Examiner's report

Machine Learning in Drug Discovery and Design

Ibraheem Ajibola Ganiyu

12844772
BSc. (Hons) Software Engineering.

School of Computing, Engineering and Mathematics
University of Brighton

Contents

1 Project Evaluation

The title of my dissertation was *Machine Learning in Drug Discovery and Design* with the main focus being on predicting the penetration of drugs into the blood brain barrier.

It is a research-based project that is inspired by my placement spent at AstraZeneca, a biopharmaceutical company, where I gained practical experience in software development for clinical trials and research- based tasks. The project draws on the knowledge I have gained from my programming modules in the second year and also my final year module *CI346 Programming Languages and Client-Server Computing* and also the research skills I have gained both at the university and at AstraZeneca.

A significant amount of the time spent on the project was used to learn the necessary techniques needed to perform machine learning on large datasets and also on the required chemoinformatics knowledge required to successfully complete the project.

2 Background Research

I conducted research into Artificial Intelligence especially Machine Learning before embarking on the project. With little real world knowledge on how to perform machine learning, I turned to machine learning texts such as *Machine Learning by Tom Mitchell* and *Artificial Intelligence: A Modern approach by Peter Norvig.*

For the implementation of the machine learning algorithms, I also researched and experimented with numerous machine learning libraries but eventually settled on the Scikit-Learn and RDKit software libraries, due to their popularity and ease of use for data science. I also read *Chemoinformatics: Concepts, Methods and Tools for Drug Discovery* to get the necessary chemistry knowledge I needed to successfully work with the dataset.

3 Methodology and Planning

As the project is a research-based one, the core requirements or experiments were established before the project life cycle began. The core requirement of the project was to build a machine learning classifier that could be trained with our dataset on how to predict the probabilities of drugs passing through the blood brain barrier. As stated in the interim report, the main stages of the implementation phase with deliverables of the projects are

- Data Cleaning

- Classifier Training
- Ensemble Classifier training
- Deployment
- Testing and Quality Assurance

3.1 Detailed discussion the project stages

3.1.1 Data Cleaning

As is the norm for most data science tasks, the dataset needed to be parsed, unnecessary data removed and the required portions transformed into another form for use.

3.1.2 Classifier and Ensemble Classifier Training

Arguably one of the most important sections, the respective machine learning classifiers chosen, needed to be trained with the processed dataset. Here, four machine learning classifiers (K-Nearest Neighbours, Support Vector Machines, Random Forests and Neural Networks) were trained. In the original project plan, deep neural networks were scheduled to be trained with the dataset but this couldn't be done due to time constraints.
All the trained classifiers were then combined successfully into an Ensemble classifier with an average accuracy of $87\%(+/-7\%)$.

3.1.3 Deployment

In the interim report, it was stated that the project would yield 3 deliverables: a web client, a web server and a prediction API (engine). The ensemble classifier was successfully wrapped in a web server and a REST API was exposed to receive and return prediction results. A web client, however, was not built; This was noted down as an extension that would be done given enough time as it would provide an intuitive and easy accessible UI to make the product easier to use.

3.1.4 Testing and Quality Assurance

All the classifiers were tested using the standard cross-validation technique for evaluating machine learning classifiers and a ROC (Receiver Operating Characteristic) curve was drawn to compare the ensemble classifier against a dummy classifier that is making random guesses.

4 Research Evaluation

The core of the research was to build a classifier to predict the probability of a potential drug candidate passing through the blood brain barrier. This was successfully achieved with an Ensemble classifier that performs at best with an accuracy of 94% and at worst with an accuracy of 80%.

4.1 Problems encountered

One of the common problems encountered throughout the training of the classifiers was achieving accuracy scores lower than 50% on the classifiers. A classifier achieving lower than 50% on prediction results would achieve poorer results than a dummy classifier making random guesses.

Also, due to the high dimensional nature of the dataset, it took longer to train some classifiers, especially the support vector machine classifier as based on read literature we know it would perform poorly on a high dimensional dataset which it did.

Another problem encountered at the beginning during the preprocessing of the dataset was the inability of RDKit to parse some SMILE formats. This is most likely due to the SMILE formats being malformed and probably due to a bug in the RDKit library. However, only a small number of molecule SMILEs (13) , failed to be parsed. This was safely ignored as we still had 2040 molecule SMILEs left that successfully got parsed.

The last problem encountered in the project was combining all the individually trained classifiers into an Ensemble classifier. An option explored was to create a custom class in python where it receives as input the feature vectors and then passes this vector to the individual classifiers and collects their results, however, this resulted in scope creep as doing such as task turned out to be much more difficult than expected.

4.1.1 Solutions put forward

A solution implemented to prevent the classifiers from achieving poor results was to perform feature engineering on the input dataset. An example was the scaling of values in the dataset to have a maximum of 1 and a minimum of -1 using a *MinMaxScaler*. This helped normalize the dataset as some classifiers that utilise distance metrics would return poor results due to the large variance in the dataset. Also, the neural network classifier would converge much faster during gradient descent with a normalised data due to the normalisation.

To help reduce the training time of the support vector machine (SVM) classifier. The Principal Component Analysis (PCA) algorithm was used on the dataset

before feeding it to the SVM classifier. Different values were experimented with to the determine the best number of components to use in the SVM classifiers that would achieve the maximum result whilst still reducing the training time of the algorithm.

The solution to the incompatible ensemble classifier problem was to utilise the *Voting Classifier* class in Scikit-Learn. This had the advantage of being an Ensemble classifier and also had the options of specifying the voting mechanism to use. The voting mechanism here refers to the metric to use when deciding the final prediction result of the Ensemble from the results of the individual classifiers.

4.2 Assessment on the success of the project

Based on the prediction accuracy of the ensemble classifier, the project could be said to be successful. In a paper by Ana Teixeira et al, *A Bayesian Approach to in Silico Blood-Brain Barrier Penetration Modeling*, they listed all the accuracy of the classifiers numerous other researchers have applied to the BBB problem. The lowest result was an accuracy of 74.8% in 2000 using the partial least-squares method whilst the highest was an accuracy of 97.2% in 2007 using a recursive partitioning model and partial least squares method.

This project achieves an average prediction accuracy of 87% (+/- 7%) which lies in between the results mentioned earlier.

5 Areas of Interest

5.1 Prediction API

The prediction API is one of the areas of the project that can provide real world value. The decision to use a REST API was to enable the integration of the Ensemble classifier into any other program or software that the scientist uses in the form of micro-services. A sample use case would be web client that draws a list of molecules from a virtual library micro-service, the user of the client then performs some filtering on the list and sends the remaining molecules to our prediction micro-service, which then returns a list of molecules with their respective probabilities of passing through the blood-brain barrier returned. An example taken from the project is shown below, where a sample curl call is made.

```
$ curl -H "Content-Type: application/json" -X POST -d
'{"smile":"Cn1c2CCC(Cn3ccnc3C)C(=O)c2c4ccccc14"}'
http://localhost:5000/api/prediction
```

And a prediction result is returned as shown in figure 1

```
{
  "category": "p",
  "probability": {
    "n": 0.07729955064888563,
    "p": 0.9227004493511144
  },
  "smile": "Cn1c2CCC(Cn3ccnc3C)C(=O)c2c4ccccc14"
}
```


Figure 1: Sample prediction result for the BBB Rest API

5.2 Learning curve of Ensemble classifier

The learning curve of the ensemble classifier is also another area of interest as shown in figure 2.

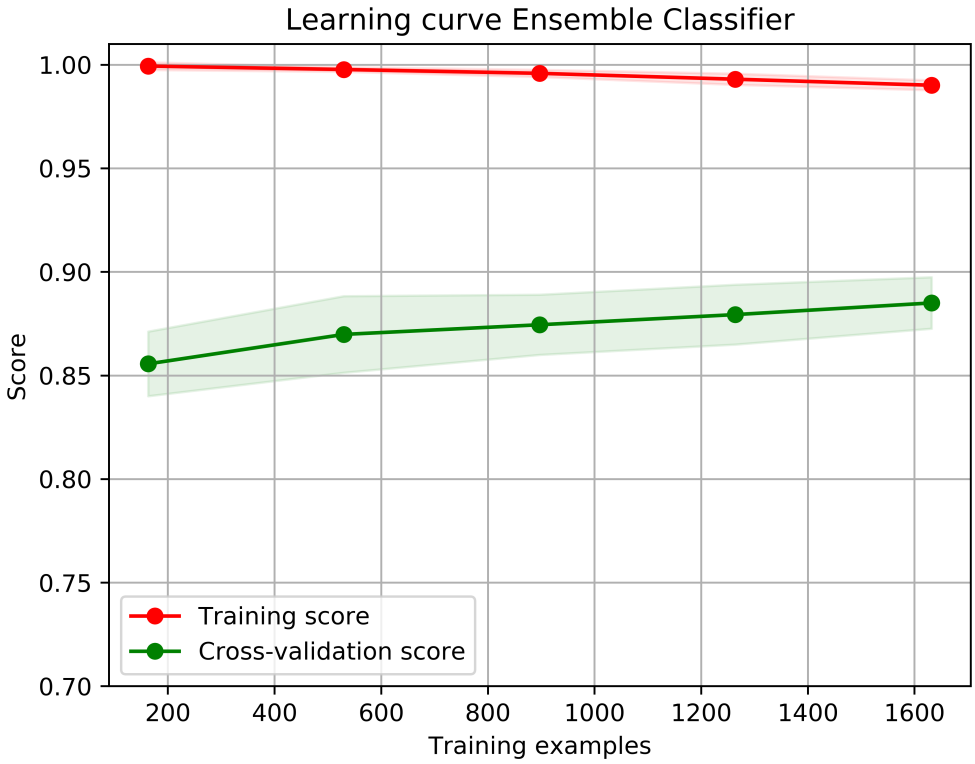


Figure 2: Learning Curve for the Ensemble classifier

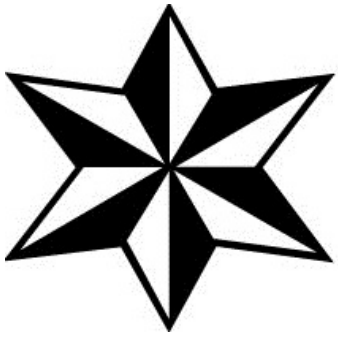
The reduction in training score shows that the classifier over fits less with more data which is ideal as we want it to perform well on new datasets. The cross-validation score also increases with more data, the total number of samples

in the dataset was 2048; Which is a small amount of data compared to large body of chemical datasets available. The assumption here re-enforced by figure 2 is that with more data, we can achieve a higher accuracy score on the ensemble classifier

6 Further Areas for improvement

Given more time, other machine learning classifiers could have been explored, especially deep learning techniques. Based on read literature, they have found a profound and effective use in virtual screening.

Also, the total training time of the classifiers takes roughly about 30 minutes. One of the proposed solutions at the beginning of the project was to implement "Map Reduce" using Apache Spark to train all the respective classifiers on different clusters and combine the result. This would greatly reduce the training time, especially if dealing with a larger dataset.



Machine Learning in Drug Discovery and Design

Predicting the Blood Brain Barrier Penetration of Drugs

Ganiyu Ajibola Ibraheem

A dissertation presented for the degree of
BSc. (Hons) Software Engineering.

12844772
School of Computing, Engineering and Mathematics
University of Brighton

Contents

List of Figures

List of Tables

List of source codes

Abstract

Drug design and discovery is a very expensive process, with numerous new compounds being developed at a rapid rate. Only around 2% of known drugs can pass through the BBB, this presents a problem in the Central Nervous System (CNS) drug development.
This project aims to develop a solution that can predict, with confidence, the probability of a drug passing through the BBB in the hope that this can speed up the process of developing a CNS drug.

Chapter 1

Introduction

Chemical data is growing exponentially as there are currently more than 123 million organic and inorganic substances to date [5], which means for drug development purposes, there is an abundance of chemical data to analyse in search for potential drugs.

Machine Learning is a form of artificial intelligence (AI) that enables computer programs to learn concepts from data without being explicitly programmed. This technique of AI can be applied to problems in many domains, which is what this paper aims to perform, by applying it to chemical datasets to build computer models, aka Virtual Screening, models which can predict with high accuracy, the probability of a drug passing through the blood-brain barrier (BBB) of living organisms.

In Silico models (computer models) have a profound use in virtual screening as they can enable scientist to scan through a large database of drugs to speed up a time-consuming process of analysing drug candidates.

In the context of CNS (Central Nervous System) drug development, when a drug is absorbed into the blood stream, it needs to be able to pass through the Blood Brain Barrier (BBB) to its target which could be the brain or the nervous system, an example would an anti-migraine agent, ergotamine,

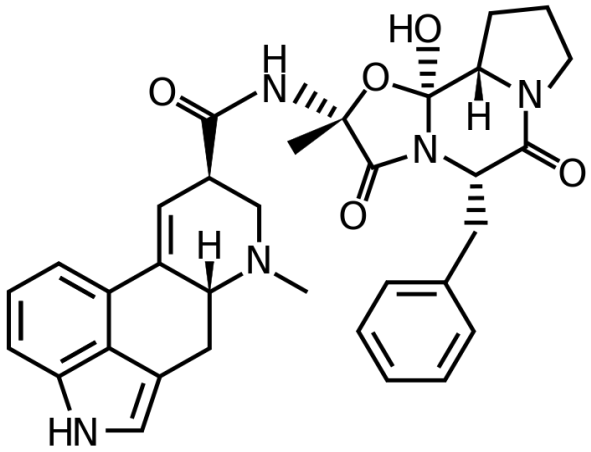


Figure 1.1: Chemical Representation of Ergotamine

which has to pass through the blood-brain barrier to the lining of the brain where it constricts the blood vessels there to decrease the pain from migraine headaches [8].

According to Ana Teixeira et al, 2012 [10], only 2% of the currently known small molecules can pass through the blood-brain barrier, which translates to more time in the drug discovery pipeline spent on analysing drug candidates that can pass through the BBB barrier.

1.1 Proposed Solution

The problem of determining which drug candidates can pass through the blood-brain barrier is a very challenging one and this paper approaches the problem through the use of computer models built with machine learning techniques, applied to a large database of molecules that pass through the Blood Brain Barrier, with the aim of predicting the probability of an unknown candidate drug passing through the BBB.
Ana et al [10] points out that there is a lack of extensive dataset on BBB prediction as most of them are not comprehensive enough to build complex models out of, as a result, they have compiled a dataset of 2040 molecules for use in BBB prediction. Based on analysis carried out [10] by Ana et al, They show that certain machine learning classifiers such as support vector machines and random forests outperform other classifiers, especially for BBB classification tasks. This paper will attempt to utilise that analysis as a baseline for developing our computer models.

