UNIVERSITY OF THESSALY

SCHOOL OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

COMPUTATIONAL METHODS FOR OPTIMIZING THERAPIES FOR DUCHENNE MUSCULAR DYSTROPHY

Diploma Thesis

ELENI KOUTSONI

Supervisor: Panagiota Tsompanopoulou

February 2022

UNIVERSITY OF THESSALY

SCHOOL OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

COMPUTATIONAL METHODS FOR OPTIMIZING THERAPIES FOR DUCHENNE MUSCULAR DYSTROPHY

Diploma Thesis

ELENI KOUTSONI

Supervisor: Panagiota Tsompanopoulou

February 2022

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΥΠΟΛΟΓΙΣΤΙΚΕΣ ΜΕΘΟΔΟΙ ΓΙΑ ΒΕΛΤΙΣΤΟΠΟΙΗΣΗ ΘΕΡΑΠΕΙΩΝ ΓΙΑ ΤΗΝ ΜΥΪΚΗ ΔΥΣΤΡΟΦΙΑ DUCHENNE

Διπλωματική Εργασία

ΕΛΕΝΗ ΚΟΥΤΣΩΝΗ

Επιβλέπουσα: Παναγιώτα Τσομπανοπούλου

Φεβρουάριος 2022

Approved by the Examination Committee:

Supervisor	**Panagiota Tsompanopoulou** Associate Professor, Department of Electrical and Computer Engineering, University of Thessaly
Member	**Georgios Paliouras** Senior Researcher, Institute of Informatics and Telecommunications at NCSR “Demokritos”
Member	**Aspassia Daskalopulu** Assistant Professor, Department of Electrical and Computer Engineering, University of Thessaly

Acknowledgements

First of all, I would like to thank the Head of the SKEL The AI Lab (NCSR Demokritos) Dr. George Paliouras, and his team, Dr. Anastasia Krithara, Anastasios Nentidis and Vasileios Konstantakos for the opportunity they gave me to work on this project and for the guidance and feedback they provided me throughout this thesis.

Additionally, I would like to thank Prof. Panagiota Tsompanopoulou and Prof. Aspassia Daskalopulu, for supervising my thesis.

Last but not least, my parents and sisters deserve endless gratitude for providing me unconditional love and support throughout all of these years.

DISCLAIMER ON ACADEMIC ETHICS AND INTELLECTUAL PROPERTY RIGHTS

«Being fully aware of the implications of copyright laws, I expressly state that this diploma thesis, as well as the electronic files and source codes developed or modified in the course of this thesis, are solely the product of my personal work and do not infringe any rights of intellectual property, personality and personal data of third parties, do not contain work / contributions of third parties for which the permission of the authors / beneficiaries is required and are not a product of partial or complete plagiarism, while the sources used are limited to the bibliographic references only and meet the rules of scientific citing. The points where I have used ideas, text, files and / or sources of other authors are clearly mentioned in the text with the appropriate citation and the relevant complete reference is included in the bibliographic references section. I also declare that the results of the work have not been used to obtain another degree. I fully, individually and personally undertake all legal and administrative consequences that may arise in the event that it is proven, in the course of time, that this thesis or part of it does not belong to me because it is a product of plagiarism».

The declarant

ELENI KOUTSONI

Diploma Thesis

COMPUTATIONAL METHODS FOR OPTIMIZING THERAPIES FOR DUCHENNE MUSCULAR DYSTROPHY

ELENI KOUTSONI

Abstract

Duchenne Muscular Dystrophy (DMD) is a neuromuscular disorder caused by the absence of the dystrophin protein. If left untreated, it causes movement problems at the age of 10-12 years, and death occurs in the 20-30 years due to heart failure. There is currently no cure for this disease, only symptomatic treatment.

Genome editing approaches like the CRISPR-Cas9 technology can provide new opportunities to ameliorate the disease by eliminating DMD mutations and restoring dystrophin expression. Because of its capability to modify specific genes and genomic regions complementary to an engineered single guide RNA (sgRNA), the CRISPR-Cas9 system has sparked much interest as a genome editing approach in recent years. While it is true that on-target activity can be influenced by the guide specificity, we would focus here on the devastating results that off-target cleavage can cause (e.g., unexpected mutations). This is why reducing off-target effects is the first priority in guide design.

The rapid growth of the Artificial Intelligence field has helped researchers employ artificial feature extraction and Machine Learning approaches to evaluate the potential off-target scores.

This thesis presents our approach in evaluating off-targets of CRISPR-Cas9 gene editing specifically for the DMD disorder, using Machine Learning. We offer a comparison between four regression methods that predict the insertions-deletions (indels) produced based on a pair guide RNA and the equivalent off-target. We evaluate the results using the Spearman correlation metric.

