightly more complicated. The PTC values measured at one concentration are categorized by placing them into ten percentiles. The categories are them transformed into 10-bit vectors in which each bit represents a category. A vector $\mathbf{c}_i$ that encodes category i is full of zeros except for its ith bit which is set to one. This increases the number of features in the reference matrix $\mathbf{Y}$ by tenfold. Thus the number of samples and features in the reference matrix $\mathbf{Y}$ are $n = 31549, m_y = 210$ respectively.

Naturally, there are also other ways to select more features from the input data than those mentioned above. One such method using ProjSe could be to run the algorithm T times, so that at iteration t, for example m_y features would be selected from $\tilde{\mathbf{X}}_t \in \mathbb{R}^{n\times(m-tm_y)}$, which is the fingerprint data matrix from which the tm_y previously selected features have been removed. This way the reference data matrix $\mathbf{Y}$ would remain intact. Using this method, T internally nonredundant feature subsets in total would be selected, but there would be some redundancy between the subsets. By using this method, more features could be selected from $\mathbf{X}$ than with the two previously mentioned ones. However, as in this thesis feature selection is used, above all, to try to find as small feature subset as possible for producing accurate predictions of response values of the anti-cancer drugs, the two discussed methods should be sufficient.

4.3 Feature selection evaluation

When evaluating the performance of feature selection using kernel alignment, there are two different test settings: feature selection with the linear variable ranking method and feature selection with ProjSe using PTC values measured at five different concentrations. Since both the computation of Tanimoto kernels and the evaluation of their alignment have a time complexity $O(n^2)$, the computation quickly becomes unfeasible with large samples of data, not to mention the amount of memory required for computing the kernels of full fingerprint data sets with approximately 30 000 rows. Therefore, in every test setting, a sample of "only" $N = 10000$ drugs is taken both with full features and with $m = \{10, 25, 50, 75, 100\}$ features, and their Tanimoto kernels are computed. For each m, the kernel alignment between the kernel with m features and the kernel with full features is evaluated.

In the prediction tasks, mainly two different regressors are used: a very basic linear regressor and a nonlinear random forest regressor. For the latter algorithm, the following parameters are given: the number of trees in the forest is set to 100. The function used to measure the quality of a split is the mean squared error (MSE). The minimum number of samples required to split an internal node is 2, and the number of samples required to be at a leaf node is 1. After performing regression, the accuracy of the predictions is measured by calculating the Spearman correlation between the predicted values and ground truth labels. As with kernel alignment, there are a couple of different test settings for the prediction tasks as well: in addition to performing linear and random forest regression on the different fingerprint datasets containing the all the features, both feature selection methods are applied on the fingerprint data to select m features. Additionally, the two previously discussed methods to increase the size of the reference data matrix $\mathbf{Y}$ are used in order to improve the performance of feature selection.

In addition to linear and random forest regression, other models are used as well: SVR and two different MLPs. In SVR, squared epsilon insensitive loss is used as the loss function with ϵ set to 5. Since the implementation of SVR used in the experiments is linear, the variables are transformed into nonlinear ones before performing regression in order to improve the accuracy of the predictions. And since transforming the variables involves computing a nonlinear kernel (polynomial) on the variables and then calculating its eigenvalue decomposition, an approximation is used by taking a random subset of the kernel to speed up the process (Scholkopf et al., 1999). SVR is performed after applying ProjSe to the fingerprint data.

The modified MLP is given the fingerprint data with full features as it performs feature subset selection in itself, but the regular MLP performs regression both with and without feature selection (ProjSe). Both MLPs have a single hidden layer with $5m$ neurons, where m is the number of features in the input data, followed by a ReLU activation. The regular MLP is optimized using Adam (Kingma and Ba, 2017) and the network is trained without any regularization. The learning rate is set to 0.001.

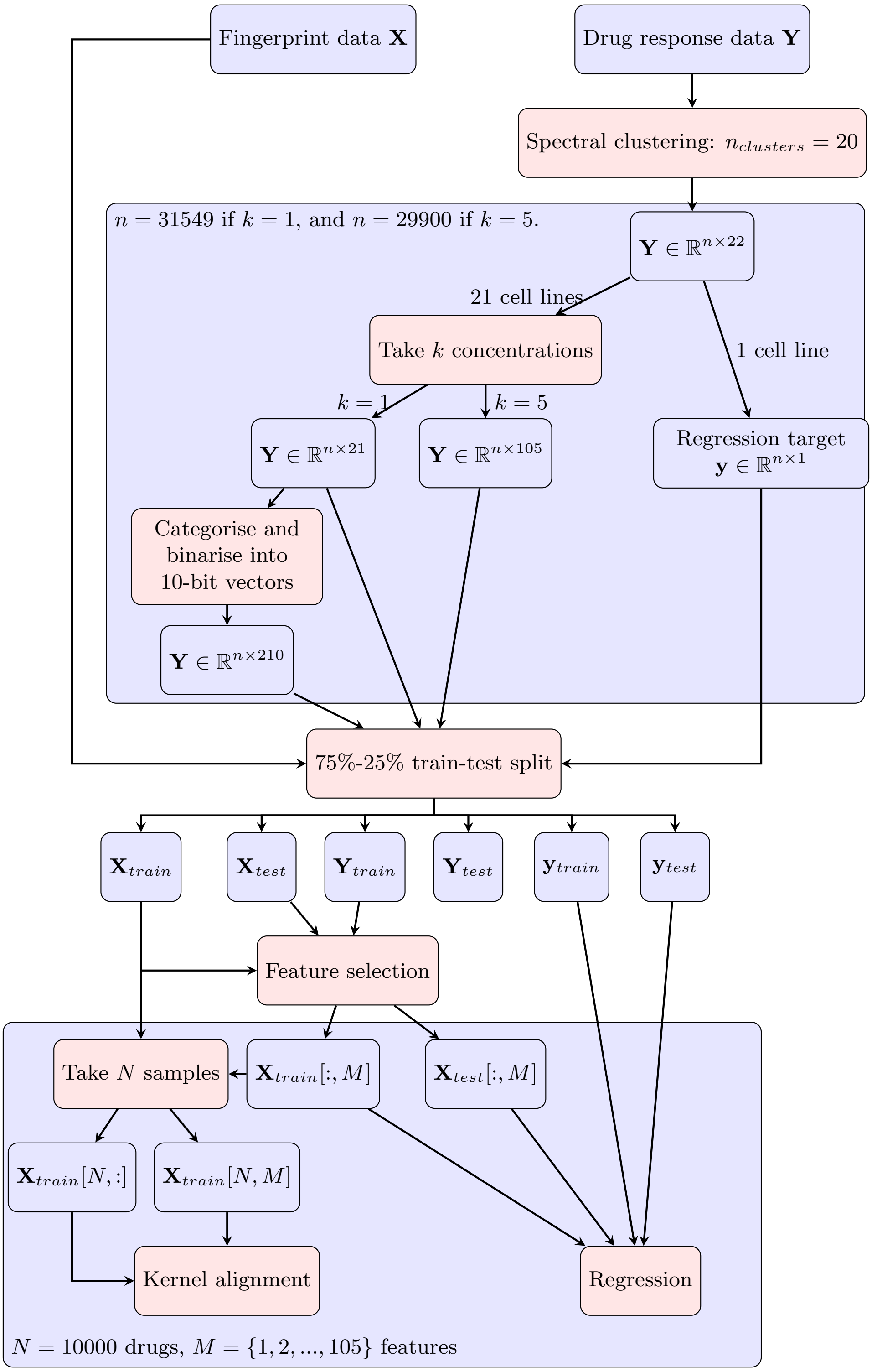


Figure 9: Flowchart showing the workflow of the experiments.