Chapter 2

Background

Everything around us is composed of molecules, they are an electrically neutral group of two or more atoms held together by chemical bonds. They are the smallest particle in a compound that exhibits the chemical properties of the compound. Trees can be said to be made up of molecules and historically parts of trees have always been used to cure or alleviate symptoms of illness. Over time, the individual molecules in these herbal medicines were recognized for their effects and they were being produced synthetically, which further gave rise to Modern Drug Design and Discovery.
A potential molecule i.e drug candidate is usually screened against a target protein to test its effectiveness and the screening can either be virtual (Virtual Screening) or through a method known as High Throughput Screening (HTS) - a method of experimentation involving the use of robots and control software to conduct millions of scientific tests; These HTS machines can also be credited with the exponential growth in chemical data [7]. Drug discovery is a very expensive and time-consuming process; It is usually broken down into numerous stages with the most expensive stage being the clinical trials.
The problem of the Blood Brain Barrier (BBB) prediction revolves around Chemistry and Computer Science being applied to the Drug Design domain with a thorough understanding of the constraints imposed by the Central Nervous System. This chapter introduces the necessary concepts needed to understand the solution taken to the BBB prediction problem.

2.1 Cheminformatics

Cheminformatics also known as Chemoinformatics or Chemical Informatics can be visualised as a cross domain of Chemistry and Computer Science. As defined by Brown, it is the mixing of numerous information that a scientist needs to transform

data into information for the intended purpose of making better decisions in drug lead identification and optimisation [4].
The applications of Cheminformatics that are of particular interest are *Storage and Retrieval of Chemical Data* and *Virtual Libraries and Screening* as they both would enable the manipulation and transformation of molecular information for our machine learning algorithms.

2.1.1 Representing Molecules in Computers

The two most common formats for representing chemical molecules in computers are Simplified Molecular-Input Line Entry System (SMILES) and SMART, with SMART being created by Daylight Chemical Information Systems and both formats being actively supported by them [14].

SMILES are specifications in the typographical notation system that describe the structure of a molecule using short ASCII strings and they are more commonly used. Water can be written in the SMILE format as $[OH2]$ An example would be the SMILE representation of protriptyline

$$CNCCCC1c2ccccc2C = Cc3ccccc13$$

This format can then be utilised by cheminformatics software to extract meaningful chemical information about the molecule. Throughout the project, the Open-Source Cheminformatics Software, RDKit, will be used to transform the SMILEs in the dataset and also for the extraction of molecular information, which will be explained below.

2.1.2 Molecular Representation and Similarity

Similar molecules exhibit similar properties [11], with this knowledge, it is possible to approach the BBB problem with the idea that if we know for a given molecule x, that it can pass through the Blood Brain Barrier, then we can expect another given molecule y which is similar to x to pass through the barrier.
This further raises the question of how to represent a molecule (Molecular Representation) in a format that can enable us to compare its similarity to another given molecule. The approach taken was to represent the molecule as a linear vector and then use known statistical techniques to compute the distance between another molecule.

Molecular Representation

For the BBB task, the goal of molecular representation is to create a function $f(x)$ that receives as input, the SMILE representation of molecule x and returns as output a feature vector. The **feature vector** is an n-dimensional vector of numerical features (Real numbers) that represent the molecule.

There are other forms of molecular representation, as Jurgen [2] highlights that other common representations of molecules include encoding the molecular data into a set, graph, vector or function-based representation to enable the use of distance as a form of molecular similarity [2]. Each representation has its advantages and disadvantages, where graphs have their shortcomings, feature-based vectors can prove beneficial depending on the task to the accomplished. Building upon the research of Ana Teixeira et al [10], we would choose to represent the molecules in our dataset as a feature-vector of its chemical descriptors and as a vector of its molecular fingerprint, both which will be explained in the upcoming paragraphs. The Open-Source Cheminformatics Software, RDKit, implements the functionality we will need to extract the chemical descriptors and the molecular fingerprint vectors.

Chemical Descriptors as a Feature Vector: Chemical Descriptors, also known as Molecular Descriptors, as defined by Roberto and Viviana, are the final result of a logic and mathematical procedure which transforms chemical information encoded within a symbolic representation of a molecule into a useful number or the result of some standardized experiment.[21]. The molecule could be represented as a feature vector of its chemical properties which can be its molecular fragments, partial atomic charge, molecular weight, logP etc.
For example, the images shown below in 2.1, are random molecules taken from our dataset. Using the RDKit Software library, we can extract the relevant chemical descriptors. For our dataset, 206 Chemical descriptors were extracted, shown below in table 2.1, are the same random molecules but with the first 5 descriptors displayed.

The beauty of the approach to the BBB problem is that from a statistical standpoint, we do not need a deep knowledge of Chemistry, we can simply make an educated neglect of the molecular names and the headers of table 2.1 and generate the matrix (rounded up to 3 decimal places) shown as the feature vector

$$\begin{bmatrix} -0.256 & 197.102 & 76 & \dots \\ -0.951 & 256.125 & 96 & \dots \\ -0.158 & 387.171 & 144 & \dots \\ \dots & \dots & \dots & \dots \end{bmatrix}$$

of size 2040×206, where 2040 is the number of molecules processed and 206 is the

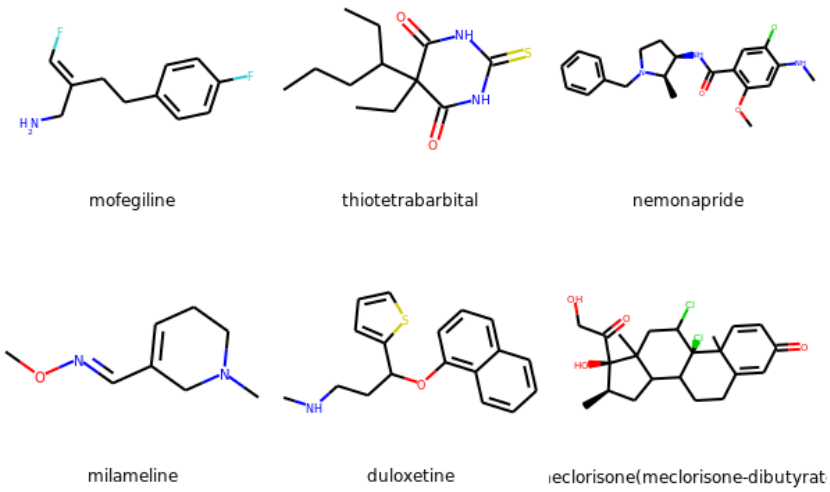


Figure 2.1: Molecular Diagram of Random molecules from the Dataset

Molecule Name	MinEStateIndex	ExactMolWt	NumValence Electrons
Mofegiline	-0.2557111626	197.101605856	76
Thiotetrabarbital	-0.9513194444	256.12454888	96
Nemonapride	-0.1575794701	387.171354752	144
Milameline	0.9790277778	154.110613068	62
Duloxetine	0.1002371504	297.118735228	108
Meclorisone (meclorisone-dibutyrate)	-1.6040011915	426.136464736	154

Table 2.1: Chemical Descriptors of 6 Random molecules from the dataset

total number of chemical descriptors generated. This feature vector would then be utilised by the statistical processing techniques that would be described in the upcoming sections.

Molecular Fingerprints as a Binary-Valued Feature Vector: Another method of representing molecules is as a form of molecular fingerprint. Molecular Fingerprints are an abstract representation of molecular structure and properties of a given molecule in the form of a boolean array [2]. The fingerprint representation is the result of applying a given kernel function to a molecule to generate an n-component bit vector. There are numerous types of formats for a molecular

fingerprint; with the Morgan fingerprint (default fingerprint format for our prediction task) having a 2048 bit representation but the sizes of the vectors can vary depending on the kernel function used.
The vector representation of a fingerprint is given by the general format

$$\vec{V_A} = (v_A(x_1), v_A(x_2), ..., v_A(x_k), ..., v_A(x_n)) \tag{2.1}$$

where x_k indicates the absence or presence of a given feature [2]. i.e

$$v_A(x_k) = \begin{cases} 1 & \text{Feature present} \\ 0 & \text{Feature absent} \end{cases} \tag{2.2}$$

The table 2.2 shows the transformation of a given molecule from our dataset into its SMILE format and then its fingerprint representation using the RDKit software library.

Molecule Name	SMILE	Morgan Fingerprint
111-trifluoro-2-chloroethane	C(CCl)(F)(F)F	[0.0, 0.0, 0.0, . . . , 1.0, 0.0, 0.0, . . . ,0.0, 0.0, 0.0]
Butanone	CCC(C)=O	[. . . , 1.0, 0.0, 0.0, 0.0, . . . , 0.0, 0.0]

Table 2.2: Morgan Fingerprint representation of 2 Random molecules from the dataset

Similarity Measures

The simplest way to cluster molecules together in a chemical database when performing virtual screening is through the concept of *molecular similarity*. It is the core of Molecular Similarity Analysis (MSA) where the similarity measure, that characterizes the degree of proximity between pairs of molecules are manifested by their "molecular patterns", which are compromised of sets of features (chemical descriptors or Morgan fingerprint) [2].
Many different molecular similarity measures exist and some are more suitable to certain virtual screening tasks than another, which is what prompts the discussion of molecular similarity measures in section 2.1.2. To determine the similarity between molecules, some form of similarity measure has to exist to compare molecules in the same representation. The similarity measures (similarity coefficient) are functions that map pairs of compatible molecular representation into a real number.

For our prediction task, the main use of molecular similarity measures is in similarity searching when performing the nearest neighbour search. Here the molecules are ordered by their chosen chemical descriptors when we apply our similarity measure to calculate some form of structural relatedness between a target molecule and every other molecule in the dataset. The result is then a sorted list of molecules where the most similar molecules to our target molecule are located at the top of the list and in order of decreasing similarity.
According to Jurgen [2], the most important components of similarity measures are

- The Representation: Which is used to characterize the molecules that are being compared. An example would be molecular fingerprint representation of a molecule.
- The Weighing Scheme: Used to assign differing degrees of importance to the various components of the molecular representation. An extra measure of accuracy in the weighing schemes would be the use of a chemical ontology database (e.g CheBL) when determining the importance of each component [19].
- Similarity Coefficient: A quantitative measure of the degree of structural relatedness between two molecules.

The main application of any similarity measure hinges on the fact that **structurally similar molecules exhibit similar properties** as stated by Johnson and Maggiora (1990) [11]. However, they also noted that an exception to the rule is that sometimes a small change in the structure of the molecule can result in a radical change in the properties that it exhibits. Which is why Ana Teixeira (2014) [19] notes that the use of a chemical ontology database would be a plausible method to combat this exception. For our prediction task, the exception to this rule is ignored for practical purposes a majority of molecules exhibit similar properties to one another.
In similarity measure calculations, the most common measure for calculating fingerprint similarity is the Tanimoto coefficient, given by

$$Tanimoto(\vec{v_i}, \vec{v_j}) = \frac{\vec{v_i} \bullet \vec{v_j}}{\sum_k v_{ik} + \sum_k v_{jk} - \vec{v_i} \bullet \vec{v_j}} \tag{2.3}$$

where $\vec{v_i}$ and $\vec{v_j}$ are the bit vectors for molecule i and j. There are also other similarity coefficients such as the Tversky and Dice coefficients.

2.2 Drug Design and Discovery

According to Dr A.N Boa [3], the different stages of drug discovery are

- Programme Target Selection (Choosing the disease to work on)
- Identification and Validation of the drug target
- Assay Development
- Identification of a Lead Compound
- Lead Optimisation
- Identification of a drug candidate
- Clinical Trials
- Release of the drug
- Follow-up Monitoring

Majority of the targets for the drugs we consume are usually proteins e.g enzymes, receptors and nucleic acids and the structure of the target is confirmed through a virtual screening method known as *molecular docking*; it can be used to predict how the drug will bind to its target protein through various search/optimisation algorithms [2]. Another Virtual Screening technique usually used is *Quantitative Structure-Activity Relationships* (QSAR), here the underlying idea is that molecules with similar structures behave in the same way, as a result, the activity of a protein against a certain group of compounds is recorded and a QSAR model is the constructed from there and used to determine whether a given compound will bind to the target, thus screening the virtual compound library for drugs of interest.

2.3 Central Nervous System Drug Design and the Blood Brain Barrier

CNS Drugs aiming to pass through the Blood Brain Barrier often need to possess certain physical-chemical properties; some of which are Hydrogen bonding, ionization properties, molecular flexibility etc. [15]. The epithelial cells form an interface between the blood and the brain and these are commonly referred to as the Blood Brain Barrier. This interface occurs in other places within the body but what makes the BBB epithelial cells different are the tight junctions they form which makes it harder for drugs to pass through. Hassan and George (2005) further claim in their article that the majority of BBB penetration is passive diffusion through the cellular membrane.

ADME Properties of CNS Drugs

For a CNS drug to be therapeutically effective, it must be easily disposed asides from having a high degree of potency. The ADME properties of a drug refer to its ability to be easily absorbed, distributed, metabolised and excreted. Some properties of CNS drugs affect their ADME properties, some of which are [15]:

- Solubility: A drug must be very soluble in the blood and still be in high enough concentration at its target, in this case, the Blood Brain Barrier, so that it can easily be absorbed.
- Amount of Protein Binding: Majority of CNS drugs tend to have high binding property towards proteins - this results in the drug being metabolised easily.
- Partition Coefficient (LogP): This is sometimes referred to as the *lipophilicity* of the compound and has served as one of the most important factors in drug design. Higher lipophilicity results in drugs with a higher metabolic turnover but lower solubility and absorption [15].

Chapter 3

Machine Learning Classifiers

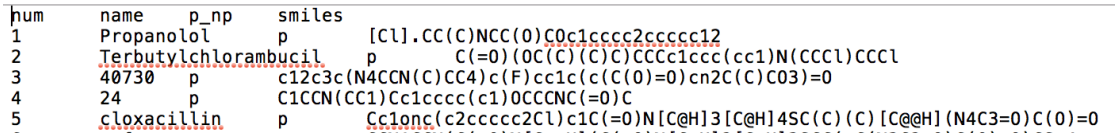
Machine Learning is a sub-field of Artificial Intelligence that involves the design of algorithms that can learn, make predictions and improve based on data without being explicitly programmed. There are 4 main categories of machine learning algorithms, namely [12]

- Supervised Learning: Here the machine learning algorithms are provided with training data where the correct outputs to the inputs are labelled and the algorithm learns to generalise from the training data.
- Unsupervised Learning: This is the opposite of supervised learning, the training data contains only the inputs with no outputs; the algorithm then learns to generalise and cluster similar inputs together.
- Reinforcement Learning: This lies in between supervised and unsupervised learning as no correct labels are provided to the training data, however, the algorithm is corrected when it makes a wrong prediction and it is required to iterate over numerous possibilities till it makes the correct prediction
- Evolutionary Learning: This is a bio-inspired model where the concept of evolution is applied, numerous solutions are developed and are evaluated based on a fitness score. Models with a higher fitness scores progress to the next training iteration whilst the lesser models are dropped till an optimal solution is found.

For the blood-brain barrier (BBB) task, supervised learning techniques would be utilised for prediction purposes whilst the unsupervised learning techniques would be used for data exploratory purposes and to further understand and highlight patterns in the BBB dataset. The Scikit-Learn software library would be used extensively for the implementation of the supervised and unsupervised learning models.