Finally, we proposed the most suitable method (Decision Tree Regressor) for this problem and compared the results with some state-of-art tools. The performance of our tool with CV is better than the independent performance of the other tools except from Elevation which performed about as good as ours.

Keywords:

Machine Learning, DMD, CRIPSR-Cas9, off-targets

Διπλωματική Εργασία

ΥΠΟΛΟΓΙΣΤΙΚΕΣ ΜΕΘΟΔΟΙ ΓΙΑ ΒΕΛΤΙΣΤΟΠΟΙΗΣΗ ΘΕΡΑΠΕΙΩΝ ΓΙΑ ΤΗΝ ΜΥΪΚΗ ΔΥΣΤΡΟΦΙΑ DUCHENNE

ΕΛΕΝΗ ΚΟΥΤΣΩΝΗ

Περίληψη

Η μυϊκή δυστροφία Duchenne (DMD) είναι μια νευρομυϊκή διαταραχή που προκαλείται από την απουσία της πρωτεΐνης δυστροφίνης. Εάν δεν εφαρμοστεί θεραπεία, προκαλεί κινητικά προβλήματα στην ηλικία των 10-12 ετών και ο θάνατος επέρχεται στην ηλικία των 20-30 χρόνων λόγω καρδιακής ανεπάρκειας. Επί του παρόντος, δεν υπάρχει διαθέσιμη θεραπεία, αλλά μόνο συμπτωματική αντιμετώπιση.

Οι προσεγγίσεις επεξεργασίας γονιδιώματος, όπως η τεχνολογία CRISPR-Cas9, μπορούν να προσφέρουν νέες ευκαιρίες για τη βελτίωση της νόσου εξαλείφοντας τις μεταλλάξεις DMD και αποκαθιστώντας την έκφραση της δυστροφίνης. Λόγω της ικανότητάς του να τροποποιεί συγκεκριμένα γονίδια και γονιδιωματικές περιοχές συμπληρωματικά σε ένα κατασκευασμένο single guide RNA (sgRNA), το σύστημα CRISPR-Cas9 έχει προκαλέσει μεγάλο ενδιαφέρον ως προσέγγιση επεξεργασίας γονιδιώματος τα τελευταία χρόνια. Αν και είναι αλήθεια ότι η δραστηριότητα επί του στόχου μπορεί να επηρεαστεί από την ειδικότητα του gRNA, θα εστιάσουμε εδώ στα καταστροφικά αποτελέσματα που μπορεί να προκαλέσει η διάσπαση εκτός στόχου (off-target) (π.χ. απροσδόκητες μεταλλάξεις). Για αυτό το λόγο η μείωση των αποτελεσμάτων εκτός στόχου είναι η πρώτη προτεραιότητα στο σχεδιασμό οδηγών.

Η ταχεία ανάπτυξη του τομέα της τεχνητής νοημοσύνης βοήθησε τους ερευνητές να χρησιμοποιήσουν προσεγγίσεις τεχνητής εξαγωγής χαρακτηριστικών και μηχανικής μάθησης για την αξιολόγηση των πιθανών αποτελεσμάτων εκτός στόχου.

Η παρούσα διατριβή παρουσιάζει την προσέγγισή μας στην αξιολόγηση των ακολουθιών εκτός στόχου της γονιδιακής επεξεργασίας CRISPR-Cas9 ειδικά για τη διαταραχή DMD, χρησιμοποιώντας μηχανική μάθηση. Προσφέρουμε μια σύγκριση μεταξύ τεσσάρων μεθόδων παλινδρόμησης που προβλέπουν τις εισαγωγές-διαγραφές (indels) που παράγονται με βάση ένα ζεύγος οδηγού RNA και τον αντίστοιχο εκτός στόχου. Αξιολογούμε τα αποτελέσματα χρησιμοποιώντας τη μετρική συσχέτισης Spearman.

Τέλος, προτείναμε την καταλληλότερη μέθοδο (Decision Tree Regressor) για αυτό το πρόβλημα και συγκρίναμε τα αποτελέσματα με ορισμένα εργαλεία τελευταίας τεχνολογίας. Η απόδοση του εργαλείου μας με cross-validation είναι καλύτερη από την ανεξάρτητη απόδοση των άλλων εργαλείων εκτός από το Elevation το οποίο είχε περίπου την ίδια καλή απόδοση με το δικό μας.