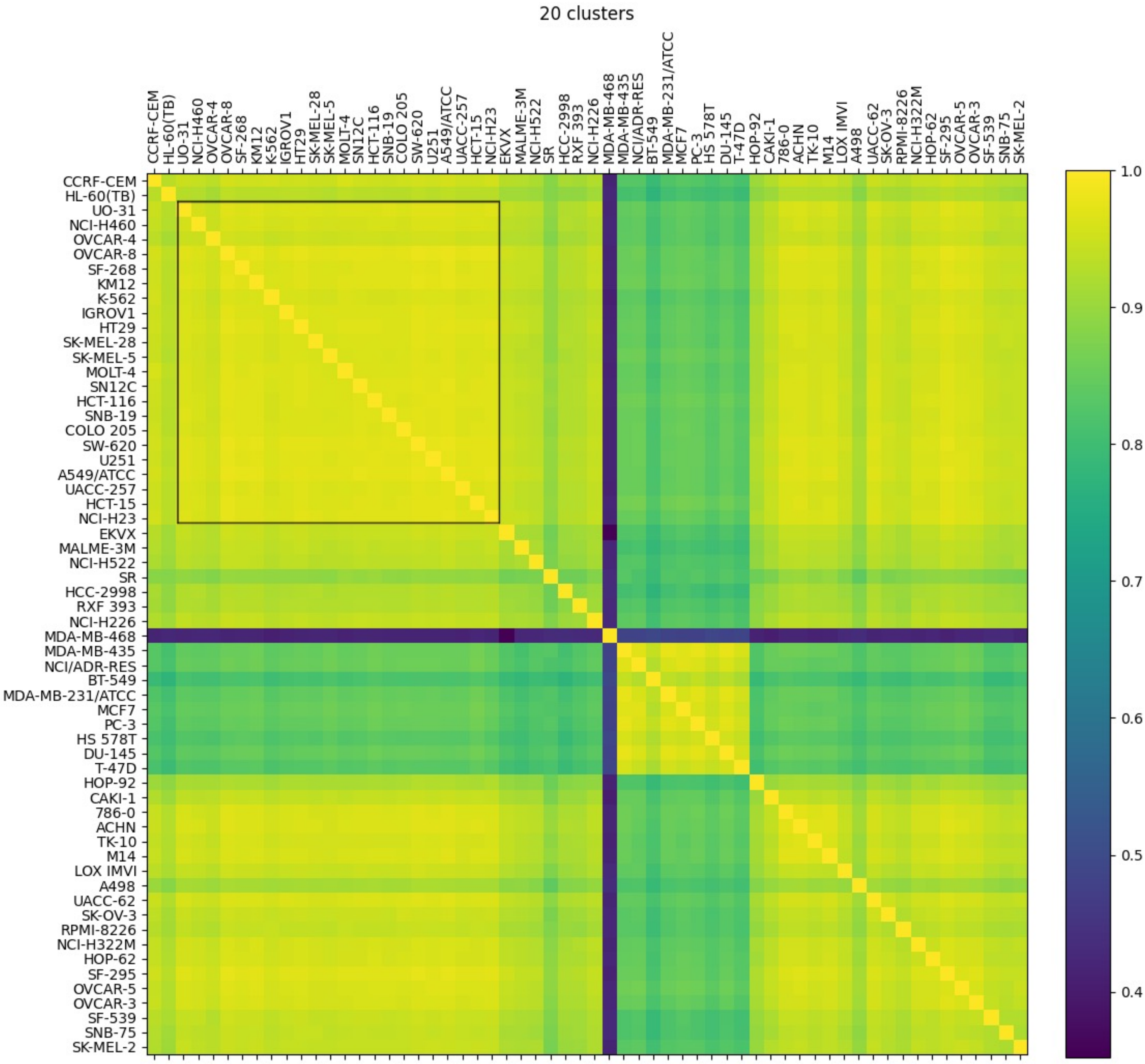


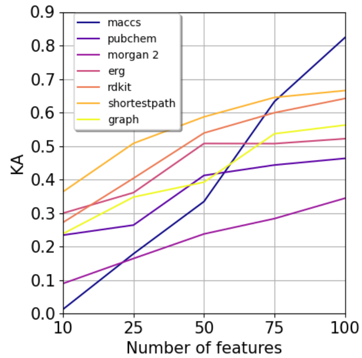
Figure 10: Spectral clustering on normalized common pairwise count matrix. The selected cluster containing 22 cell lines near the left upper corner of the matrix is highlighted.

5 Results

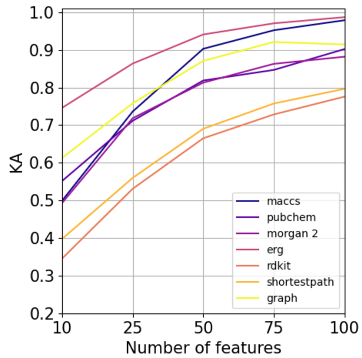
In this section, the results from the different experiments conducted are discussed. Results from kernel alignment and prediction tasks are presented in their respective subsections. Some of the results are pruned: the full results are shown in Appendices A-C.

5.1 Kernel alignment results

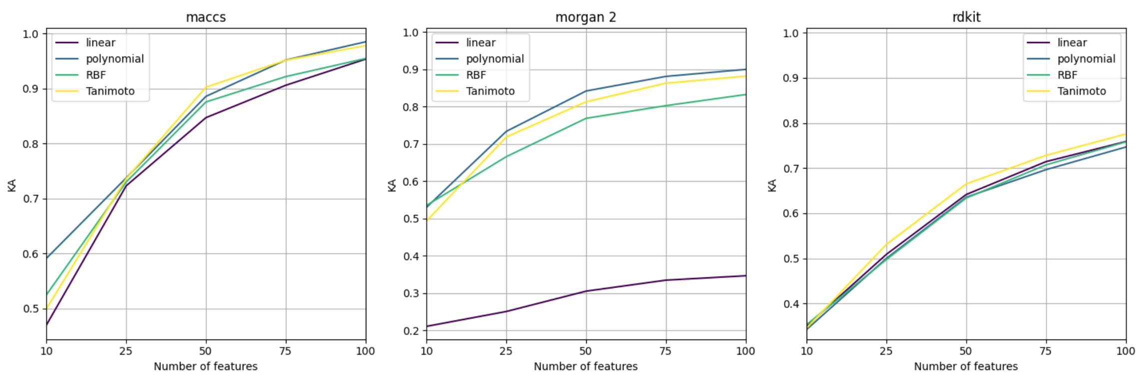
As discussed in Section 4.3, two different feature selection methods are used in order to select $m = \{10, 25, 50, 75, 100\}$ features from the seven different fingerprints. For every fingerprint dataset, a sample of 10 000 drugs is taken both with full and m features, and their Tanimoto kernels are evaluated. For each m, kernel alignment between the two Tanimoto kernels is computed. Results for kernel alignment when using the Pearson correlation coefficient (PCC) based ranking method in feature selection are shown in Figure 11a. For every fingerprint, the kernel alignment between



(a) PCC-based feature ranking



(b) ProjSe using Tanimoto kernel



(c) ProjSe with selected fingerprints

Figure 11: Kernel alignment between Tanimoto kernels with m and full features, computed for each type of fingerprint from samples of 10 000 drugs. In the PCC-based feature ranking, PTC values measured only at the highest concentration are used as reference data/regression targets while ProjSe is performed using PTC values at 5 different concentrations.

Tanimoto kernels with full features and with the least selected features, i.e. $m = 10$, is below 0.4, and particularly low for kernels evaluated from the MACCS and Morgan 2 fingerprints. As the m is increased, the kernel alignment improves for each fingerprint type, although more for some than others: for most fingerprints, the kernel alignment increases only by 0.2 to 0.3, but for the MACCS the increase is as high as approximately 0.8. The massive improvement in kernel alignment with the MACCS fingerprints is as expected as most of the features are included in the kernel at $m = 100$ since the MACCS has only 166 features in total. For the same reason, it is surprising that the kernel alignment with the MACCS is the lowest of the comparison at $m = 10$. Even the PubChem, the other substructure-keys based fingerprint with significantly more features (881) in total, produces higher kernel alignment at $m = 10$. For most m, the lowest kernel alignment is for kernels evaluated from Morgan 2 fingerprints. It is not unexpected that the Morgan 2 fingerprints yield the lowest kernel alignments as it has the most features (2048) in total, but on the other hand, the RDKit fingerprints, also having 2048 features, produce one of the highest kernel alignments in the comparison. This is likely due to the fact that practically all features in the RDKit fingeprints are at least weakly correlated to each other as was already seen in Section 2.2. The same is true for the Shortestpath fingerprints, also yielding better than average results.

ProjSe is used as the other feature selection method. Since it can select at most as many features from the input data $\mathbf{X}$ as there are in the reference data $\mathbf{Y}$, in order to be able to compare ProjSe to the results obtained using the other feature selection method, the size of $\mathbf{Y}$ has to be increased using either one of the two methods described in Section 4.2. Here, PTC values measured at five different dose concentrations are used as the reference data. Figure 11b shows the results from this setting with one of the better performing kernels, the Tanimoto kernel, used in ProjSe. With any m, the Tanimoto kernels evaluated from the ErG fingerprints yield the highest kernel alignment. Whilst starting with lower kernel alignment at $m = 10$, the performance of the MACCS fingerprints is satisfactory as well, reaching kernel alignment close to 1 at $m = 100$. Apart from the two, also the Graph, PubChem, and Morgan 2 fingerprints produce relatively high kernel alignments, while the kernels evaluated from the Shortestpath and RDKit fingerprints yield slightly poorer results, although still better than any fingerprint, except the MACCS, when the PCC-based feature ranking method is used for feature selection. It is noteworthy that with all fingerprints, unlike in Figure 11a, kernel alignment appears to improve more in the beginning ($10 \leq m \leq 50$) than later ($50 < m \leq 100$) which indicates that, from the start, ProjSe is better in finding the features that maintain similarity between data points.