3.1 Data Representation and Preprocessing

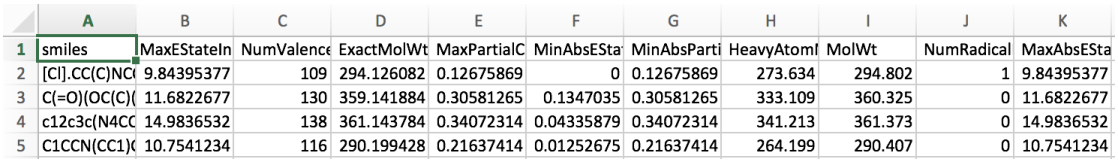
The Pandas and Numpy libraries are used throughout the project and they are also the underlying library used by Scikit-Learn. In code listings 1, the training data shown in figure 3.1 is loaded using Numpy and then converted into the RDKit molecular format.



num	name	p_np	smiles
1	Propanolol	p	[Cl].CC(C)NCC(O)COc1cccc2ccccc12
2	Terbutylchlorambucil	p	C(=O)(OC(C)(C)C)CCCc1ccc(cc1)N(CCCl)CCCl
3	40730	p	c12c3c(N4CCN(C)CC4)c(F)cc1c(c(C(O)=O)cn2C(C)CO3)=O
4	24	p	C1CCN(CC1)Cc1cccc(c1)OCCCNC(=O)C
5	cloxacillin	p	Cc1onc(c2ccccc2Cl)c1C(=O)N[C@H]3[C@H]4SC(C)(C)[C@@H](N4C3=O)C(O)=O

Figure 3.1: Image of some sample rows from the dataset

The failed indices are recorded and deleted from the Numpy array and the resulting data is then transformed based on their chemical descriptors and stored in a Numpy array for use by the supervised and unsupervised models, with an example of the processed results shown in figure 3.2.



	A	B	C	D	E	F	G	H	I	J	K
1	smiles	MaxEStateIn	NumValence	ExactMolWt	MaxPartialC	MinAbsESta	MinAbsParti	HeavyAtomI	MolWt	NumRadical	MaxAbsESta
2	[Cl].CC(C)NC(	9.84395377	109	294.126082	0.12675869	0	0.12675869	273.634	294.802	1	9.84395377
3	C(=O)(OC(C)(	11.6822677	130	359.141884	0.30581265	0.1347035	0.30581265	333.109	360.325	0	11.6822677
4	c12c3c(N4CC	14.9836532	138	361.143784	0.34072314	0.04335879	0.34072314	341.213	361.373	0	14.9836532
5	C1CCN(CC1)(	10.7541234	116	290.199428	0.21637414	0.01252675	0.21637414	264.199	290.407	0	10.7541234

Figure 3.2: Image of some sample rows from the processed dataset

It is worth noting the dataset used is a highly biased one; After preprocessing and cleaning, there is a total of 1563 molecules that pass through the blood-brain barrier and a total of 477 molecules that do not pass through the barrier. This bias would be accounted for during the training phase as the machine learning classifiers would be cross validated as explained in the model evaluation section.

3.2 Unsupervised Learning

Before training the classifiers with the preprocessed data; it is usually best to carry out some form of exploration into the dataset to enable educated guesses about the best machine learning models to utilise for the prediction task. In this section, unsupervised learning techniques would be used to gain insights into the BBB dataset with the aim being to determine if it is possible to linearly fit a classifier that can symbolically divide between the molecules that pass through the barrier and the ones that do not. The main areas focused on would be the transformation

```
import numpy as np
from rdkit import Chem
from rdkit.Chem import Descriptors
from rdkit.ML.Descriptors import MoleculeDescriptors

def load_data(from_url,desc_url):
    data = np.genfromtxt(from_url,dtype="i4,U256,U256,U256",
    ↪ comments=None,skip_header=1,names=['num','name','p_np','smiles'],
                         converters={k: lambda x: x.decode("utf-8")
    ↪ for k in range(1,4,1)})
    fail_idx = []
    for idx,entry in enumerate(data):
        smiles = entry[3]
        molecule = Chem.MolFromSmiles(smiles)
        if molecule is None:
            fail_idx.append(idx)
            continue
    data = np.delete(data,fail_idx)
    print("Fail Count: ", len(fail_idx))
    print("{} molecules used in the
    ↪ calculations".format(len(data)))
    return calc_descriptors(desc_url, data)

def calc_descriptors(file_url,data):
    chem_descriptors = [desc[0] for desc in Descriptors._descList]

    calculator =
    ↪ MoleculeDescriptors.MolecularDescriptorCalculator(chem_descriptors)
    print("Using",len(chem_descriptors), "chemical Descriptors")

    with open(file_url,'w') as f:
        f.write("smiles," +
                ", ".join(["{}".format(name) for name in
    ↪ calculator.descriptorNames]) +
                ",p_np\n")
        for entry in data:
            smiles = entry[3]
            molecule = Chem.MolFromSmiles(smiles)
            print("Calculating chemical descriptors for",smiles)
            f.write( smiles + "," +
                    ", ".join(["{}".format(value)
                               for value in
    ↪ calculator.CalcDescriptors(molecule)]) +
                    ",{}\n".format(entry[2]))
    return data
```

Listing 1: Functions to load, preprocess and transform the dataset

of the dataset using the Manifold learning algorithms and the clustering of the data points.

3.2.1 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) is a form of dimensionality reduction technique that aims to reduce the number of dimensions of the data mainly for visualisation purposes whilst still preserving the statistical meaning of the data and reducing it to its important (principal) components; Here only a small subset of the features of the data are selected according to how important they are;

To illustrate further using the feature vector shown in 2.1.2, the vector has a dimensional size of 206 features, as all the 2040 molecules have a chemical descriptor size of 206. Plotting a single molecule on a graph with 206 dimensions would be nearly impossible as 2D and 3D data are the best ways to illustrate statistical data for humans. By applying the Principal Component algorithm to the feature vector, it proceeds to find the direction of maximum variance for each molecule that is, it returns a new vector that contains the most information. The directions of maximum variance returned are referred to as the *principal components*.

In order to gain a deeper understanding of our data, the PCA algorithm is applied with the number of principal components set to 2 i.e (n_components = 2). In the 2D representation shown below in figure 3.3, the data points are tightly clustered in the middle, and whilst this looks like it is linearly inseparable, with some clustering techniques that would be shown later, a better representation where it is more distinguishable between the data points would be shown.

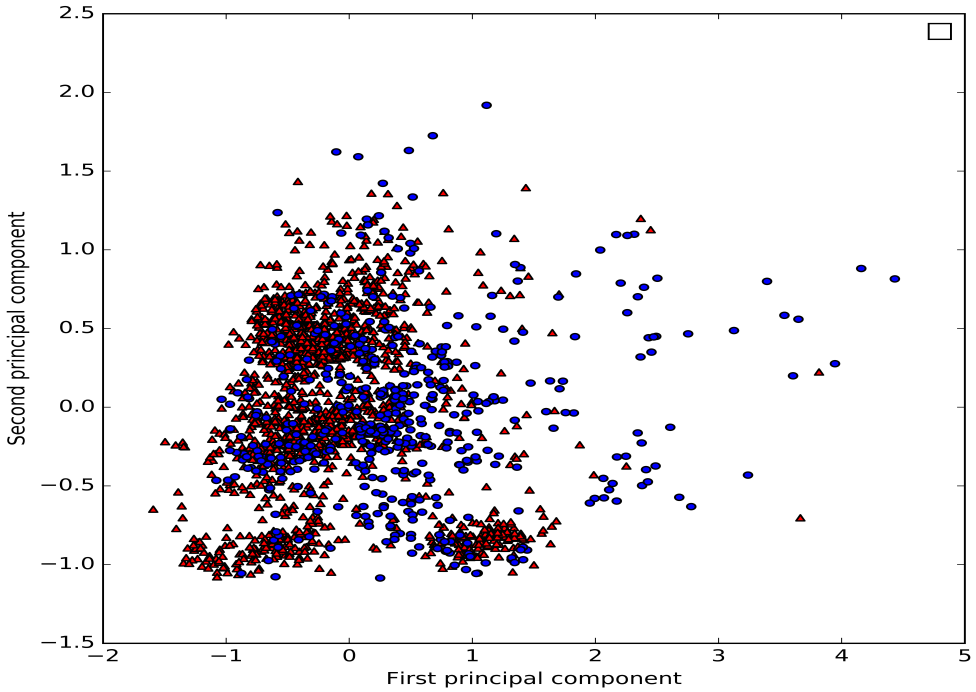


Figure 3.3: 2D Scatter Plot after applying the PCA Algorithm

To further examine the data, the next 3 principal components are selected to create a 3D plot. The 3D plots are shown in figures 3.4 and 3.5 also has the same clustering of a significant proportion of the blue molecules in the same space as the red molecules but looks more linearly separable than the 2D plot. As would be shown in the upcoming sections, with a little bit of feature engineering, it would be possible to separate the data points into 2 clusters of red and blue i.e molecules that pass through the barrier and the ones that do not.

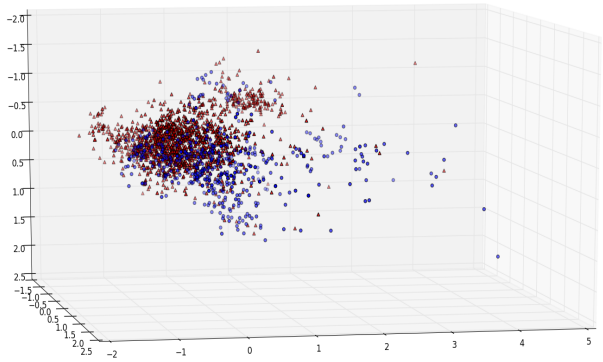


Figure 3.4: 3D Scatter Plot after applying the PCA Algorithm (I)

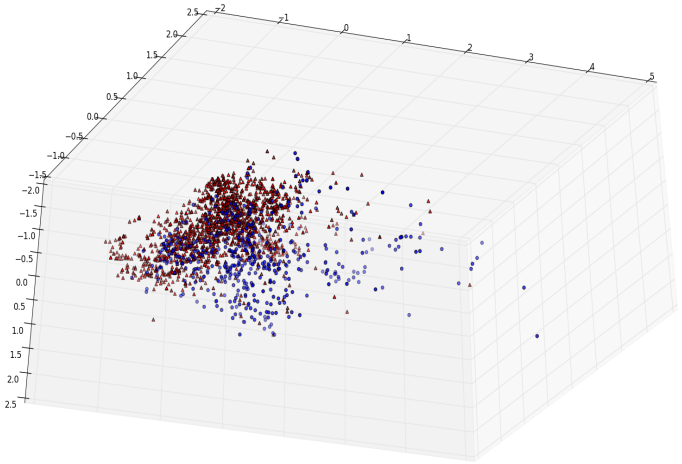


Figure 3.5: 3D Scatter Plot after applying the PCA Algorithm (II)

A major benefit of exploratory data analysis as we have seen here is that it is

worth gaining insights into the dataset to determine the kind of feature engineering required to achieve high performing machine learning models.

3.2.2 Manifold Learning with t-SNE

Manifold Learning algorithms are also another form of dimensionality reduction techniques similar to the Principal Component Analysis algorithm. The t-SNE algorithm, a manifold learning technique, would also be used to further visualise the data in another attempt to further see if it is visually possible to separate between the molecules that do and do not pass through (the red and blue dots).

Manifold Learning Algorithms excel at data visualisation tasks, they provide more complex mappings and often better data representations but perform poorly for data classification tasks. They perform poorly because they do not allow transformations of new data once they have been fitted [22].

The t-SNE algorithm is used because it tries as much as possible to maintain the distance between the data points in the original feature vector space during transformation; This way the feature vector can be transformed and reduced to better suit our data exploratory purposes whilst still preserving the integrity of the data. It puts more emphasis on points that are close by rather than preserving points that are far apart [1].

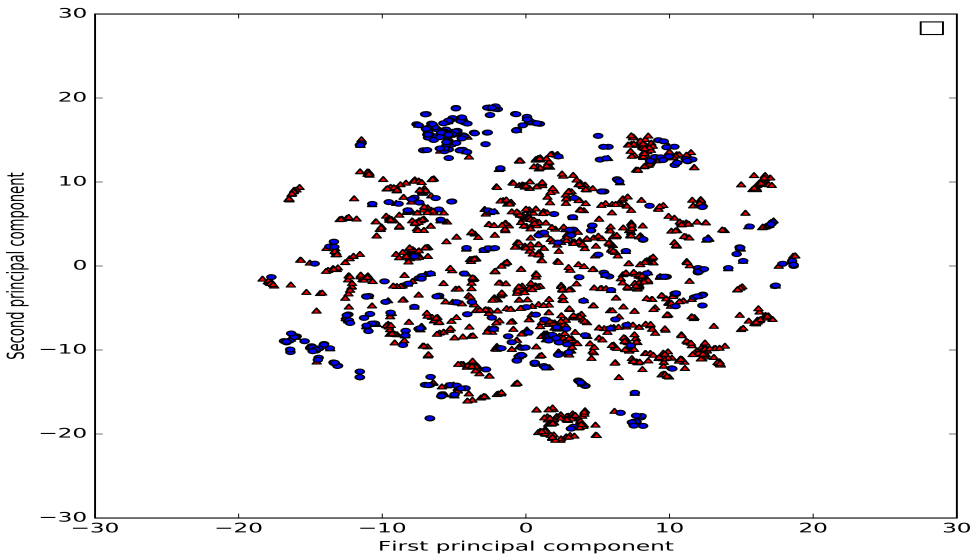


Figure 3.6: 2D Scatter Plot after applying the t-SNE Algorithm (Molecular Descriptor dataset)

Looking at the t-SNE scatter plot, there is a small cluster of the molecules that do not pass through (B) at the top and some randomly distributed throughout the plot, whilst the ones that pass through the BBB barrier (A) maintain an even distribution around the feature space. This result is to be expected as the data is biased towards molecules that cross the barrier and it also hints at the fact that the molecules that do not cross the barrier might share a lot in common with molecules that cross the barrier.

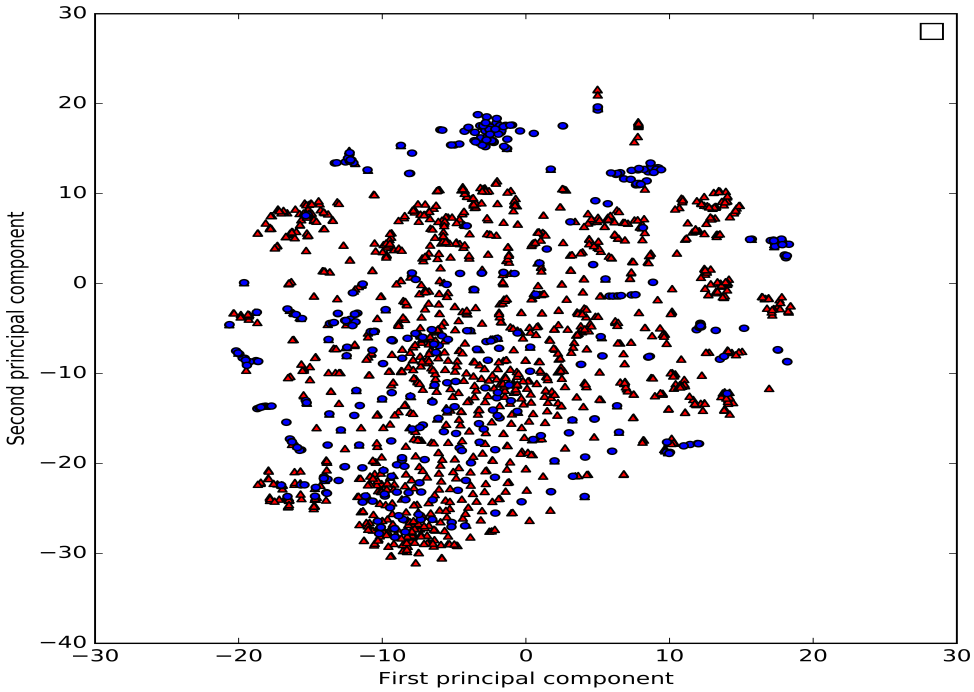


Figure 3.7: 2D Scatter Plot after applying the t-SNE Algorithm (Morgan Fingerprint dataset)

The scatter plot has no significant difference from the Morgan Fingerprint scatter plot shown in figure 3.7, it highlights the clusters of the B molecules more visibly. Further analysis would need to be carried to create a more linear separable representation.