Λέξεις-κλειδιά:

Μηχανική Μάθηση, Μυϊκή Δυστροφία Duchenne, CRISPR-Cas9, ακολουθίες εκτός στόχου

Table of contents

List of figures

List of tables

Abbreviations

DMD	Duchenne Muscular Dystrophy
CRISPR	Clustered Regularly Interspaced Short Palindromic Repeats
PAM	Protospacer-associated motif
sgRNA	Single guide RNA
indel	Insertion or deletion
DMD_DTR	The Decision Tree Regressor developed for this thesis

Chapter 1

Introduction

Duchenne Muscular Dystrophy (DMD) is a neuromuscular disorder caused by the absence of the dystrophin protein. It is an X-linked disease, affecting one in 3,500 [9] males, making it the most common muscular dystrophy. DMD is caused by mutations in the DMD gene that encodes the dystrophin protein, which acts as a link between the cytoskeleton and the extracellular substance. If left untreated, it causes movement problems at the age of 10-12 years, and death occurs in the 20-30 years due to heart failure.

Genome editing approaches like the CRISPR-Cas9 technology can help patients present a less severe disorder or hopefully get entirely cured. The CRISPR-Cas9 system has been widely used in target gene repair and gene expression regulation as a genome-editing technique. The CRISPR-Cas9 system can increase on-target knockout effectiveness with great sensitivity and specificity by selecting optimal sgRNA (sgRNAs can be synthesized or produced from a DNA template). However, off-target cleavage can occur when the CRISPR-Cas9 system is used. Several prediction approaches for sgRNA's off-target tendency at specific DNA segments have been established so far. To get the off-target scores, most of them employ artificial feature extraction and Machine Learning approaches. However, all the existing tools that have been developed are generic and could not perform ideally in specific diseases.

1.1 Subject of the thesis

In this thesis, we present our approach in evaluating off-targets specifically for the DMD disorder. The goal was to create a model that can serve as an advisory tool so that the application of CRISPR-Cas9 gene therapy is sufficiently good. Our method was derived from the comparison of four Machine Learning Regression algorithms (Decision Tree Regressor, XGBoost Regressor, Random Forest Regressor, and Support Vector Regressor) that predict the indels produced based on a pair guide RNA and the equivalent off-target. Moreover, we evaluate the results using the Spearman correlation metric.

1.2 Organization of the thesis

The thesis is organized in the following manner. In Chapter 2, a background about DMD, the CRISPR-Cas9 system, the off-target effects, and the Machine Learning algorithms we studied is presented. Then, in Chapter 3, the basic methodology of the implementation is introduced. Chapter 4 describes the results and the comparison we conducted with some state-of-the-art tools. Finally, in Chapter 5, we provide a conclusion of the thesis and some thoughts about future work.

Chapter 2

Background

2.1 Duchenne Muscular Dystrophy

Dystrophinopathies are X-linked muscle diseases that can be characterized as either more severe or milder. They are the result of mutations in the DMD gene, which encodes the dystrophin protein, which is needed to stabilize the plasma membrane of striated muscle cells. In the category of mild dystrophinopathies we observe the phenotypes of asymptomatic increase in serum creatine phosphokinase (CK) concentration and muscle cramps with myoglobinuria, while the most severe include Duchenne Muscular dystrophy (DMD), Becker Muscular dystrophy (BMD) and DMD-associated dilated cardiomyopathy (DCM) [10].

Duchenne muscular dystrophy is the most serious and common dystrophinopathy which, if left untreated, leads to the death of the patient. Due to the way it is inherited, it mainly affects boys (1: 3500) while girls can be carriers of the disease with rare cases of girls who get sick (<1 per million). From a very young age the first symptoms appear such as gait abnormalities, difficulty getting up from the ground and frequent falls of the patient. At the age of 10 to 12 years, most patients with DMD are confined to a wheelchair. Dilated cardiomyopathy and arrhythmias as well as chronic respiratory failure are also observed. If left untreated the maximum life expectancy is the 20's due to cardiorespiratory complications that lead to death [11].

DMD is caused by mutations in the DMD gene (Figure 2.1) which produces the dystrophin protein, located in skeletal muscle and heart muscle and is necessary as it acts as a link between the cytoskeleton and the extracellular substance. The DMD gene is the largest in the human body and mutations in this gene cause some dysfunctional or no dystrophin

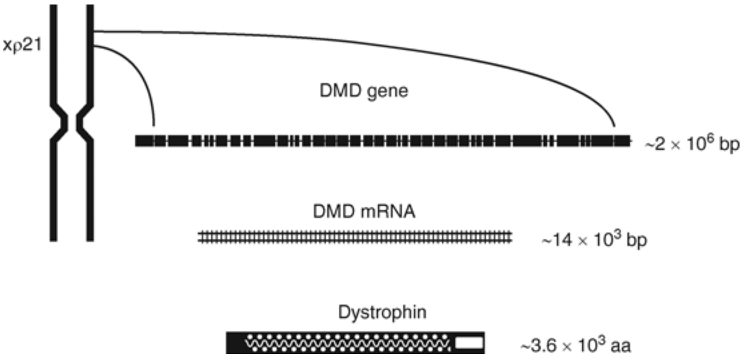

Figure 2.1: The location of the dystrophin gene on the Xp21 chromosome, the gene, the translated mRNA and the protein produced. Adapted from Sinnreich et al. [1]

production at all. It extends 2.4 mega bp in the short arm of the X chromosome, includes 79 exons and takes 16 hours to complete the transcription including transcription splicing (con-trascriptional splicing). Dystrophin corresponds to 1% of the X chromosome and 0.08% of the entire genome.