In Figure 11c, kernel alignments using three different fingerprints, all producing very different results, are shown. Like with the Tanimoto kernel, the results show that the MACCS fingerprints yield high kernel alignments when using any of the other three kernels in ProjSe as well. When using the RDKit fingerprints, the resulting kernel alignments are notably lower than with the MACCS. On the other hand, all the curves are very congruent indicating that the importance of the choice of kernel is negligible. Instead, the Morgan 2 fingerprints, when using the linear kernel in ProjSe, yield very poor results compared to the other three kernels: kernel alignment starts significantly lower at $m = 10$ and improves just slightly, while with the other kernels, the performance is close as good as when using any kernel with the MACCS fingerprints. The full results for all fingerprints can be found in Appendix A.

Comparing the kernel alignment results using the two feature selection methods shows that with this kind of data ProjSe is superior to the PCC-based feature ranking method. For ProjSe, it is a particularly good sign that kernel alignments are moderately high (≥ 0.5) with most fingerprints already with just $m = 10$ features, indicating that ProjSe manages to find first the most significant features to maintain the similarity between data points. The better performance of ProjSe compared to the other method is not unexpected, as ProjSe can handle more complex data by using nonlinear kernels. Nevertheless, it appears that even with linear kernel, ProjSe outperforms the other method. Being able to take in considerably more reference data is likely to help with this. Fingerprint-wise, the MACCS fingerprints perform well (with sufficient features) with either feature selection method, but otherwise the results are quite dissimilar. While the RDKit and Shortestpath fingerprints yield the second and the third highest kernel alignments when using the PCC-based feature ranking method for feature selection, the two produce the lowest kernel alignments when using ProjSe with Tanimoto kernel. On the other hand, while the ErG fingerprints yield only mediocre kernel alignment with the PCC-based feature selection method, they produce the highest kernel alignment when ProjSe is used for feature selection. Furthermore, while the Morgan 2 fingerprints yield, on average, the lowest kernel alignment of the comparison with the PCC-based feature selection, they yield approximately median results with ProjSe if linear kernel is ignored.

5.2 Regression results

In the regression tasks, predictions of PTC values are made from the different fingerprints using linear, random forest and support vector regression, and the two MLPs as was discussed in Section 3.3.2. The models are trained with the different fingerprints using PTC values for one tumor cell line (A549/ATCC) as regression targets. The results for linear and random forest regression on fingerprint data without feature selection are shown in Table 3. The results show that there are notable differences in the performance of linear regression on the different fingerprints. Linear regression on the Morgan 2, PubChem and RDKit produce the most accurate predictions that are significantly better than the predictions from the worst two, i.e. the ErG and Graph fingerprints.

In random forest regression, the best prediction accuracy is achieved with the PubChem fingerprints, although it is almost as high with any other fingerprint except for the Graph. Comparing linear to random forest regression, it is evident that the latter manages to predict PTC values with notably higher accuracy. Spearman correlations of the predictions from any of the fingerprints are higher than when using linear regression. There are at least two likely reasons why random forest regression performs better than linear regression: 1) random forest is able to capture nonlinearities in the data and 2) its bootstrapping of the input data, and aggregation and averaging of the results make it more robust method to possible outliers in the data.

5.2.1 Feature selection with 21 features

Including feature selection into the prediction task, both ProjSe and the PCC based feature ranking method are used. First, without using either data manipulation technique for increasing the size of reference data matrix **Y** (see Section 4.2), linear and random forest regression are performed on fingerprint data with just 21 features. In the PCC-based feature ranking method, the 21 features are selected from the different fingerprints using PTC values measured at one concentration for only one tumor cell line as the reference, since the method accepts only a 1D vector as the regression target. The feature selection and the linear and random forest regression are repeated so that PTC values for each cell line is used at a time. The resulting correlations shown in Table 4 are averaged for the 21 cell lines.

Comparing these results to Table 3, it is evident that 21 features are not enough to produce nearly as accurate predictions as with full features. Overall, from linear regression, the resulting Spearman correlations are very close to zero. Like in regression with full features, random forest regression performs notably better even on such a small number of features: only the lowest prediction accuracy (ErG) is worse than the best in linear regression (MACCS, PubChem). Comparing these results to regression with full features, only random forest regression with feature selection on the RDKit fingerprints produces predictions with approximately the same order of

Fingerprint	LR	RFR
MACCS	0.45	0.62
PubChem	0.51	**0.64**
Morgan 2	**0.52**	0.60
ErG	0.37	0.59
RDKit	0.51	0.61
Shortestpath	0.46	0.61
Graph	0.41	0.53
mean	0.46	0.60
median	0.46	0.61

Table 3: Spearman correlation coefficients from linear (LR) and random forest (RFR) regression on the fingerprint data sets with full set of features. Fingerprint producing the most accurate predictions for each learner is highlighted.

Fingerprint	LR	RFR
MACCS	**0.18**	0.27
PubChem	**0.18**	0.22
Morgan 2	0.15	0.20
ErG	0.12	0.17
RDKit	0.06	**0.39**
Shortestpath	0.10	0.32
Graph	0.17	0.21
mean	0.14	0.25
median	0.15	0.22

Table 4: Spearman correlations from linear (LR) and random forest (RFR) regression of 21 runs. Feature selection was performed using a filter-type PCC-based linear feature selection method with PTC values of one tumor cell line at a time as the reference data/regression targets. Each time 21 features were selected. Fingerprint producing the most accurate predictions for each learner is highlighted

magnitude of accuracy as linear regression with full features, let alone random forest regression.

Fingerprint-wise, similarly to the regression results with full features, there are notable differences in prediction accuracies between the different fingerprints, but they are much greater than with full features, relatively speaking. In linear regression the best Spearman correlations (MACCS, Pubchem) are up to three times higher than the worst (RDKit). Respectively, in random forest regression, the highest Spearman correlation (RDKit) is more than twice as high as the lowest (ErG). It appears that regression performing relatively well on some fingerprint with full features does not necessarily imply that it performs well on the same fingerprints also with feature selection. For example, regression on Morgan 2 fingerprints with full features produce predictions with comparatively high accuracy, but the same is not true with feature selection. Another example are the RDKit fingerprints: linear and random forest regression produce relatively good results on the RDKit with full features. This is also true for random forest regression with feature selection, but, surprisingly, linear regression on the RDKit with feature selection produces the lowest prediction accuracy in the comparison.

Overall, as the regression results with feature selection are notably worse compared to those produced with full features, it is evident that the PCC-based feature ranking feature selection method does not manage to find the features that contribute to the drug response the most. Not at least with so few features.

With ProjSe as the other feature selection method, drug response data measured at the highest concentration for 21 tumor cell lines is used. As ProjSe can select at

Fingerprint	LR					RFR				
	lin	poly	rbf	tani	mean	lin	poly	rbf	tani	mean
MACCS	0.33	0.28	0.35	0.37	**0.33**	0.43	0.40	0.43	0.47	0.43
PubChem	0.29	0.33	0.32	0.30	0.31	0.38	0.39	0.41	0.39	0.39
Morgan 2	0.22	0.23	0.22	0.25	0.23	0.25	0.37	0.32	0.35	0.32
ErG	0.31	0.25	0.27	0.28	0.28	0.53	0.53	0.54	0.54	**0.54**
RDKit	0.20	0.20	0.24	0.22	0.22	0.43	0.38	0.43	0.42	0.42
Shortestpath	0.24	0.22	0.24	0.29	0.25	0.36	0.34	0.38	0.42	0.38
Graph	0.19	0.26	0.29	0.29	0.26	0.24	0.31	0.33	0.37	0.31
mean	0.25	0.25	0.28	0.29		0.37	0.39	0.41	0.42	
median	0.24	0.25	0.27	0.29		0.38	0.38	0.41	0.42	

Table 5: Spearman correlations from linear (LR) and random forest (RFR) regression on the fingerprint data sets with 21 features selected using ProjSe with linear, polynomial, RBF and Tanimoto kernels. PTC-values from 21 cell lines are used as the reference data in feature selection. The kernel contributing to the best prediction accuracy is marked for each fingerprint with yellow for linear regression and blue for random forest regression. Additionally, the mean Spearman correlation of the fingerprint yielding the highest prediction accuracy is highlighted.

most the number of features as there is in the reference data, 21 features are selected from the different fingerprints, making the results also comparable to those shown in Table 4. The results using ProjSe are shown in Table 5. First, comparing the four different kernels, on average, linear regression yields the most accurate predictions when Tanimoto kernel is used in ProjSe, while the second best choice is the RBF. Using either polynomial or linear kernel lead to slightly less accurate predictions. In random forest regression, the ranking of the kernels is approximately the same.