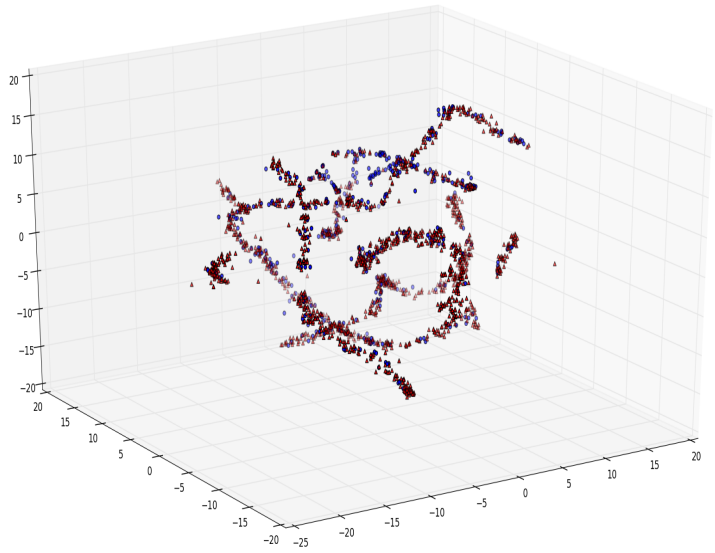


Figure 3.8: 3D Scatter Plot after applying the t-SNE Algorithm (Molecular Descriptor dataset)

The 3D scatter plot of the simple molecular descriptor data looks slightly more separable than its 2D counterpart.

3.2.3 Clustering

K-Means Clustering

In our further attempts to have some form of linearly separable data, we further explore more clustering techniques. Here, the 2 principal components of our feature vector are selected and applied to the k-means clustering algorithm.

The K-means algorithm, also known as *Lloyd's algorithm*, aims to find cluster centroids that are representative of regions within the dataset or feature vector. The algorithm selects the approximate center of the clusters and iteratively assigns each data point to this center, and then sets the cluster center again as the average

of the data points in the cluster using the equation 3.1.

$$\sum_{i=0}^{n} min_{\mu_j \in C}(||x_j - \mu_i||^2) \tag{3.1}$$

where C is the cluster, μ_j is the mean of the cluster, x_j is the current data point and μ_i is the current mean of the calculation [18]. It repeats this process until the cluster centroids do not change significantly.

Applying the algorithm to a feature vector derived from the Morgan fingerprint dataset. We now have a visualisation showing two clusters with their centroids in yellow, shown in figure 3.9. This is a more linearly separable diagram as the molecules that pass through the barrier shown in red are clustered around their centroid and also the molecules that do not pass through.

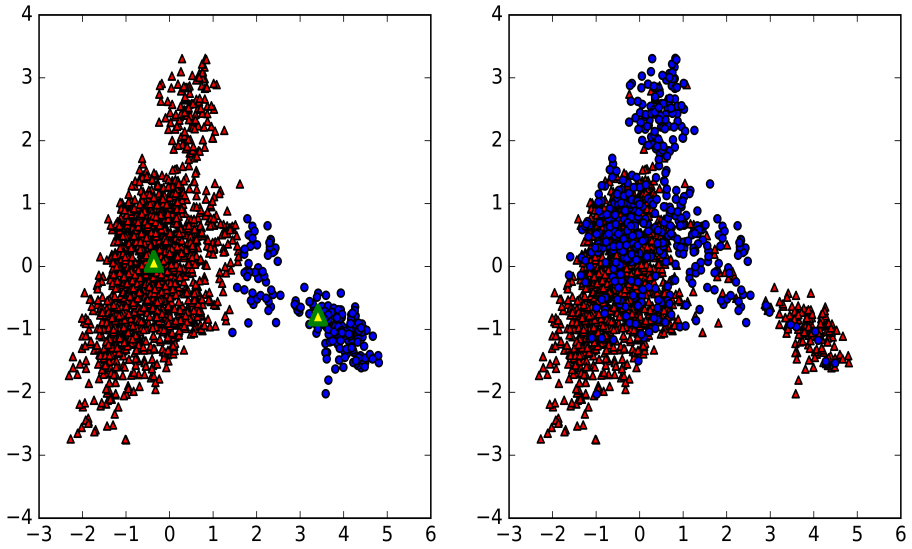


Figure 3.9: 2D Scatter Plot found by k-means on fingerprint dataset. *Original data points on the right and clusters found on the left*

Agglomerative Clustering

These are a form of hierarchical clustering techniques with major applications in natural language processing. Hierarchical techniques form clusters by continuously merging or splitting data points till it meets the required number of clusters. Agglomerative clustering works in a similar manner where each data point is a

cluster of its own and iteratively the algorithm merges the clusters together using a *merging strategy*.

The merging strategy used for our dataset was the **Ward Strategy**, where the algorithm minimises the sum of squared differences within all the clusters, which is similar in a way to the K-Means algorithm.

Figures 3.10 and 3.11 show the result of applying the algorithm to a feature vector derived from the two principal components of the simple molecular descriptors and the morgan fingerprint descriptors.

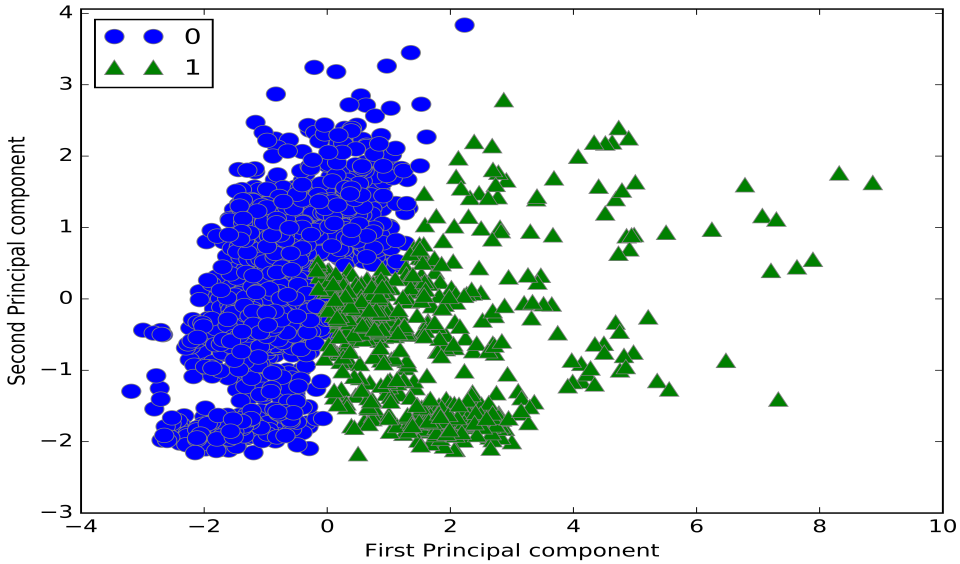


Figure 3.10: 2D Scatter Plot of clusters found by the agglomerative ward algorithm on the simple molecular dataset

With the blue circles being the molecules that pass through the barrier and the triangles being the molecules that do not pass through the barrier.

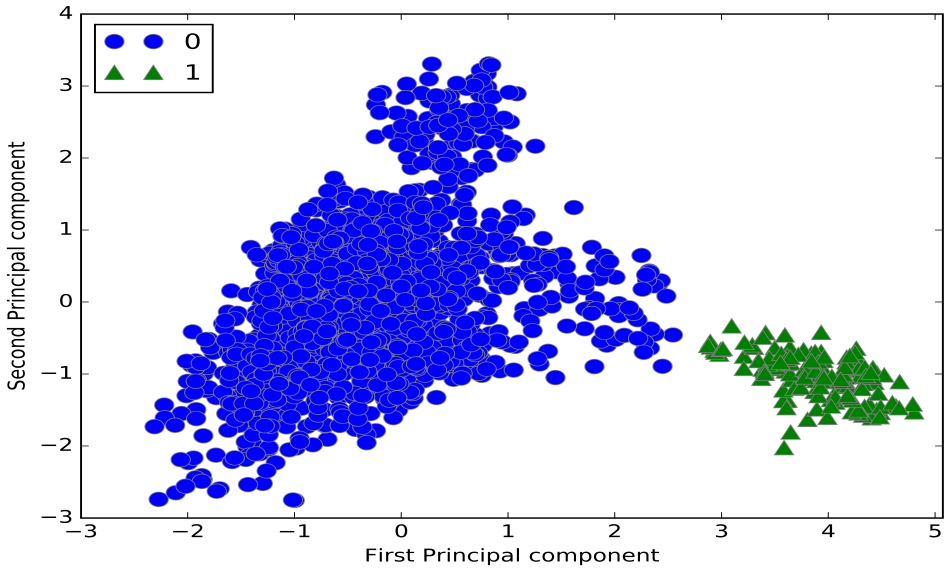


Figure 3.11: 2D Scatter Plot of clusters found by the agglomerative ward algorithm on the Morgan Fingerprint dataset

3.3 Machine Learning Models

3.3.1 K-Nearest Neighbours

K-Nearest Neighbours (k-NN) is a form of instance-based learning; Instanced-based learning techniques store the training example in memory and utilise them to make predictions. All data points (nearest neighbours) in the feature vector are defined in a form of euclidean space; where a prediction for an instance is made by finding its euclidean distance to its nearest k neighbours.

The k-NN training and classification Algorithm [13]

- **Training Algorithm**
  For each training example v, add it to the list of *training_examples*
  - Ensure the elements of the training sample are sorted using the

distance function

$$d(x_i, x_j) \equiv \sqrt{\sum_{r=1}^{n} (a_r(x_i) - a_r(x_j))^2} \tag{3.2}$$

where $a_r(x)$ denotes the rth attribute of instance x and x_i, x_j are two arbitrary instances to be compared

- **Classification Algorithm**
  Given a query instance x_q to be classified

  Let $x_1, \ldots, x_k$ represent k instances from the *training_example* that are nearest to x_q.

  – Return the class of x_q denoted by $\hat{f}(x_q)$; given by

$$\hat{f}(x_q) \leftarrow argmax_{v \in V} \sum_{i=1}^{k} \partial(x_i) \tag{3.3}$$

  where V is set of all possible classes an instance x can belong to and $\partial(x_i)$ represents the class of the current instance x_i from the k instances.

Applying the k-NN algorithm to the simple molecular dataset yields the results shown in figure 3.12. The reduction in training accuracy as the number of neighbours increases is a good indicator that the model is not overfitting; Overfitting occurs when the model memorises the training data instead of learning the statistical patterns within them.

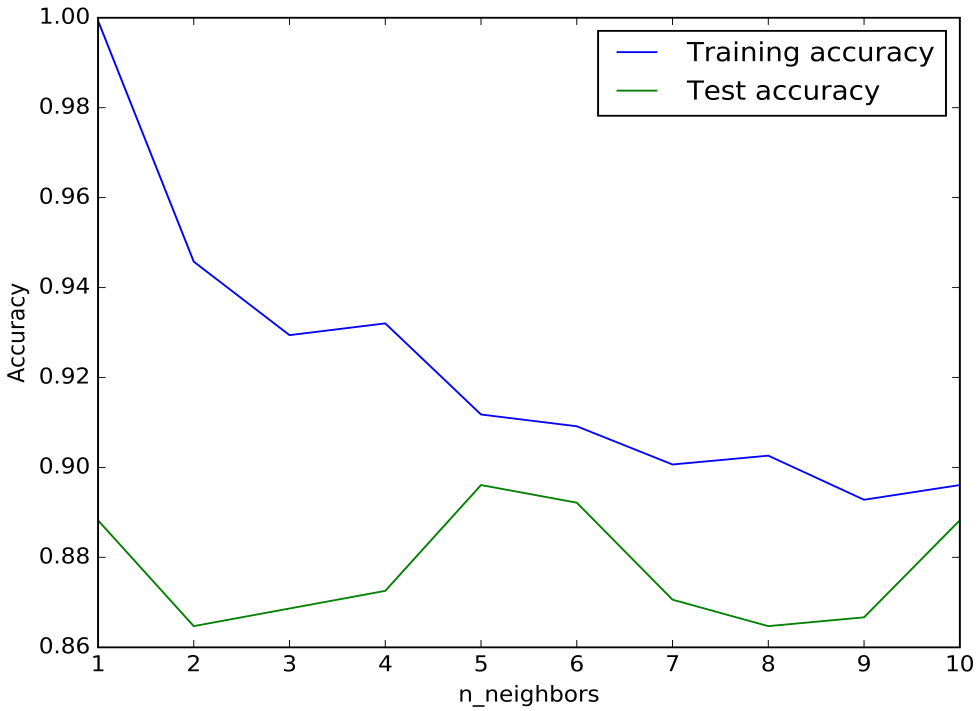


Figure 3.12: k-NN classifier for 10 neighbours using the Simple Molecular Descriptors

The algorithm is also applied to the morgan fingerprint dataset, shown in figure 3.13. There is little or no improvement in accuracy between both datasets with the number of neighbours at 10

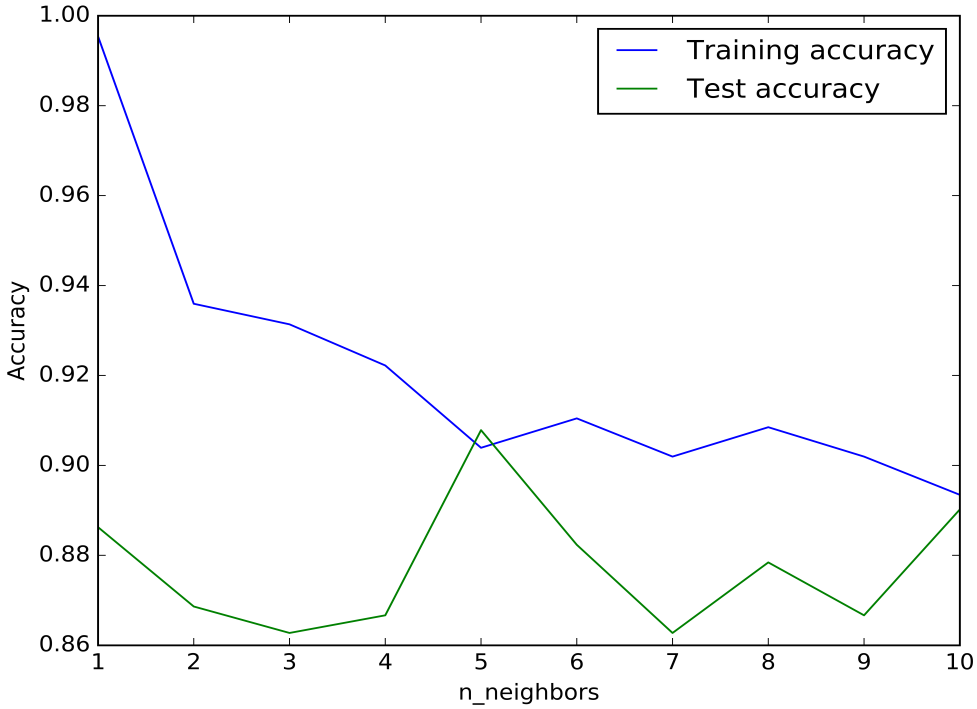


Figure 3.13: kNN classifier for 10 neighbours using the Morgan Fingerprint dataset

3.3.2 Support Vector Machines

Support Vector Machines (SVMs) are a set of supervised learning techniques that make predictions by learning the statistical hyper-plane (support vectors) that determines the categories of data in the input space. They are very effective in high dimensional situations as is the case with our dataset (2048 x 206) and also very memory efficient; as only a subset of the training points in the decision function i.e support vectors are stored in memory.