When comparing the two feature selection methods, it is especially interesting to examine how well the linear PCC-based method competes to ProjSe with linear kernel. As the results show, ProjSe with linear kernel produces higher prediction accuracy in linear regression on every fingerprint. The only fingerprints on which linear regression produces approximately as good predictions with both feature selection methods are the Graph. Similarly, with random forest regression, using ProjSe with linear kernel in feature selection produces more accurate predictions from all fingerprints.

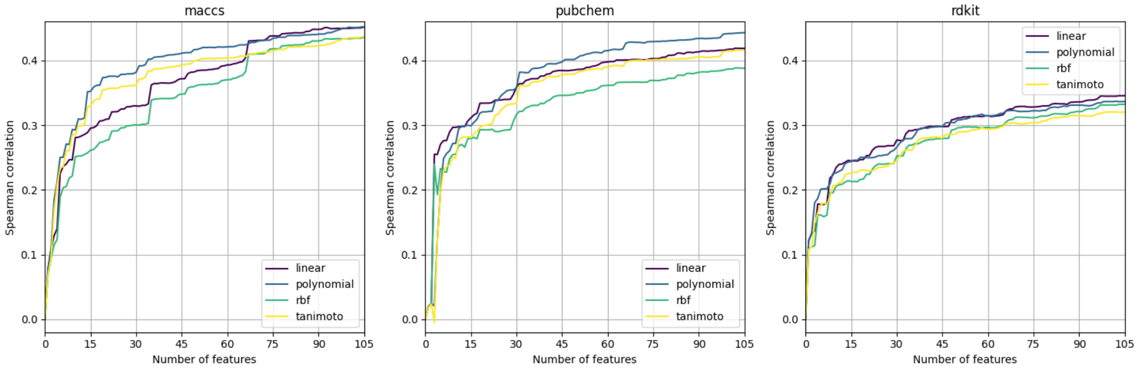
When comparing these results with ProjSe from linear regression to those with full features (Table 3), all Spearman correlations are notably smaller, indicating that 21 features may not be enough to predict drug response well enough. The same is true for most fingerprints with random forest regression as well. The only exception are the ErG fingerprints which, with just 21 features, produce predictions with almost as high accuracy as with full features.

Fingerprint-wise, in linear regression, the highest prediction accuracy, on average, is from the MACCS fingerprints. The Pubchem fingerprints produce relatively good predictions as well. Instead, the most inaccurate predictions are made from the RDKit fingerprints. In random forest regression, the ErG is clearly producing the most accurate predictions with any kernel as the mean Spearman correlation is notably higher than with any other fingerprint. The least accurate predictions in random forest regression are produced using the Graph fingerprints.

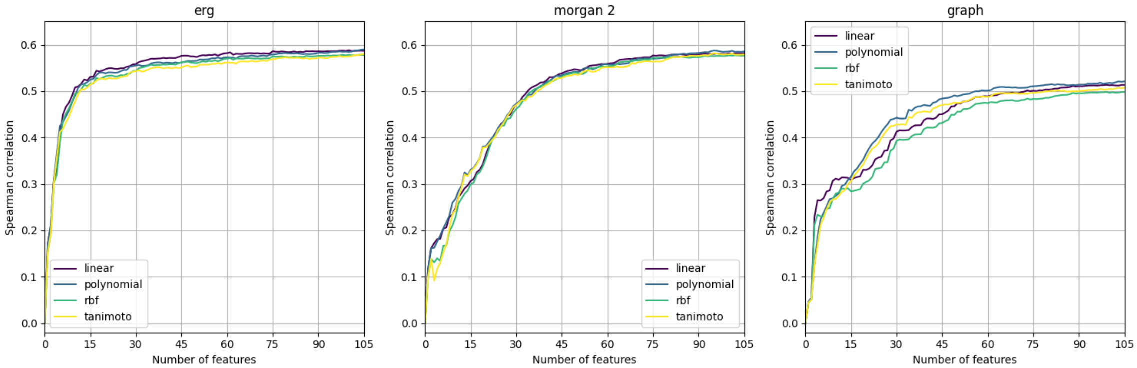
Concluding from the results, using either feature selection method on all fingerprints leads to notably less accurate predictions of drug response values than with full features with one exception: random forest regression on the Erg fingerprints. The fact that the results are almost always worse with than without feature selection is not unexpected as only 21 features are selected from the fingerprints that contain hundreds, if not thousands, of features in total. Therefore, to obtain prediction accuracies closer to those achieved with full features, adding more features needs to be investigated. Since it is evident that, on average, ProjSe performs notably better than the PCC-based feature ranking method, in the following regression tasks, only ProjSe is used for feature selection.

5.2.2 Feature selection with more features

In order to select more features from the fingerprints, the number of features in the reference data $\mathbf{Y}$ is increased by taking PTC values measured at the five different concentrations which results in $m_Y = 105$ features. Therefore, 105 features are selected using ProjSe, and linear and random forest regression are performed on the fingerprints iteratively adding one feature at a time. The results for linear regression



(a) Linear regression



(b) Random forest regression

Figure 12: Regression on selected fingerprints. PTC-values from five different concentrations are used as the reference data in ProjSe to select 105 features in total from each fingerprint.

on a few selected fingerprints are shown in Figure 12a. The results show that the correlation curves are all more or less r-shaped, i.e. the prediction accuracy increases quickly at the beginning as new features are added, but the improvement starts slowing down relatively soon. This indicates that ProjSe manages to find first the features that contribute the most to the increase in the prediction accuracy. Fingerprint-wise, the most accurate predictions are made from the MACCS and PubChem, while using the RDKit produces the most inaccurate predictions. The full results, i.e. for all seven fingerprints can be found in Appendix B.1. The plots also show that the differences in how well the different kernels perform varies. While with some fingerprints, the differences are greater, with others like the RDKit the curves are more congruent. Furthermore, it appears that there is no clearly the best kernel as their relative performance varies on the fingerprints. Yet, using the polynomial kernel seems to produce just about the highest correlations on most fingerprints (see Appendix B.1).

For the same setting, the results from random forest regression with three selected fingerprints are shown in Figure 12b. The plot for the ErG fingerprints shows the same behavior already seen in Table 5. Random forest regression performs comparatively well on the ErG with very few features as the prediction accuracy increases very sharply at the beginning and starts converging soon after. On the contrary, when

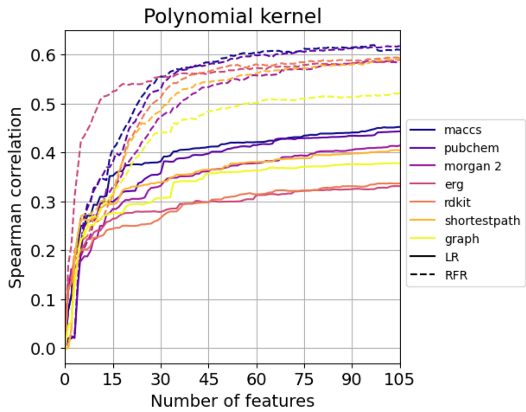
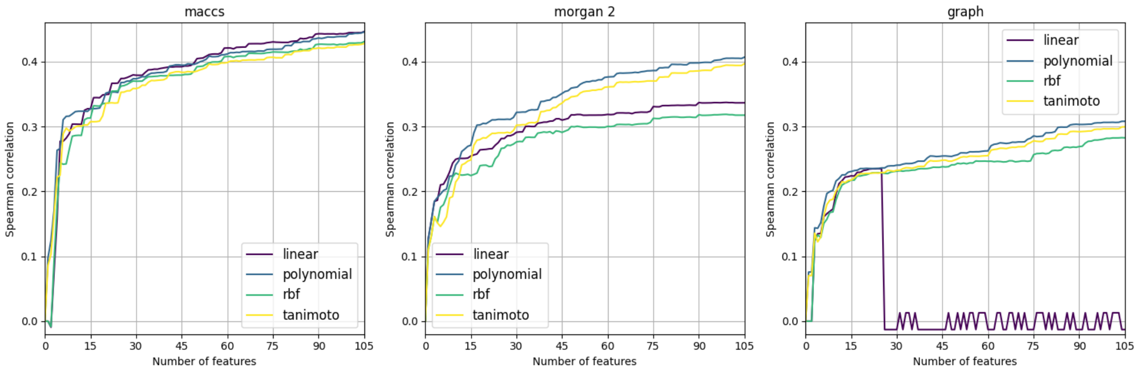


Figure 13: Linear vs. random forest regression with polynomial kernel used in ProjSe. PTC values at five concentrations used in as the reference data in ProjSe.