SVMs are computationally intensive and as a result, they do not scale well with large datasets; The driving idea behind SVMs is to infer the hyper-plane that divides the data into categories. Applying this to our BBB dataset, the goal is to have the SVM learn the support vectors that classify a molecule as either positive (passes through the barrier) or negative. Figure 3.14 shows the result of applying the SVM classifier to the simple molecular dataset.

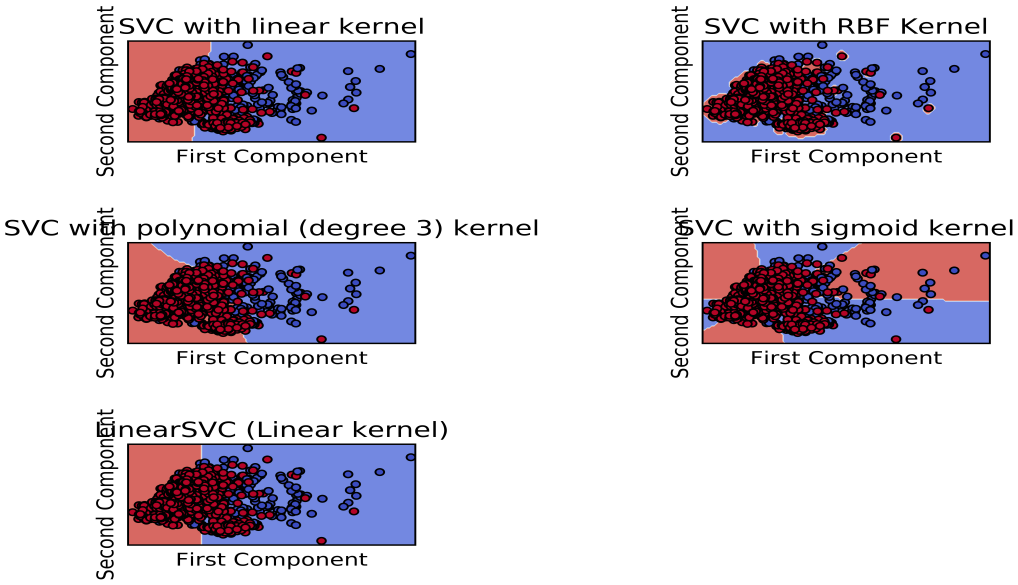


Figure 3.14: Support Vector Machines with different kernel functions on Simple Molecular Dataset

The figures show the algorithm applied on our datasets using numerous kernel functions. Kernel functions are basically a form of similarity functions, they compute the distances between data points in the input space. To enable the SVM classifier to make predictions for a new data point, its distance to the support vectors are calculated using any of the kernel functions (linear, polynomial, sigmoid or Radial Basis function (RBF)) [12]. Due to the highly mathematical nature of these kernel functions, their intricate details can be safely ignored and their classification results instead focused on for our BBB classification tasks.

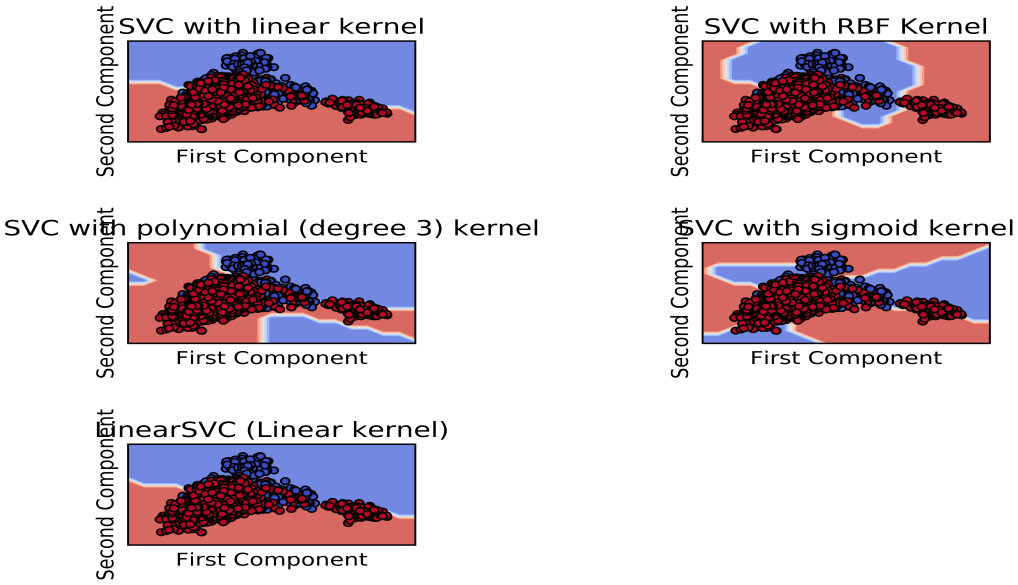


Figure 3.15: Support Vector Machines with different kernel functions on Morgan Fingerprint Dataset

3.3.3 Decision Trees

Decision trees are an example of non-parametric supervised learning algorithms; The aim of the algorithm is to have a model that can classify an input data (feature vector) based on simple rules the algorithm has inferred from the features of the dataset. Here observations learnt from the dataset are represented in branches (decision boundaries), which then lead to classifications of the input represented in the leaves.

To illustrate further, an example from the BBB classification is shown in figure 3.16. The overview of the algorithm is that it learns the necessary if/else questions to classify an input.

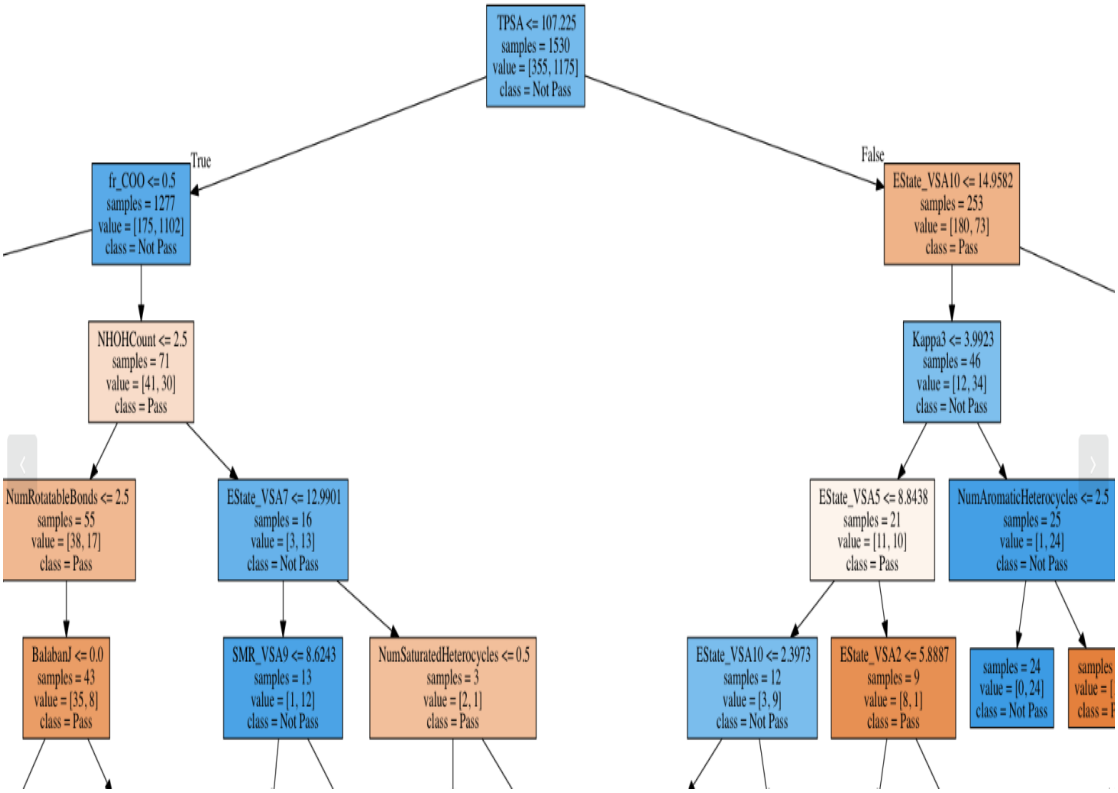


Figure 3.16: Decision Tree

Here in figure 3.16, the algorithm performed a search over all the training data and picked *TPSA* (total polar surface area) as the most informative variable in deciding whether a molecule passes through the barrier or not; Splitting the tree into 1277 molecules that do not pass and 253 molecules that pass. The algorithm then recursively searches each branch and further creates more branches till it reaches a node (leaf) that contains a single target value (target classification).

Figure 3.17 shows the generated tree after applying the decision tree algorithm to our simple molecular dataset; With the algorithm achieving an accuracy of $\sim 85\%$ on the simple molecular dataset.

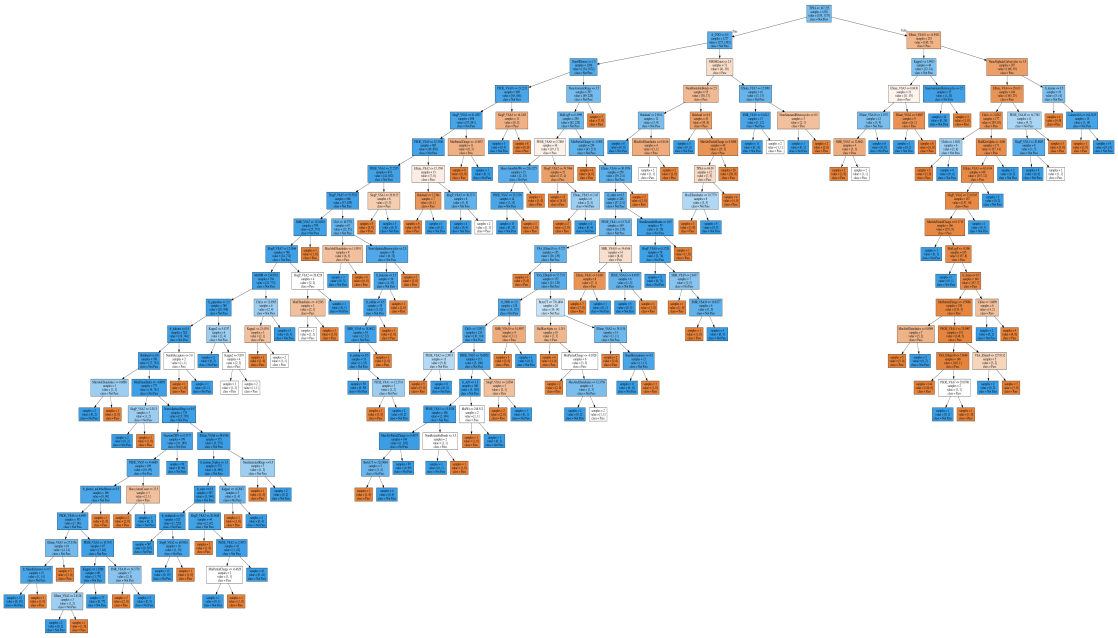


Figure 3.17: Decision Tree: Simple Molecular Descriptors

Ensembles of Decision Trees Ensembles in the context of machine learning refers to the combination of numerous machine learning models to create a more powerful model. This technique would be utilised in section 3.3.5, where all the machine models that would be discussed will be combined to create an even more powerful and stable model.

Decision trees tend to overfit and learn the rules only in the training data and perform poorly when presented with new data [1]. To overcome this problem, numerous decision trees would be combined to create an ensemble of decision trees: random forests, gradient boosted decision trees, extra trees and Adaboost decision trees.

Random Forests

Random forests are an ensemble of decision trees where each decision tree has been trained with a subsection of the dataset with the knowledge that the said tree would overfit on that part of the dataset. The overfitting of the random forest is reduced by averaging the result from all the individual trees to return a classification.

An accuracy of $\sim 87\%$ was achieved using the random forest classifier on the simple molecular descriptor dataset whilst an accuracy of $\sim 89\%$ was achieved on the Morgan fingerprint dataset; With the number of trees in the random forest set

to 20.

Extremely Randomized Trees

Extremely randomized trees also known as *Extra Trees* are very similar to Random Forests. The difference between them is that at each step of the algorithm the entire dataset is used as opposed to partial datasets in random forests and the decision boundary is also chosen at random instead of the best one [16].

The extra trees algorithm performed slightly better than the random forests algorithm; On the simple molecular dataset, the extra trees score an accuracy of $\sim 90\%$ and on the Morgan fingerprint dataset, the extra trees score an accuracy of $\sim 89\%$, performing just slightly less than the random forest on the same dataset with the number of extra trees was set to 20.

Gradient Tree Boosting

Gradient boosted trees differ from random forests in the sense that in gradient tree boosting, very small trees called *weak learners* are trained serially where the current tree learns from the mistakes of the previous tree. The weak learners usually have a shallow dept of 5 and can, therefore, make faster predictions. The learning rate is also another parameter of the gradient boosted trees that can be optimised; the learning rate is the degree to which the current tree learns from its predecessors [1] with a higher rate resulting in stronger corrections.

Applying the gradient boosted trees algorithm to both derivations of our dataset with the number of estimators set to 150 and the learning rate to 0.45. An accuracy score of $\sim 86\%$ was achieved on the simple molecular descriptors and $\sim 82\%$ on the Morgan fingerprint dataset.

3.3.4 Neural Networks

A Neural Network is a computational model inspired by the human brain, it consists of a network of neurons connected together by axons. The inspiration behind the model is that if learning occurs in the brain and if we can represent this in a computational model then we can achieve some form of intelligence in our systems.