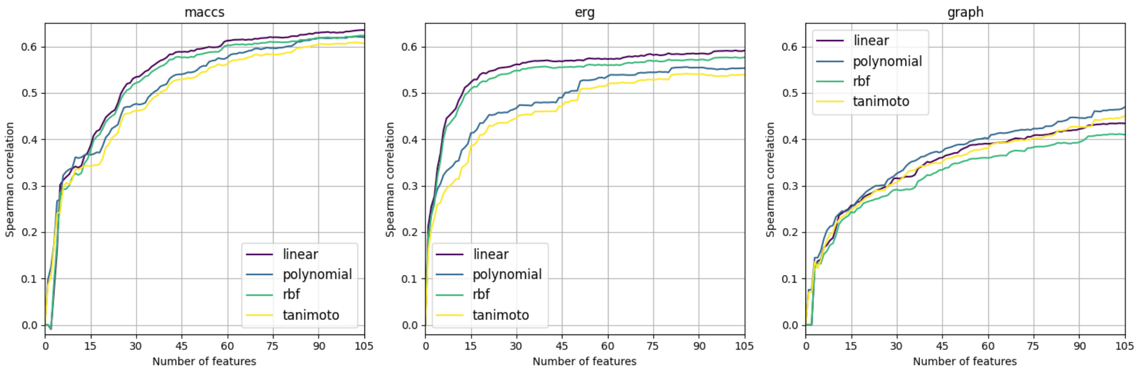
using the Morgan 2 or Graph fingerprints, the Spearman correlations improve at much slower pace. The correlation curves of the different kernels are somewhat more congruent than in the case of linear regression suggesting that the choice of kernel in ProjSe is slightly less important with random forest regression. For the results of other fingerprints, see Appendix B.2.

The two learning algorithms are compared in Figure 13 where polynomial kernel is used in ProjSe. The plot shows clearly that random forest regression performs better on any of the fingerprints already with very few features ($m \approx 15$). The plot also shows that the performance of the two learners on the different fingerprints varies. While the two best performing fingerprints, i.e. the MACCS and PubChem, produce comparatively good results with either learner, that is not the case with all fingerprints: random forest regression performs well on the ErG fingerprints, whereas linear regression performs poorly when compared to the other fingerprints. Another example are the Graph fingerprints: while the performance of random forest regression on the said fingerprints is clearly the worst, linear regression on them produces comparatively mediocre results.

Another way to increase the size of the reference data matrix $\mathbf{Y}$, instead of including PTC values measured at multiple dose concentrations is by categorizing and binarizing the PTC values measured at one concentration as already discussed in Section 4.2. Using this method in feature selection, the results from linear regression for three selected fingerprints are shown in Figure 14a. Again, linear regression performs relatively well on the MACCS fingerprints with anl the kernels used in ProjSe producing very similar results. When using the Morgan 2 fingerprints, there is notably more variation how well the different kernels perform. While using the polynomial or Tanimoto kernel produce close as high prediction accuracy as linear regression on the MACCS, using either linear or RBF kernel results in significantly lower prediction accuracy. The worst prediction accuracy, on average, in this test setting with linear regression is achieved when using the Graph fingerprints. Furthermore, when using linear kernel in ProjSe, the regression on the Graph fails as the correlation curve suddenly drops to zero when approximately the 25th feature is added to the fingerprints.



(a) Linear regression



(b) Random forest regression

Figure 14: Regression on selected features. PTC-values are categorized and binarized into 10-bit vectors and used as the reference data in ProjSe to select 105 features from each fingerprint.

Interestingly, when using Pearson correlation as the measure of similarity between predicted PTC values and ground truth labels, the same does not occur. Results for the rest of the fingerprints are fairly similar with correlations being slightly worse than with the MACCS but also better than with the Graph fingerprints. They can be found in Appendix B.3.

Results from random forest regression for three selected fingerprints, the MACCS, ErG, and Graph, are showed in Figure 14b. Again, the highest prediction accuracy is obtained using the MACCS, but random forest regression performs well on the ErG fingerprints as well, especially with small number of features. However, there is significant difference between the different kernels when using the ErG fingerprints: while linear and RBF kernels produce similar results to those shown in Figure 12b, Spearman correlations using polynomial and Tanimoto kernels increase at notably slower pace. As with linear regression, random forest regression produces, on average, the most inaccurate predictions from the Graph fingerprints. Results for all fingerprints can be found in Appendix B.4.

Furthermore, when comparing linear and random forest regression in this test setting, similarly to Figure 13, Figure 15 shows that random forest regression outperforms

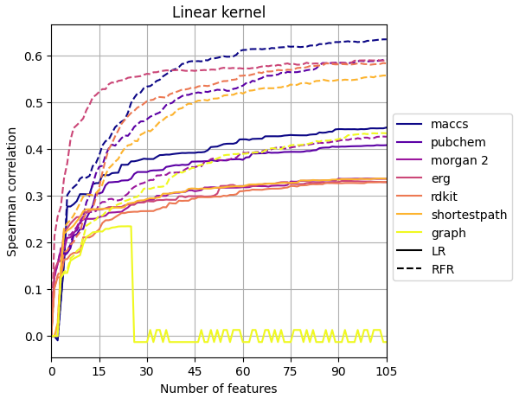


Figure 15: Linear vs. random forest regression with linear kernel used in ProjSe. PTC values used in the reference data in ProjSe are categorized and binarized into 10-bit vectors.

linear regression on any fingerprint. Instead, when comparing the two methods to increase the size of the reference matrix, it appears that neither of them is clearly better than the other. However, using PTC values at five different concentrations appears to produce slightly more stable results. Therefore, in the following experiments where linear and random forest regression are compared to SVR and the two MLPs, PTC values from five different concentrations are used as reference data in ProjSe.

5.2.3 Regressor comparison

In the final experiments, the different regressors are compared to study which one can produce the most accurate predictions of drug response values from molecular fingerprint data. SVR is performed on the different fingerprints with 105 features selected using ProjSe with drug response data at five different dose concentrations used as the reference data. Of the two MLPs, the regular performs regression both on the fingerprints with full features and with 105 features selected by ProjSe similarly as with SVR, while the modified MLP is given only fingerprints with full features as inputs since it performs feature selection itself. In the modified MLP, the sparsity of the $\mathbf{T}$ matrix, that gives the weights to the inputs, and the strength of L_1 regularization are varied, but only the best results for each fingerprints are reported. The results are shown in Table 6. The earlier results from linear and random forest regression on fingerprints both with full features and feature selection ($m = 105$) are included in the table for comparison.

As the results show, on fingerprint data with full features, random forest regression, on average, yields the most accurate predictions. However, the regular MLP with a large single hidden layer yields, on average, almost as accurate predictions. In fact, on some fingerprints, the MLP outperforms random forest regression, although just slightly. On the other hand, both random forest regression and the regular MLP outperform linear regression by a large margin.