Multi Layer Perceptron

The Perceptron can also be referred to a collection of neurons along with a vector of inputs and a vector of weights to fasten the inputs to the neuron. This neuron in this network consists of its inputs, weights, threshold and activation function. Each neuron, receives an input along with the strength of the input i.e weight, which it multiplies together and if the value is greater than the threshold, the

neuron fires.
For our dataset the input $X = [x_1, x_2, x_3, ..., x_n]$ to a Perceptron could be either a simple molecular feature vector value e.g the number of hydrogen atoms in the molecule or vectors of 1s and 0s if we're using the Morgan fingerprint dataset to train the network. Each neuron is also given a vector of small random positive and negative values as its weights. The neuron calculates its activation value as

$$h = \sum_{i=1}^{m} w_i x_i \tag{3.4}$$

This activation value is then passed on to the activation function $f(h)$ which fires if the activation value is greater than the threshold. A threshold value of 0 was used irrespective of the training dataset. The output value of the function is given by

$$output = f(h) = \begin{cases} 1 & \text{if h}>\theta \\ 0 & \text{if h} \leq \theta \end{cases} \tag{3.5}$$

Multi Layer Perceptron The Input layer to the neuron is the same as the length of the feature vector plus one (the fixed bias input) as shown in figure 3.18. For the simple molecular feature dataset, the number of inputs n to the network is 206, as that is the number of simple chemical descriptors that were calculated. For the Morgan Fingerprint dataset, $n = 2048$ where n represents the length of the bit vector.

The bias input is used as a control value to adjust the value that the neuron fires at. An example is when the feature vector to a neuron is all zeros and the neuron should fire. By our activation function, the neuron wouldn't fire because its activation value does not exceed the threshold. By adding a fixed value (bias) of ± 1 to the feature vector, we can then adjust the weight to make the neuron fire or not.

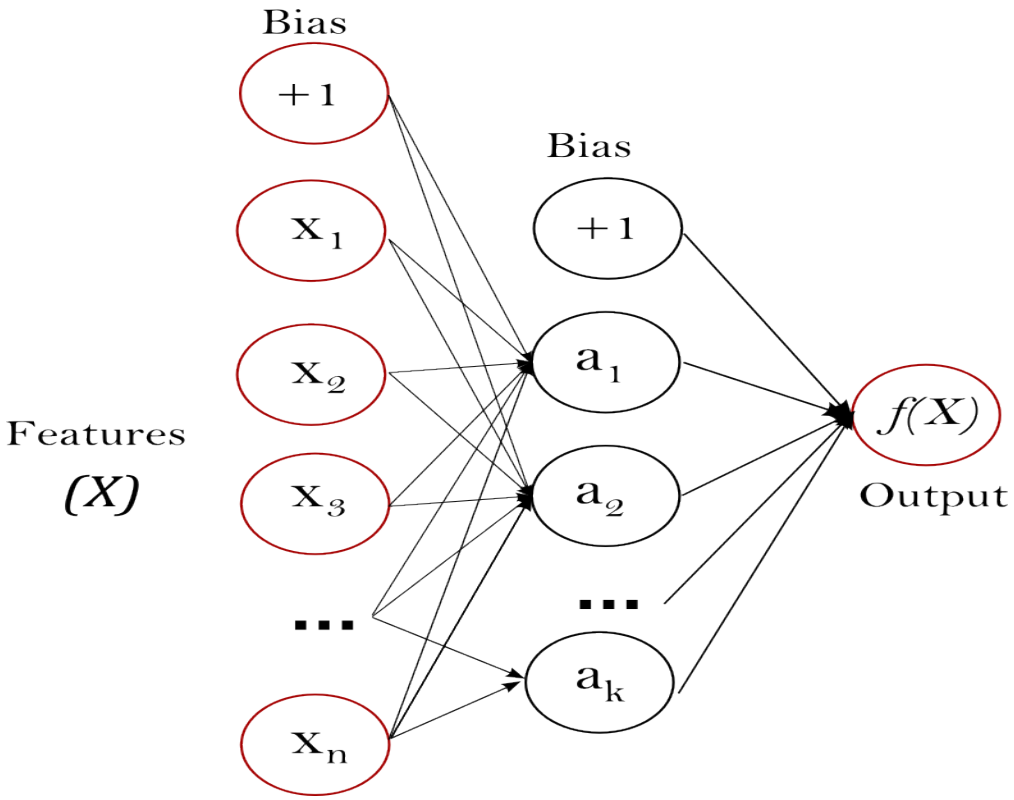


Figure 3.18: Multi Layer Perceptron [17]

Application to the Blood Brain Barrier prediction

Let $X = [x_1, x_2, x_3, ..., x_n]$ be the input vector to our network for n features and $W = [w_1, w_2, w_3, ..., w_n]$ the weights. Each neuron calculates whether it should fire or not based on its activation vector. We then examine the wrong neurons i.e the neurons that fired when they should have not. Figure 3.19 shows the activations of the neurons in the first layer of the neural network during the training phase with the darkest color representing no activation.

For each wrong neuron i, we calculate the difference in weights for it $\Delta w_i = -(y_i - t_i) * x_i$ where y_i is the output and t_i is the target output, the weight for neuron i is then updated as $w_i = w_i + \Delta w_i \eta$, where η is the learning rate (usually around $0.1 < \eta < 0.4$) which determines how much to change the weights by and as a result, determines how fast the network learns. The training is run again till the algorithm converges and the network gets all of its input right.

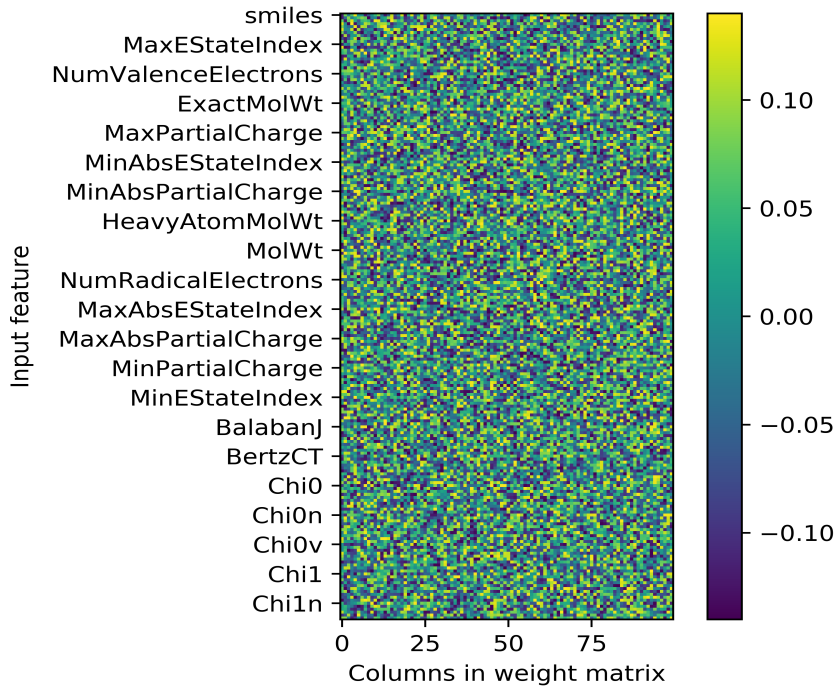


Figure 3.19: Heat map showing the weights learnt in the first layer of the neural network

The Learning Algorithm [12]

- **Initialisation**
  n small (positive and negative) random numbers are assigned as weights w_{ij} where i is the number of weights and j represents the neuron being examined.

- **Training** for T iterations or until all the outputs are correct

  - For each input vector $X = [x_1, x_2, x_3, ..., x_n]$
    * Compute the activation value (h) of each neuron j:

$$h = \sum_{i=0}^{n} w_{ij} x_i \tag{3.6}$$

* Calculate the activation (y) of neuron j using the activation function g:

$$y_j = g(h) = \begin{cases} 1 & \text{if h}>\theta \\ 0 & \text{if h} \leq \theta \end{cases} \tag{3.7}$$

* Update the weights of neuron j using the formula:

$$\Delta w_{ij} = -\eta(y_j - t_j) * x_i \tag{3.8}$$

$$w_{ij} = \Delta w_i j + w_i j \tag{3.9}$$

- **Recall or Prediction**
  To predict the value of new molecule with feature vector X using:

$$y_j = g(\sum_{i=0}^{m} w_{ij} x_i) = \begin{cases} 1 & \text{if } w_{ij} x_i > 0 \\ 0 & \text{if } w_{ij} x_i \leq 0 \end{cases} \tag{3.10}$$

  where 1 represents a molecule that passes through the barrier and 0 a molecule that doesn't.

Optimisations for Model Training Univariate Model Selection:
Here for each input data point in the feature vector, we compute whether there is a statistical relationship between the input and the target. The input features which are calculated to be highly unrelated to the target are then dropped from the dataset. Applying this model selection to the molecular descriptor dataset with a percentile of 50 gives us the features in figure 3.20. The neural network was

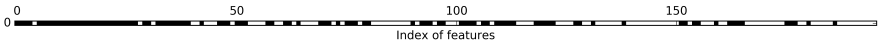


Figure 3.20: Matrix diagram of selected features

then trained with the new feature vector and a percentage increase in accuracy was achieved - with a new accuracy of $\sim 87\%$ on the molecular descriptor dataset and an accuracy of $\sim 88\%$ on the fingerprint dataset.

3.3.5 Ensemble Classifier

Ensembles were used earlier to combine many decision trees to achieve better classification results.

Ensembles have the benefit that they tend to perform better than single classifiers [6]; With this knowledge, all the previously trained classifier were combined to create an ensemble learner that makes predictions by taking a weighted vote of the individual prediction of the classifiers its composed of.

The ensemble classifier achieved an accuracy of $\sim$ 94% on both the simple molecular and the fingerprint dataset with its learning curve shown in figure 3.21; Which further enforces the claim by Thomas G. Dietterich [6] that ensemble learners perform better than individual learners and also that the ensemble classifier can achieve a much higher performance if trained with more data. Also, the reduction in the training score shows that the classifier over fits less with more data.

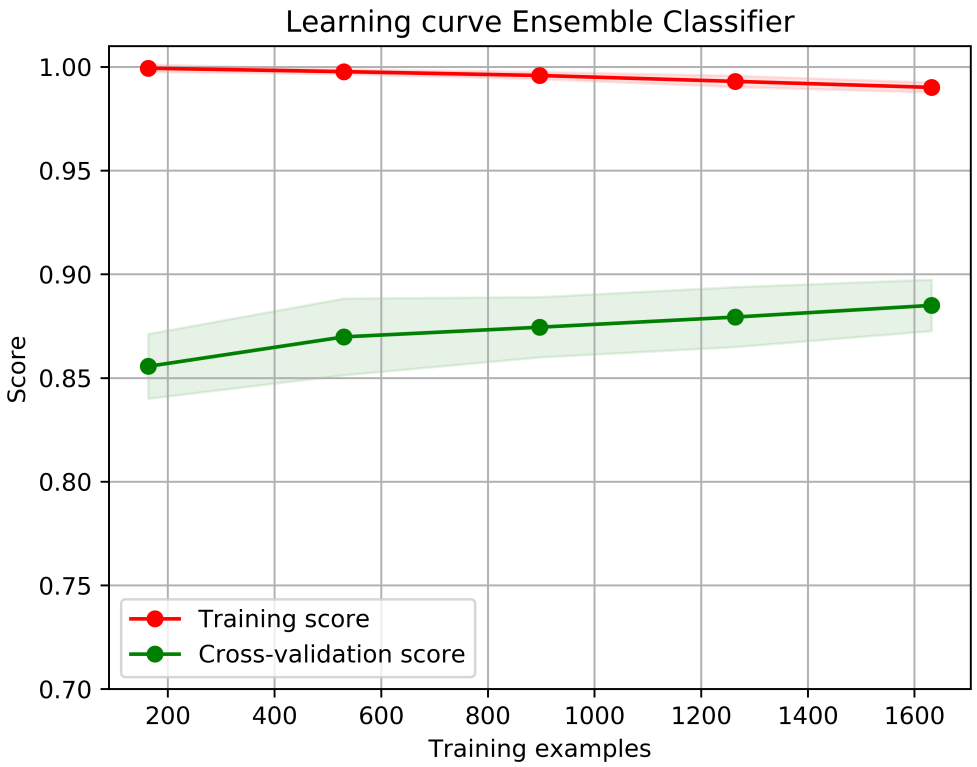


Figure 3.21: Learning Curve for the Ensemble classifier

Chapter 4

Model Evaluation and Persistence

4.1 Model Evaluation

The next step after training the classifier is to evaluate their performance. There is a myriad of techniques available for evaluating classifier performance and these techniques help us answer two basic questions about the model, which are:

- How accurate is the machine learning model?
- How wrong is the model when it is wrong?

4.1.1 Measuring Accuracy using K-Fold Cross-Validation

To avoid a machine learning model from over fitting (i.e failing to make meaningful predictions on new data), we employ a technique known as cross-validation. Here, a portion of the dataset is removed during the training phase and then used to evaluate the model during the test phase. Which further brings us to the concept of k-fold cross validation; In k-fold cross validation, the entire dataset is divided into k equal folds and the model is trained k times iteratively. In each iteration, the model is trained with the remaining $k-1$ folds and tested with the k^{th} fold.

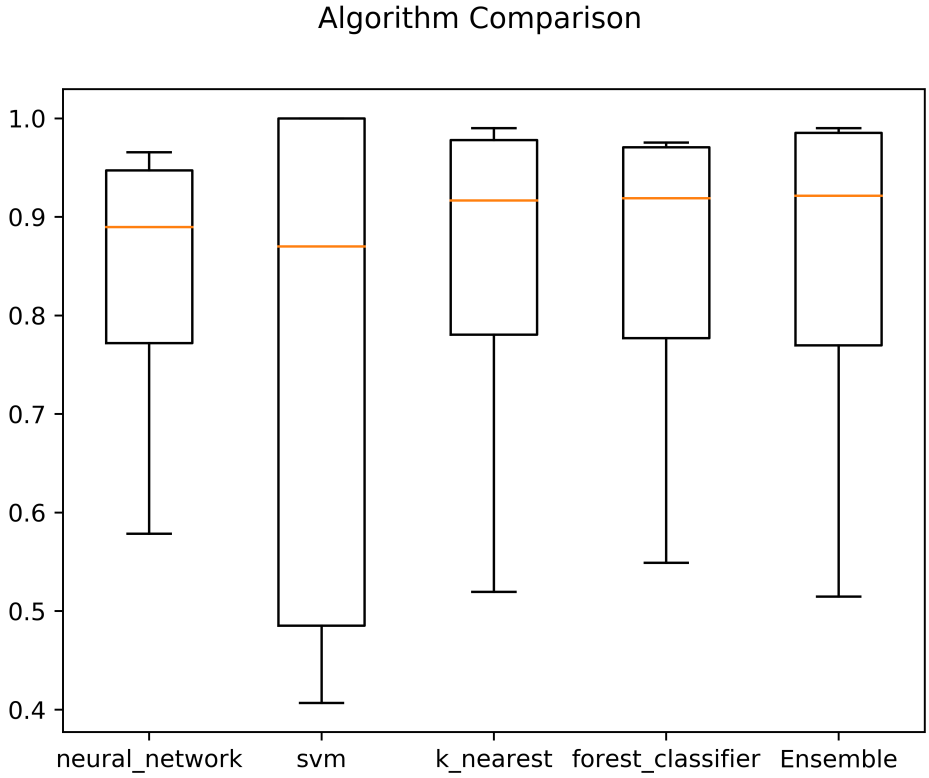


Figure 4.1: Box and whisker plot comparing the accuracy ranges of the classifiers

Applying the k-Fold Cross validation metric to the ensemble classifier on the simple molecular dataset with the number of folds set to 10 yields the accuracy results shown in figure 4.1.

4.1.2 Understanding models using Confusion matrices

A confusion matrix shows a detailed breakdown of the miscalculations of a machine learning model and it is sometimes known as an error matrix. It highlights in a table of two dimensions (*actual* and *predicted*) where the model made a correct and wrong prediction.

Applying the confusion matrix metric to the ensemble classifier, the result is shown in figure 4.2; The diagonals represent the number accurate predictions made for each class, while the non-diagonals represent the misclassification of the model. For example, in the confusion matrix, the ensemble classifier correctly predicted 454 molecules as being able to pass through the barrier and falsely classified the remaining 15 molecules as not being able to pass.

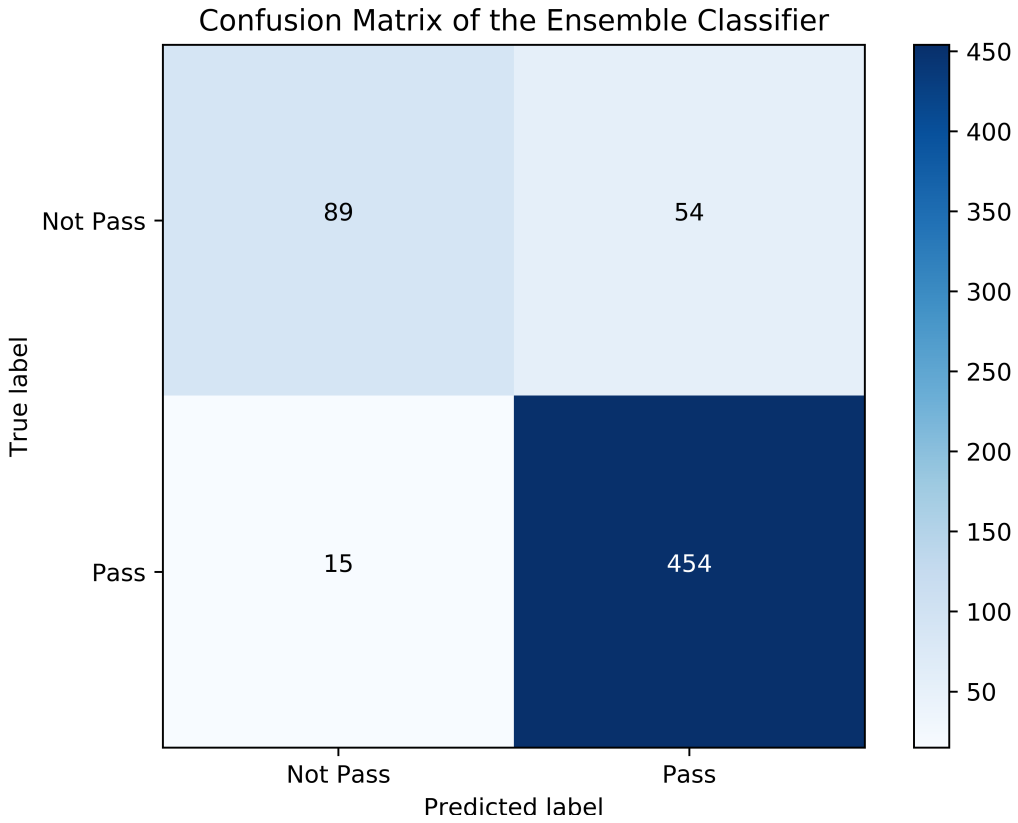


Figure 4.2: Confusion Matrix of the Ensemble classifier

Receiver Operating Characteristic Curve (ROC curve)

The ROC curve according to Tom Fawcett is a technique for visualizing, organizing and selecting classifiers based on their performance [9].

The ROC curve is illustrated by plotting the True Positive Rate (TPR; The fraction of true positives out of the positive classifications) against the False Positive Rate (FPR; The fraction of False positives out of the negative classifications) at various thresholds [9]. Shown in figure 4.3 is the ROC curve of the Ensemble classifier when evaluated with the simple molecular descriptor dataset. The diagonal represents the expected curve if the model were simply guessing randomly.

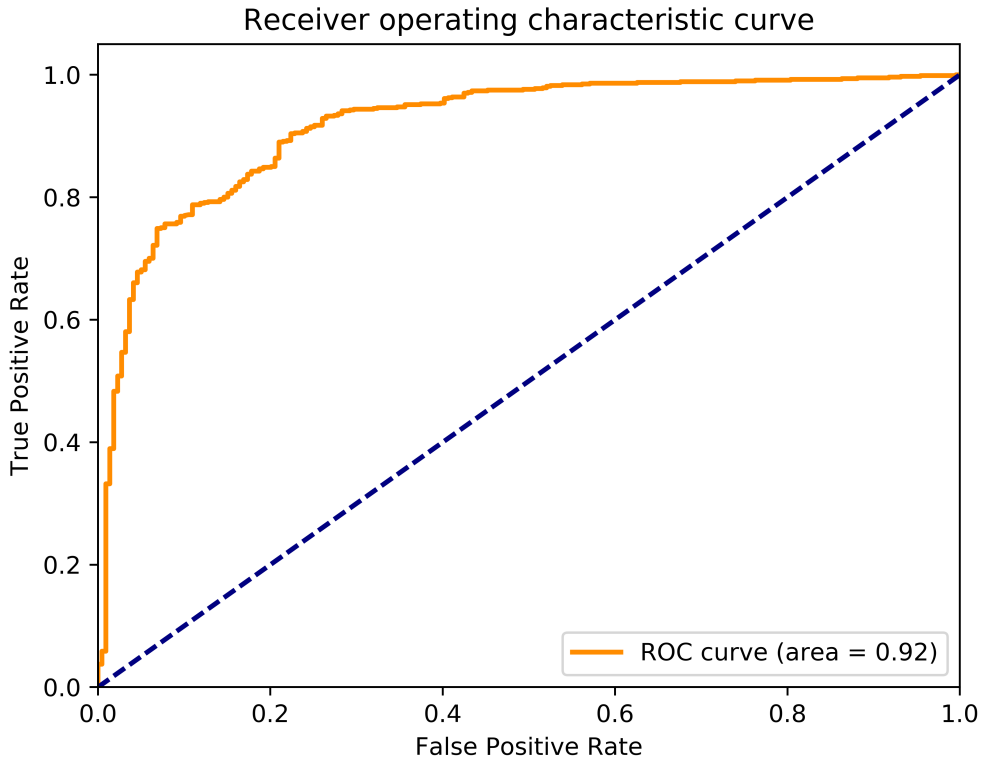


Figure 4.3: ROC for the Ensemble classifier

4.1.3 Classifier Performance Comparison

The table below simply shows the average performance of the individual machine learning models applied to the Blood-Brain Barrier datasets.

Table 4.1: Classifier average performance on different datasets

Classifier	Simple Molecular descriptors(%)	Morgan Fingerprint(%)
Decision Tree	80	79
Extra Trees	86	84
Random Forests	85	84
AdaBoost Decision Trees	84	82
Gradient Boosted Trees	86	82
k-Nearest Neighbours	87	89
Support Vector Machines	84.9	84.9
Neural Networks	87	88

4.2 Model Persistence

Using the *joblib* function made available in the *scikit-learn* library, it is possible to export a trained model to disk so it is available to use without having to retrain the model.

The model can be saved to disk using the code snippet below

```
from sklearn.externals import joblib

model_name = "brain_{:}_bbb.pkl".format(datetime.date.today())
print("Saved model as {:}".format(model_name))
joblib.dump(voting_clf, model_name)
```

And it can then be retrieved for use later using

```
voting_clf = joblib.load(model_name)
```

Chapter 5

Conclusion

In Chapter 1, we learnt that there is an abundance of chemical data and that machine learning methods can help discover patterns in these data especially in the field of Central Nervous System drug discovery.

Different data exploratory techniques were applied to our dataset and after the visualisations highlighted that it was possible to detect the statistical pattern in the dataset, we then proceeded to train numerous machine learning models with our data. Eventually, an Ensemble classifier combining all the previously trained models was created for use in any prediction task.

5.1 Integration into a Web Application

The Ensemble classifier was wrapped in a Web server written in python using the Flask framework. It exposes an API on the endpoint "/api/prediction" where a simple POST request with the smile data shown below

```
$ curl -H "Content-Type: application/json" -X POST -d
    '{"smile":"Cn1c2CCC(Cn3ccnc3C)C(=O)c2c4ccccc14"}'
    http://localhost:5000/api/prediction
```

should return a JSON object containing the prediction result as shown in figure x, where the classification result and prediction probabilities are returned to the user

```
{
  "category": "p",
  "probability": {
    "n": 0.07729955064888563,
    "p": 0.9227004493511144
  },
  "smile": "Cn1c2CCC(Cn3ccnc3C)C(=O)c2c4ccccc14"
}
```


Figure 5.1: Sample prediction result for the BBB Rest API

The entire application is then further wrapped in a docker container to enable deployment and training on any platform.

5.2 Areas for improvement

At the current stage of the project, a REST API has been made available for getting prediction results. This could further be extended into web solution with an intuitive front end where users i.e scientists could perform a bulk upload molecules from a virtual library for prediction results from our web server.
Other areas for improvements are:

5.2.1 Automatic lookup of smile from molecule name

Currently the prediction API expects the molecule SMILE in the POST data, an area that could be improved on would be the use of a molecular lookup database; With such an implementation, the consumer of the API can simply pass the molecule name and the web server can use the molecular lookup database to infer the SMILE representation of the molecule.

5.2.2 Deep Learning in virtual screening

Whilst considerable success has been made in applying the machine learning models aforementioned in Chapter 3 to our chemical dataset. According to a paper by Thomas et al, 2014 [20]. They show that Deep learning techniques have a significant opportunity in virtual screening

Bibliography

[1] Sarah Guido Andreas C. Müller. *Introduction to Machine Learning with Python: A Guide for Data scientists*. O'Reilly - O'Reilly Media, 1st edition, 2016.

[2] Jrgen Bajorath. *Chemoinformatics: Concepts, Methods and Tools for Drug Discovery*. Humana Press, Totowa, New Jersey, 2004.

[3] Dr A.N Boa. Introduction to drug discovery. http://www.hull.ac.uk/php/chsanb/DrugDisc/Introduction%20to%20Drug%20Discovery%202012.pdf. Accessed: 2016-11-28.

[4] F.K. Brown. Chemoinformatics: What is it and how does it impact drug discovery. *Annual Reports in Medicinal Chemistry*, 33:375–384, 1998.

[5] Chemical abstracts service home page CAS. Cas registry - the gold standard for chemical substance information, 2016, accessed November 7, 2016. http://www.cas.org/content/chemical-substances.

[6] Thomas G. Dietterich. Ensemble methods in machine learning. *Oregon State University, California, Oregon, USA*. http://web.engr.oregonstate.edu/~tgd/publications/mcs-ensembles.pdf.

[7] Petra Fey Takashi Gojobori Linda Hannick Winston Hide David P. Hill Renate Kania Mary Schaeffer Susan St Pierre Simon Twigger Owen White Doug Howe, Maria Costanzo and Seung Yon Rhee1. Big data: The future of biocuration. *PubMed Central (PMC)*, 455.

[8] Drugs.com. Caffeine/ergotamine: Indications, side effects, warnings - drugs.com. https://www.drugs.com/cdi/caffeine-ergotamine.html. Accessed: 2016-12-11.

[9] Tom Fawcett. An introduction to roc analysis. *Pattern Recognition Letters*, 27:861–874, 2005.

[10] Luis Pinheiro Ines Filipa Martins, Ana L. Teixeira and Andre O. Falcao. A bayesian approach to in silico blood-brain barrier penetration modeling. *Journal of Chemical Information and Modelling*, 52:16861697, 2012.

[11] M. A. Johnson and G. M. Maggiora. Concepts and applications of molecular similarity. 1990.

[12] Stephen Marshall. *Machine Learning: An Algorithmic Perspective*. Chapman and Hall/CRC, 2nd edition, 2014.

[13] Tom Mitchell. *Machine Learning*. McGraw-Hill International Editions, 1st edition edition, 1997.

[14] OpenBabel, 2017, accessed March 8, 2017. https://openbabel.org/wiki/SMARTS.

[15] Hassan Pajouhesh and George R. Lenz. Medicinal chemical properties of successful central nervous system drugs. *The American Society for Experimental NeuroTherapeutics*, 2:541–553, 2005.

[16] Louis Wehenkel Pierre Geurts, Damien Ernst. Extremely randomized trees. 2006. http://www.montefiore.ulg.ac.be/~ernst/uploads/news/id63/extremely-randomized-trees.pdf.

[17] Scikit-Learn, 2016, accessed November 7, 2016. http://scikit-learn.org/.

[18] D. Sculley. Web scale k-means clustering. *Proceedings of the 19th international conference on World Wide Web*, 2010. http://www.eecs.tufts.edu/~dsculley/papers/fastkmeans.pdf.

[19] Ana L. Teixeira. *Machine learning methods for quantitative structure-property relationship modelling*. PhD thesis, Universidade De Lisboa, Faculdade De Cincias, 2014.

[20] Gnter Klambauer Marvin Steijaert Jrg Wegner Hugo Ceulemans Sepp Hochreiter Thomas Unterthiner, Andreas Mayr. Deep learning as an opportunity in virtual screening. *Conference, Neural Information Processing Systems Foundation*, 2014.

[21] Roberto Todeschini and Viviana Consonni. Molecular descriptors for chemoinformatics. *Wiley-VCH*, 2, 2009. Accessed: 2017-03-26.

[22] Dan Kushnir Yair Goldberg, Alon Zakai and Yaacov Ritov. Manifold learning: The price of normalization. *Journal of Machine Learning Research*, 2008. http://www.jmlr.org/papers/volume9/goldberg08a/goldberg08a.pdf.