With feature selection included, random forest regression outperforms every other regressor on most fingerprints yielding the most accurate predictions while the least

Fingerprint	LR full	RFR full	MLP full	mMLP	LR fs	RFR fs	SVR fs	MLP fs	mean
MACCS	0.45	0.62	0.54	0.50	0.44	0.61	0.51	0.52	0.52
PubChem	0.51	0.64	0.61	0.55	0.41	0.59	0.49	0.51	**0.54**
Morgan 2	0.52	0.60	0.63	0.56	0.38	0.53	0.44	0.43	0.51
ErG	0.37	0.59	0.55	0.45	0.35	0.58	0.33	0.53	0.47
RDKit	0.51	0.61	0.62	0.56	0.33	0.58	0.44	0.47	0.51
Shortestpath	0.46	0.61	0.62	0.52	0.38	0.57	0.46	0.45	0.51
Graph	0.41	0.53	0.49	0.47	0.35	0.47	0.40	0.42	0.44
mean	0.46	0.60	0.58	0.52	0.38	0.56	0.44	0.47	
median	0.46	0.61	0.61	0.52	0.38	0.58	0.44	0.47	

Table 6: From left to right: Spearman correlations from predictions with linear regression (LR) with full features vs. random forest regression (RFR) with full features vs. regular MLP with full features vs. modified MLP (mMLP) vs. LR with feature selection (mean of four kernels used in ProjSe) vs. RFR with feature selection (mean) vs. SVR with feature selection (mean) vs. regular MLP with feature selection (mean). The most accurate predictions from fingerprints with full features (yellow) and with feature selection, either by using ProjSe (m = 105) or using the mMLP (blue) are highlighted. Fingerprint yielding the highest Spearman correlation on average is highlighted as well. In ProjSe, PTC values from five different concentrations are used as the reference data.

accurate predictions are again from linear regression. After random forest regression, the second best performing learner is the modified MLP, yielding better prediction accuracy than random forest regression on the Morgan 2 fingerprints. Of the two remaining algorithms, the regular MLP outperforms SVR on all but two fingerprints.

Fingerprint-wise, while regression on the PubChem produces the most accurate predictions on average, regression on four other fingerprints, i.e. the MACCS, Morgan 2, RDKit, and Shortestpath, yield almost as high Spearman correlations. The two fingerprints producing the least accurate predictions are the ErG and Graph. However, it must be noted that these results do not show how the different regressors perform with fewer number of features. Earlier it was shown that, for example, random forest regression on the ErG performs very well with very few features compared to other fingerprints.

When comparing the performance of models with and without feature selection, it appears that linear regression and the regular MLP perform notably worse with feature selection, except on two fingerprints: the MACCS and ErG. Instead, random forest regression produces predictions with almost as high accuracy with and without feature selection, with few exceptions.

Concluding, of the models used in this comparison, it appears that random forest regression is, on average, the best model to predict drug response values from molecular fingerprint data. Furthermore, it is evident that linear regression is the weakest algorithm of the comparison. The regular MLP with a single large hidden layer performs relatively well on the data as well, but it is significantly heavier algorithm ($5m$ neurons in the hidden layer) compared to the random forest regressor (100 trees). Comparing the two MLPs, it seems that the regular MLP without feature selection always outperforms the modified MLP, but with most fingerprints, the modified MLP produces more accurate predictions than the regular MLP with feature selection using ProjSe. SVR, while producing more accurate predictions than linear regression, is mostly outperformed by both MLPs and random forest regression. Full kernel-wise results for SVR can be found in Appendix C.

6 Discussion

This thesis focused on the NCI-60 human tumor cell lines screen growth inhibition dataset and seven different types of molecular fingerprints generated for the anticancer drugs present in the dataset. The goal was to determine if it is possible to 1) accurately predict drug response values from molecular fingerprints using regression, and 2) find (using feature selection) features in the molecular fingerprints that contribute to the therapeutic response of anticancer drugs the most. Mainly two methods were used for feature selection: ProjSe, the recent kernel-based feature selection method using projection operators and a traditional linear PCC-based feature ranking method.

ProjSe and a Pearson correlation coefficient (PCC) based feature ranking method were applied on the fingerprints and their success was evaluated both using an unsupervised and a supervised method. The experiments showed that when comparing to the basic variable ranking method, in most cases, ProjSe was superior in unsupervised evaluation, both with linear and nonlinear kernelization of variables, managing to yield notably higher values of kernel alignment for Tanimoto kernels evaluated from most fingerprints, on which feature selection was applied, even with very few features.

As the supervised evaluation method measuring the success of feature selection was regression using a few different models: linear, random, and support vector regression (SVR), and two slightly different multilayer perceptrons (MLP) with a single large hidden layer. Regression results on fingerprints with full features showed that the random forest regression and the regular MLP performed the best yielding the highest prediction accuracy, while the most inaccurate predictions were from linear regression. Applying feature selection on the fingerprints showed that, like in the case of kernel alignment, the basic linear variable ranking algorithm used to select 21 features yielded worse results than ProjSe, even when used with linear kernel. Increasing the size of the reference data matrix in ProjSe, by either method explained in Section 4.2, improved the accuracy of the predictions of the models, as it allowed more features to be selected, in many cases leading to almost as accurate predictions as with full features. The modified MLP and SVR yielded slightly worse results, but of all the models, linear regression produced the least accurate predictions. Kernel-wise, it appears that no single kernel in ProjSe is clearly better than the others in every experiment in selecting features from molecular fingerprints. Furthermore, the results showed that, on average, linear kernel performed just as well as the nonlinear ones. Fingerprint-wise, the results showed that using substructure-keys based fingerprints, i.e. the MACCS and PubChem, produced the most accurate predictions, on average.

In conclusion, ProjSe offers a kernel-based alternative for traditional feature selection methods on data with up to a moderate number of features. It performs well on molecular fingerprint data as it is able to find features that maintain a good similarity between data points as the kernel alignment results showed. Also, several regressors perform almost as well on fingerprints with the features ProjSe selects compared to fingerprints with full features. Especially, using ProjSe on the ErG fingerprints, it is possible to find a small number of features that produce almost as good prediction accuracy in random forest regression as with full features. On the other hand, the results suggest that it is a rather difficult task to predict real valued labels from molecular fingerprint data with very high accuracy, even for nonlinear algorithms such as random forest regression and MLP. However, as the data was experimented only on few regressors, it is possible that there is more suitable algorithms for the task. If such algorithm is found, the features of molecular fingerprints that contribute to the drug response the most could be used in drug discovery to find new promising drugs.

References

Beni, G. and Wang, J. (1993). Swarm intelligence in cellular robotic systems. In *Robots and biological systems: towards a new bionics?*, pages 703–712. Springer.

Blum, A. L. and Langley, P. (1997). Selection of relevant features and examples in machine learning. *Artificial intelligence*, 97(1-2):245–271.

Breiman, L. (2001). Random forests. *Machine learning*, 45:5–32.

Brouard, C., Mariette, J., Flamary, R., and Vialaneix, N. (2022). Feature selection for kernel methods in systems biology. *NAR Genomics and Bioinformatics*, 4(1):1–17.

Cereto-Massagué, A., Ojeda, M. J., Valls, C., Mulero, M., Garcia-Vallvé, S., and Pujadas, G. (2015). Molecular fingerprint similarity search in virtual screening. *Methods*, 71:58–63.

Coleman, M. and Marks, H. (1998). Topics in dose-response modeling. *Journal of food protection*, 61(11):1550–1559.

Cortes, C., Mohri, M., and Rostamizadeh, A. (2010). Two-stage learning kernel algorithms. *Proceedings of the 27th International Conference on Machine Learning*.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20:273–297.

Cover, T. M. (1974). The best two independent measurements are not the two best. *IEEE Transactions on Systems, Man, and Cybernetics*, (1):116–117.

Durant, J. L., Leland, B. A., Henry, D. R., and Nourse, J. G. (2002). Reoptimization of mdl keys for use in drug discovery. *Journal of chemical information and computer sciences*, 42(6):1273–1280.

Dy, J. G. and Brodley, C. E. (2004). Feature selection for unsupervised learning. *Journal of machine learning research*, 5(Aug):845–889.

Fukushima, K. (1969). Visual feature extraction by a multilayered network of analog threshold elements. *IEEE Transactions on Systems Science and Cybernetics*, 5(4):322–333.

Genton, M. G. (2001). Classes of kernels for machine learning: a statistics perspective. *Journal of machine learning research*, 2(Dec):299–312.

Guyon, I. and Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of machine learning research*, 3(Mar):1157–1182.

Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, 24(6):417.

Hotelling, H. (1992). Relations between two sets of variates. In *Breakthroughs in statistics: methodology and distribution*, pages 162–190. Springer.

Huang, S. H. (2015). Supervised feature selection: A tutorial. *Artif. Intell. Res.*, 4(2):22–37.

Jović, A., Brkić, K., and Bogunović, N. (2015). A review of feature selection methods with applications. In *2015 38th international convention on information and communication technology, electronics and microelectronics (MIPRO)*, pages 1200–1205. Ieee.

Kingma, D. P. and Ba, J. (2017). Adam: A method for stochastic optimization. *arXiv:1412.6980*.

Kohavi, R. and John, G. H. (1997). Wrappers for feature subset selection. *Artificial intelligence*, 97(1-2):273–324.

Kumar, V. and Minz, S. (2014). Feature selection. *SmartCR*, 4(3):211–229.