PROJECT LOG

```
commit c1e8645105068c5abcf0ebac7cb0a1981ddf0b94
Author: Lxrd_AJ <ibraheemaj@icloud.com>
Date:   Thu May 11 07:32:31 2017 +0100

    Uncommented code in viz code

commit c58a1ef05de14344c9294aa61fbb5a371e86e0ff
Author: Lxrd_AJ <ibraheemaj@icloud.com>
Date:   Thu May 11 07:24:34 2017 +0100

    Dockerfile updated

commit ca0682094dba72b7ae05b0832d28fa89bde951ad
Author: AJ Ibraheem <ibraheemaj@icloud.com>
Date:   Wed May 10 21:31:56 2017 +0100

    Ethics form added

commit b5e972c8c450fbd71e178c0665d600e57195cc68
Author: AJ Ibraheem <ibraheemaj@icloud.com>
Date:   Wed May 10 16:00:46 2017 +0100

    Examiner's report complete

commit 456052a60b0ab863b1bd0e213b366488a5b441af
Author: AJ Ibraheem <ibraheemaj@icloud.com>
Date:   Tue May 9 14:14:03 2017 +0100

    Edits to the report

commit ffb15f6f7279e403787829e650485d63d851f9b5
Author: Lxrd_AJ <ibraheemaj@icloud.com>
Date:   Tue May 9 02:06:07 2017 +0100

    docker integration added

commit 8d1bbc9293bb969a6e178c4a558237cfaf238ec5
Author: Lxrd_AJ <ibraheemaj@icloud.com>
Date:   Mon May 8 19:10:52 2017 +0100

    Model evaluation and persistence chapter updated

commit 285fa26b994693f76c373c09f3fb369480ee755f
Author: Lxrd_AJ <ibraheemaj@icloud.com>
Date:   Mon May 8 14:02:13 2017 +0100

    Model evaluation chapter updated

commit 6086a611892e4fed7b644e68e7c3e0c3203d4ac6
Author: AJ Ibraheem <ibraheemaj@icloud.com>
Date:   Mon May 8 08:32:16 2017 +0100

    minor changes
```

```
commit 5a2951843769ad5c4ed3cdb5bf05730b2eb8d081
Author: AJ Ibraheem <ibraheemaj@icloud.com>
Date:   Sun May 7 20:34:38 2017 +0100

    Machine learning classifier chapter updated

commit 75d253d24b2450373757571c4a0fdadbed336863
Author: Lxrd_AJ <ibraheemaj@icloud.com>
Date:   Sat May 6 04:29:25 2017 +0100

    Decision trees chapter complete and neural network chapter
updated

commit e03e7b76e3504891423457ca73095ef3bd84a4fe
Author: Lxrd_AJ <ibraheemaj@icloud.com>
Date:   Wed May 3 14:57:59 2017 +0100

    Support vector machine chapter updated

commit f1ebaed34f23189b2d14227d478d383ebe197098
Author: Lxrd_AJ <ibraheemaj@icloud.com>
Date:   Tue May 2 19:21:32 2017 +0100

    k-NN Chapter updated

commit cdd846b03d4aa50ff38ec31405a2e7444c7c871b
Author: Lxrd_AJ <ibraheemaj@icloud.com>
Date:   Sun Apr 30 15:26:22 2017 +0100

    Clustering section updated

commit d8e1a4d6ff3b74188fec73aa9be2c2bf882ba245
Author: Lxrd_AJ <ibraheemaj@icloud.com>
Date:   Wed Apr 26 14:30:13 2017 +0100

    Manifold learning with t-SNE section updated

commit 26659a3f336eb8935e9af792ad660bf43a739b90
Author: Lxrd_AJ <ibraheemaj@icloud.com>
Date:   Tue Apr 25 03:23:58 2017 +0100

    Principal component analysis section updated and folder
restructuring

commit 4c5ba296629bfcdf391b69aa945b8e5a13b601a5
Author: Lxrd_AJ <ibraheemaj@icloud.com>
Date:   Wed Apr 5 21:55:11 2017 +0100

    Data Representation and Preprocessing section added

commit 27aa08e4e8c1a4152d0807ed7c012cfc947b619e
Author: Lxrd_AJ <ibraheemaj@icloud.com>
Date:   Wed Apr 5 19:21:31 2017 +0100
```

```
    Machine learning chapter restructured and introductory text
added

commit 0c1762ce8db382fea1e5909e6a40116eb0eb431f
Author: Lxrd_AJ <ibraheemaj@icloud.com>
Date:   Wed Apr 5 12:55:29 2017 +0100

    Morgan Fingerprint chapter written

commit 08ad9afd3ccd20959cb906a25b29faf0508af0e5
Author: Lxrd_AJ <ibraheemaj@icloud.com>
Date:   Thu Mar 30 09:04:04 2017 +0100

    Notes added

commit 643b53815001178df46b32141547b96438f63e73
Merge: c528c11 0060ddd
Author: Lxrd_AJ <ibraheemaj@icloud.com>
Date:   Sun Mar 26 12:51:42 2017 +0100

    Merge

commit c528c114bc9f6bd0083a289cf8e5ca2832a57b3d
Author: Lxrd_AJ <ibraheemaj@icloud.com>
Date:   Sun Mar 26 12:43:31 2017 +0100

    Updated the molecular representation and similarity subsection

commit 0060ddd64e736cc35b15dccb7915748f21b0b5d2
Author: AJ Ibraheem <ibraheemaj@icloud.com>
Date:   Thu Mar 23 01:01:04 2017 +0000

    Notes added - Molecular Representation

commit baa6bb4ec256eeb2b2b7620d7980d776dc7c36c3
Merge: a508717 60ceb42
Author: AJ Ibraheem <ibraheemaj@icloud.com>
Date:   Thu Mar 23 00:06:59 2017 +0000

    Merge branch 'master' of https://github.com/Lxrd-AJ/
ml_blood_brain_barrier_prediction

commit a508717b6c470c6c3059624134c14579a3ebbc8c
Author: AJ Ibraheem <ibraheemaj@icloud.com>
Date:   Thu Mar 23 00:06:54 2017 +0000

    Brain model added

commit 60ceb4231e445d6a59d3f3ca3e13e89409ff79eb
Author: Lxrd_AJ <ibraheemaj@icloud.com>
Date:   Wed Mar 22 18:22:43 2017 +0000

    Background-Cheminformatics-Representing Molecules-SMILES chapter
update
```

```
commit c15906e92b445eb37842f60a1e1de0a05f64c4a4
Merge: a6ee7f1 3d2b0e3
Author: AJ Ibraheem <ibraheemaj@icloud.com>
Date:   Wed Mar 15 23:48:27 2017 +0000

    Fix merge

commit a6ee7f1ea1cbf453b122950ece5de2bddade6431
Author: AJ Ibraheem <ibraheemaj@icloud.com>
Date:   Wed Mar 15 23:47:11 2017 +0000

    Prepping for merge

commit 3d2b0e345feaaf9234517147f78d9db24daa152d
Author: Ibraheem AJ <ibraheemaj@icloud.com>
Date:   Wed Mar 15 20:24:14 2017 +0000

    Application wrapped in a web server using the Flask framework
and option to start the application using a pre-trained scikit-learn
model added

commit 4e86ff9b6373cf8bb96cc0fa87e1de5b6e8d58f7
Author: Ibraheem AJ <ibraheemaj@icloud.com>
Date:   Sun Mar 12 20:51:34 2017 +0000

    TODO notes

commit 7badbe42c498f8c1578792a8bc24a7c4ee6c7f50
Author: Ibraheem AJ <ibraheemaj@icloud.com>
Date:   Sun Mar 12 15:09:50 2017 +0000

    Bug fix to the decision tree pipeline

commit ebf851047c23de8f689ef292dc57b0dfc72fe521
Author: Ibraheem AJ <ibraheemaj@icloud.com>
Date:   Sun Mar 12 01:46:49 2017 +0000

    ensemble voting classifier implemented

commit 0c60864304d49ebdb5d6a6a07d0b5598e6880f4c
Author: Ibraheem AJ <ibraheemaj@icloud.com>
Date:   Sat Mar 11 22:17:55 2017 +0000

    Staging for commit

commit df6bf5fad379462965de7ebac7d37c0d783de0b3
Author: Ibraheem AJ <ibraheemaj@icloud.com>
Date:   Wed Mar 1 15:30:48 2017 +0000

    Model pipelines + Voting classifier

commit 80e23f82299f9df5f32f387221762f790c3ffb82
Author: Ibraheem AJ <ibraheemaj@icloud.com>
```

```
Date:   Tue Feb 28 11:04:29 2017 +0000

    Custom estimator in progress

commit 0123738dc70b98840360d616dcec72414a444c5f
Author: Ibraheem AJ <ibraheemaj@icloud.com>
Date:   Tue Feb 14 18:43:55 2017 +0000

    Reinitialised git repo
```

SCHOOL OF COMPUTING, ENGINEERING & MATHEMATICS ETHICS FORM

PROJECT DETAILS

1. Name of researcher: Ganiyu Ajibola Ibraheem

2. Name of supervisor: Aidan Delaney

3. Title of project:

Machine Learning in Drug Discovery and Design: Predicting the Blood Brain Barrier Penetration of Drugs

4. Outline of the research (up to 100 words):

Drug design and discovery is a very expensive process, with numerous new compounds being developed at a rapid rate. Only around 2\% of known drugs can pass through the BBB, this presents a problem in the Central Nervous System (CNS) drug development.
This project aims to develop a solution that can predict, with confidence, the probability of a drug passing through the BBB in the hope that this can speed up the process of developing a CNS drug.

5. Location of research: University of Brighton, Mouslecoomb Campus

8. Email address: gai10@uni.brighton.ac.uk

9. Contact address:

10. Telephone number: 07447089875

Please tick the appropriate box and answer the questions where appropriate.	Yes	No
1. Does the study involve **participants who might be considered vulnerable** due to age or to a social, psychological or medical condition? (*e.g. children, people with learning disabilities or mental health problems, but participants who may be considered vulnerable are not confined to these groups).* If yes then provide details of any such participants. See the University's 'Guidance on Good Practice in Research Ethics and Governance' for more details. …… …… Note: proposals involving vulnerable participants are often likely to require ethical approval from the Faculty of Science & Engineering Research Ethics and Governance Committee (FREGC).		☑
2. Will **photographic or video recordings** of research participants be collected as part of the research? If yes then please outline consent and data protection procedures *(e.g. interviews cannot be overheard, details will not be accessible to others),* for the use of participants' images. Example consent and information forms can be found on StudentCentral and see guidance on data collection at the end of this document. …… …… If your data will not be confidential and anonymous then outline the justification for this decision here and procedures for mitigating against potential harm. …… ……		☑
3. Does the study require the **co-operation of an individual to gain access** to the participants? (*e.g. a teacher at a school or a manager of sheltered housing)* If yes then describe the procedures that will be put in place to ensure safe and ethical direct involvement of human participants. Where necessary and as appropriate, include comments on obtaining informed consent, reducing harm, providing feedback, and accessing participants through an individual providing information such as a teacher/lecturer, manager, employer etc. Example consent and information forms can be found on StudentCentral. …… ……		☑
4. Will the participants be asked to discuss what might be perceived as **sensitive topics** (*e.g. sexual behaviour, drug use, religious belief, detailed financial matters) or* could participants experience psychological stress, anxiety or other negative consequences (beyond what would be expected to be encountered in normal life)? If yes then describe the procedures that will be put in place to ensure safe and ethical direct involvement of human participants. Where necessary and as appropriate, include comments on obtaining informed consent, reducing harm, providing feedback. Example consent and information forms can be found on StudentCentral. …… ……		☑

Please tick the appropriate box and answer the questions where appropriate.	Yes	No
5. Will individual participants be involved in **repetitive/prolonged testing or vigorous physical activity, experience pain of any kind, or be exposed to dangerous situations, environments or materials** as part of the research? If yes then describe the procedures that will be put in place to ensure safe and ethical direct involvement of human participants. Where necessary and as appropriate, include comments on obtaining informed consent, reducing harm, providing feedback. Example consent and information forms can be found on StudentCentral. …… ……		☑
6. Will members of the public be **indirectly involved** in the research without their knowledge at the time? (*e.g. covert observation of people in non-public places, the use of methods that will affect privacy)*. If yes then provide brief details here *(e.g. how they will be involved and, where known, the age, gender, ethnicity and location of those who will be indirectly involved).* …… …… ` Provide details of any negative impacts members of the public will be likely to face and that would not be considered minimal impacts (e.g. invasion of privacy, harm to property, being subject to what an individual perceives to be inappropriate behaviour). Describe the risks and if appropriate explain why you believe they are only minimal. …… …… Describe any procedures that will be put in place to ensure safe and ethical indirect involvement of members of the public (*e.g. providing information and feedback if requested by the public*). Examples of participation information forms can be found on StudentCentral. …… …… Describe how you will ensure data collection is confidential and anonymous (*e.g. people will not be able to be identified by photographs or notes taken by observers*), how data will be stored and who will have access to the data. If the data will not be confidential or anonymous, outline the justification for this decision here and procedures for mitigating against potential harm. …… ……		☑
7. Does this research include **secondary data** that may carry personal or sensitive organisational information? *(Secondary data refers to any data you plan to use that you did not collect yourself, e.g. datasets held by organisations, patient records, confidential minutes of meetings, personal diary entrie).* If yes then provide details regarding any secondary data to be used that may carry sensitive personal or organisational information. …… …… If secondary data CEMs containing sensitive personal or organisational information are to be used, outline how such use will be ethically managed *(e.g. details such as anonymising data CEMs, ensuring protection of source agency, gaining consent of data owners, and how the data will be stored)*. See guidance on data collection at the end of this document. …… ……		☑

Please tick the appropriate box and answer the questions where appropriate.	Yes	No
8. Is this research likely to have significant **negative impacts on the environment**? (*For example, the release of dangerous substances or damaging intrusions into protected habitats.)* If yes then provide details of these impacts here (for example the release of dangerous substances or damaging intrusions into protected habitats) and …… …… Describe how you will mitigate against significant environmental harm and manage risks. …… ……		☑
9. Will any participants receive **financial reimbursement** for their time? (*excluding reasonable expenses to cover travel and other costs*). If yes then provide details and a short justification (e.g. amounts and form of reimbursement). …… ……		☑
10. Are there any **other ethical concerns** associated with the research that are not covered in the questions above? If yes then give details here. …… ……		☑

All Undergraduate and Masters level projects or dissertations in the School of CEM must adhere to the following procedures on data storage and confidentiality.

All data should be encrypted and stored securely. Documentation should be kept in a locked cabinet or desk, and electronic data should preferably be kept on a removable disk or data stick which can be locked away, or if this is not possible on a password protected computer. Confidential and sensitive data should not be emailed unless it is encrypted or password protected since emails are centrally archived.

For Undergraduate/Masters projects, normally only the student and supervisor will have access to the data (see the University's 'Guidance on Good Practice in Research Ethics and Governance for further details).Once a mark for the project or dissertation has been published, all data must be removed from personal computers, and original questionnaires and consent forms should be destroyed unless the research is likely to be published or data re-used. If this is the case a justification for this should be included where appropriate in this form and in the relevant consent and participant information forms.

Student: Please sign below to confirm that you have completed the Ethics form and will adhere to these procedures on data storage and confidentiality.

Signed (**Student**): ……
Date: 7th May, 2017

Supervisor: I confirm that the research ***does/does not*** (delete as applicable) include more than a **minimum level of risk**.

Signed (**Supervisor**): ……
Date: ……

Note: If the **supervisor judges** that there is more than the **minimum level of risk** then your supervisor will need to email this form to the CEM ethics committee (CEMethics@brighton.ac.uk) for discussion prior to the commencement of research.