Landrum, G. (2012). Fingerprints in the rdkit. https://www.rdkit.org/UGM/2012/Landrum_RDKit_UGM.Fingerprints.Final.pptx.pdf. Accessed on July 29, 2024.

Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. P., Tang, J., and Liu, H. (2017). Feature selection: A data perspective. *ACM computing surveys (CSUR)*, 50(6):1–45.

Linnainmaa, S. (1970). *The representation of the cumulative rounding error of an algorithm as a Taylor expansion of the local rounding errors*. PhD thesis, Master's Thesis (in Finnish), Univ. Helsinki.

MacQueen, J. et al. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA.

Meinshausen, N. and Bühlmann, P. (2010). Stability selection. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 72(4):417–473.

Mohri, M., Rostamizadeh, A., and Talwalkar, A. (2018). *Foundations of machine learning*. MIT press.

Morgan, H. L. (1965). The generation of a unique machine description for chemical structures-a technique developed at chemical abstracts service. *Journal of chemical documentation*, 5(2):107–113.

National Cancer Institute (2024). NCI-60 Growth Inhibition Data. https://wiki.nci.nih.gov/display/ncidtpdata/nci-60+growth+inhibition+data. Accessed on May 14, 2024.

National Health Institute (2009). Pubchem substructure fingerprint. https://ftp.ncbi.nlm.nih.gov/pubchem/specifications/pubchem_fingerprints.pdf. Accessed on July 29, 2024.

Ng, A., Jordan, M., and Weiss, Y. (2001). On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems*, 14.

Pearson, K. (1895). Vii. note on regression and inheritance in the case of two parents. *proceedings of the royal society of London*, 58(347-352):240–242.

Pearson, K. (1901). Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin philosophical magazine and journal of science*, 2(11):559–572.

Ralaivola, L., Swamidass, S. J., Saigo, H., and Baldi, P. (2005). Graph kernels for chemical informatics. *Neural networks*, 18(8):1093–1110.

RDKit (2013). RDKit: Open-source cheminformatics. https://www.rdkit.org. Accessed on July 29, 2024.

Rogers, D. and Hahn, M. (2010). Extended-connectivity fingerprints. *Journal of chemical information and modeling*, 50(5):742–754.

Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386.

Scholkopf, B., Mika, S., Burges, C. J., Knirsch, P., Muller, K.-R., Ratsch, G., and Smola, A. J. (1999). Input space versus feature space in kernel-based methods. *IEEE transactions on neural networks*, 10(5):1000–1017.

Shannon, C. E. (1948). A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423.

Solorio-Fernández, S., Carrasco-Ochoa, J. A., and Martínez-Trinidad, J. F. (2020). A review of unsupervised feature selection methods. *Artificial Intelligence Review*, 53(2):907–948.

Stiefl, N., Watson, I. A., Baumann, K., and Zaliani, A. (2006). Erg: 2d pharmacophore descriptions for scaffold hopping. *Journal of chemical information and modeling*, 46(1):208–220.

Swain, M. (2014). Pubchempy. https://pubchempy.readthedocs.io. Accessed on July 29, 2024.

Szedmak, S. and Bach, E. (2020). On the generalization of tanimoto-type kernels to real valued functions. *arXiv preprint arXiv:2007.05943*.

Szedmak, S., Huusari, R., Duong Le, T. H., and Rousu, J. (2023). Scalable variable selection for two-view learning tasks with projection operators. *Machine Learning*, pages 1–20.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 58(1):267–288.

Walters, W. P., Stahl, M. T., and Murcko, M. A. (1998). Virtual screening—an overview. *Drug discovery today*, 3(4):160–178.

Willighagen, E. L., Mayfield, J. W., Alvarsson, J., Berg, A., Carlsson, L., Jeliazkova, N., Kuhn, S., Pluskal, T., Rojas-Chertó, M., Spjuth, O., et al. (2017). The chemistry development kit (cdk) v2. 0: atom typing, depiction, molecular formulas, and substructure searching. *Journal of cheminformatics*, 9:1–19.

Zhao, J., Xie, X., Xu, X., and Sun, S. (2017). Multi-view learning overview: Recent progress and new challenges. *Information Fusion*, 38:43–54.

A Full kernel alignment results with ProjSe

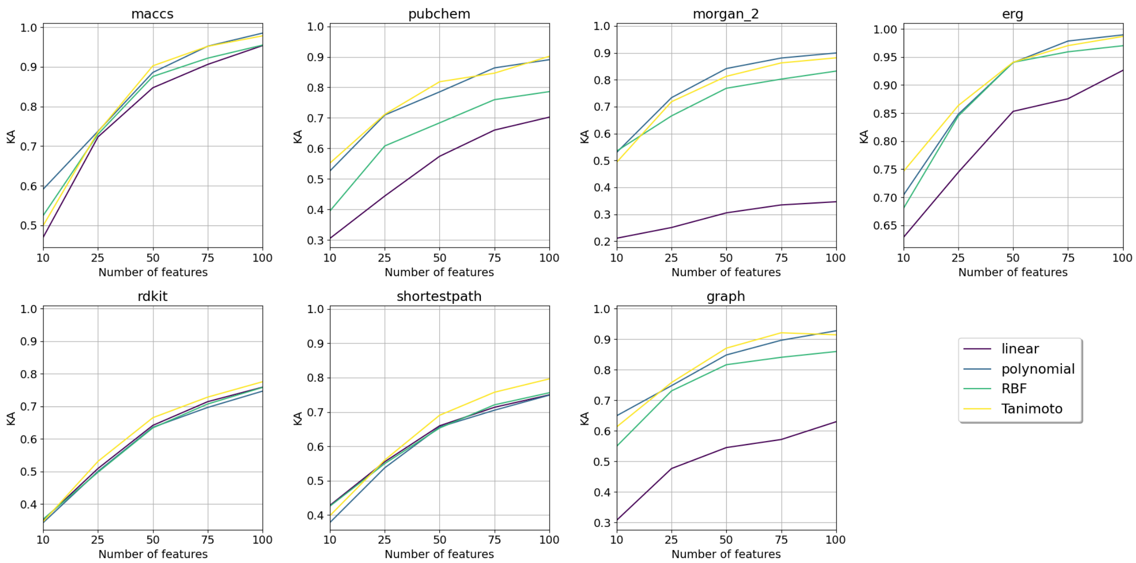


Figure A1: Kernel alignment between Tanimoto kernels of 10 000 sample fingerprint data with all features vs. m features using ProjSe in feature selection with PTC values measured at five different concentrations used as reference data.

B Linear and random forest regression correlation curves

B.1 Linear regression results, multiple concentrations in $\mathbf{Y}$

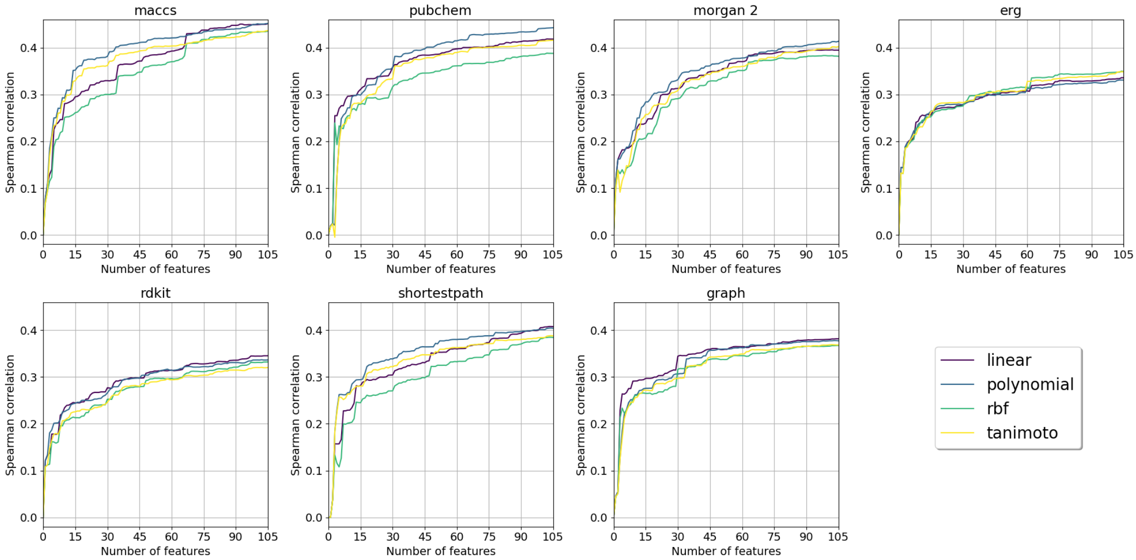


Figure B1: Linear regression. Spearman correlation between predicted PTC-values and ground truth labels for each fingerprint type. PTC-values from multiple concentrations are used to increase the size of $\mathbf{Y}$ in ProjSe to select 105 features in total from each dataset using linear, polynomial, RBF and Tanimoto kernels.

B.2 Random forest regression results, multiple concentrations in $\mathbf{Y}$

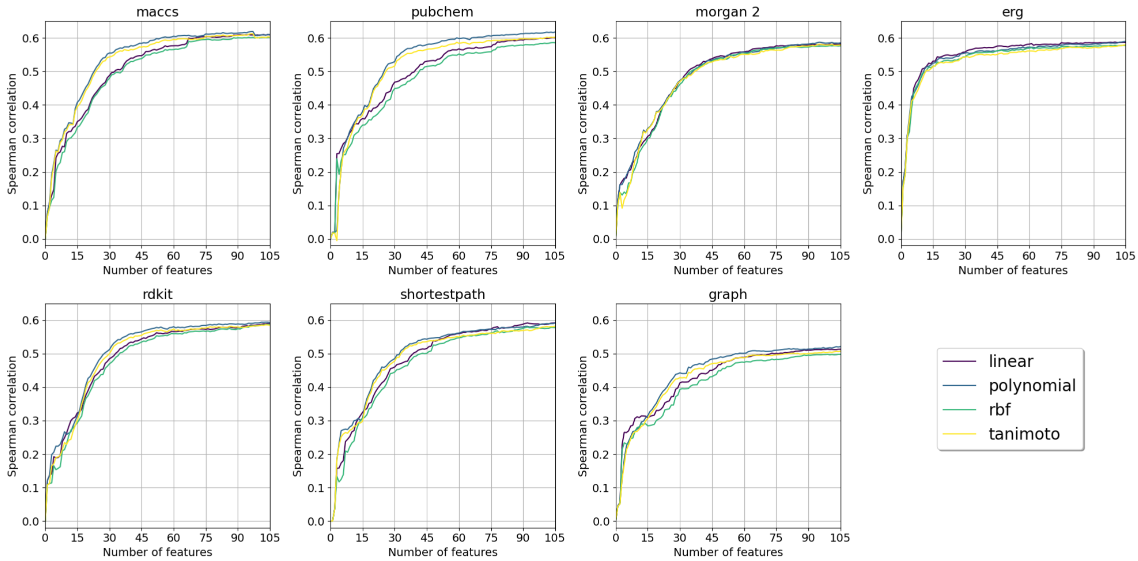


Figure B2: Random forest regression. Spearman correlation between predicted PTC-values and ground truth labels for each fingerprint type. PTC-values from multiple concentrations are used to increase the size of $\mathbf{Y}$ in ProjSe to select 105 features in total from each dataset using linear, polynomial, RBF and Tanimoto kernels.

B.3 Linear regression results, PTC values as 10-bit vectors in $\mathbf{Y}$

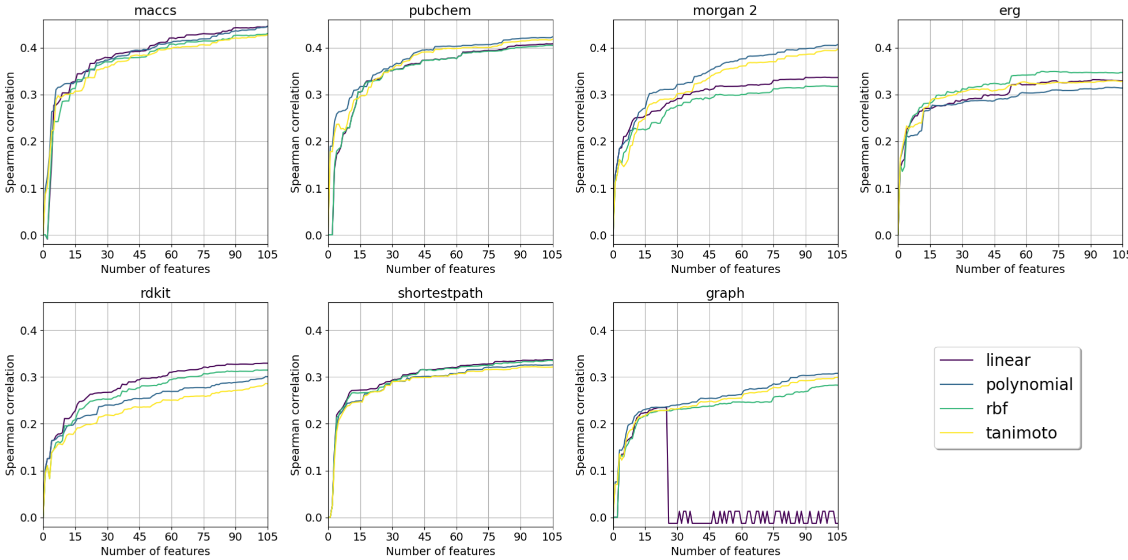


Figure B3: Linear regression. Spearman correlation between predicted PTC-values and ground truth labels for each fingerprint type. PTC-values categorized and binarized to increase the size of $\mathbf{Y}$ in ProjSe. 105 features are selected from each dataset using linear, polynomial, RBF and Tanimoto kernels.

B.4 Random forest regression results, PTC values as 10-bit vectors in $\mathbf{Y}$

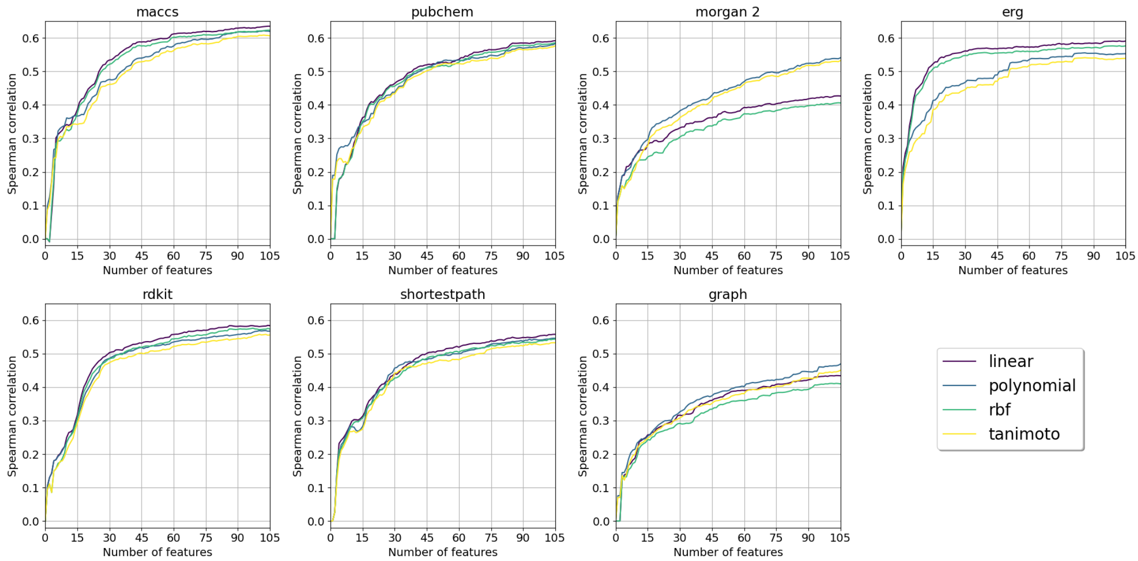


Figure B4: Random forest regression. Spearman correlation between predicted PTC-values and ground truth labels for each fingerprint type. PTC-values categorized and binarized to increase the size of $\mathbf{Y}$ in ProjSe. 105 features were selected from each dataset using linear, polynomial, RBF and Tanimoto kernels.

C Full SVR results for each kernel.

Fingerprint	Spearman correlation			
	lin	poly	rbf	tani
MACCS	0.52	0.51	0.51	0.51
PubChem	0.49	0.48	0.49	0.51
Morgan 2	0.35	0.47	0.45	0.48
ErG	0.40	0.35	0.36	0.21
RDKit	0.43	0.44	0.44	0.44
Shortestpath	0.44	0.48	0.45	0.47
Graph	0.33	0.43	0.40	0.43

Table C1: Results from SVR using samples of 1000 rows in the polynomial kernel when transforming the features into nonlinear ones. SVR failed to converge on ErG fingerprint data.