2.2 CRISPR-Cas9

2.2.1 Prokaryotes

After the discovery of the unique way in which bacteria and archaea protect themselves against cellular invaders, scientists realized that this discovery enables them to cut genomic DNA at precise points in eukaryotic cells. This defense mechanism is represented by **Clustered Regulatory Interspaced Short Palindromic Repeats** or **CRISPR**, along with the CRISPR-associated Proteins or Cas proteins.

Bacteria and archaea [2] develop cellular memory of previous invaders by incorporating DNA sequences in their genome that are identical to previous invaders (Figure 2.2). The first phase is the integration, that takes place at a variable site in the genome of these bacteria called CRISPR locus. This site has two distinct characteristics: non-contiguous repeats that are separated through variable sequences, termed spacers. This system recognizes the acquired sequences as foreign and then degrades them in case of invasion. CRISPR therefore functions as an adaptive immune system for prokaryotes.

In the next step, the RNA generated by the CRISPR region (crRNA) is loaded into a Cas protein and this complex is directed through the crRNA to the desired site in the targeted

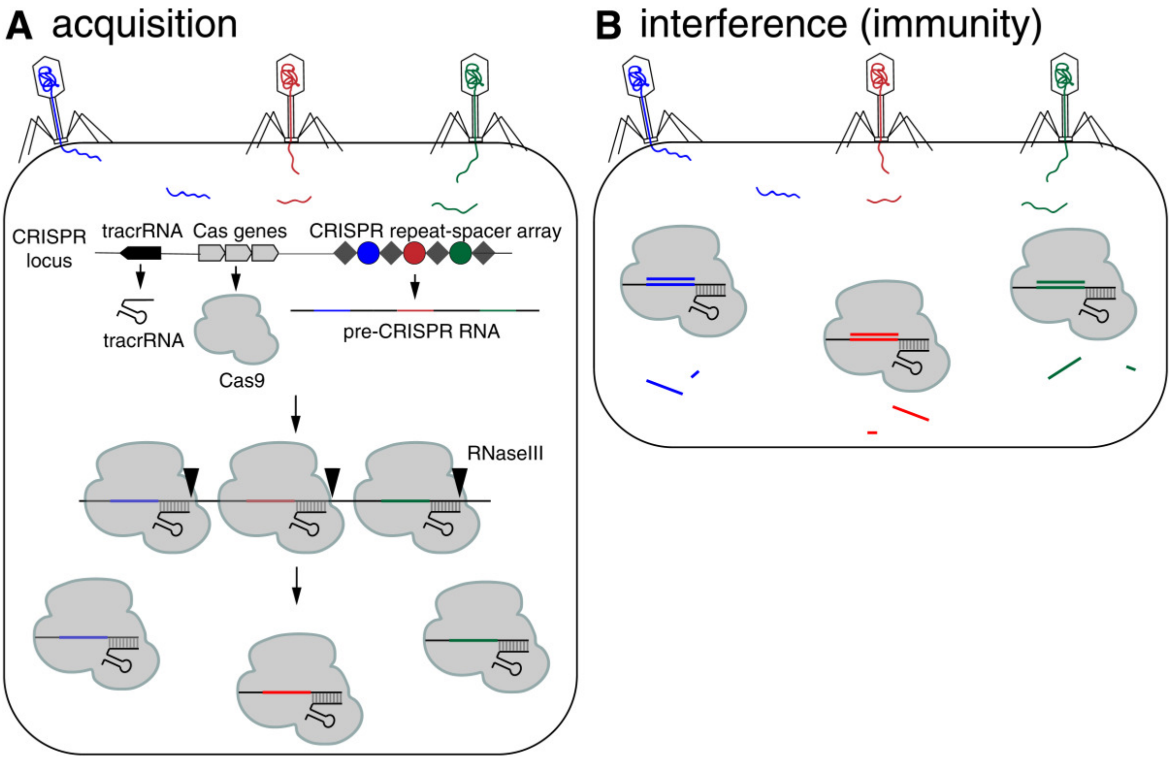

Figure 2.2: The two phases of adaptive immunity in bacteria and archaea [2])

DNA. These CRISPR-Cas systems are divided into three main subgroups, type I, type II and type III. My study focuses only on type II. To produce the crRNA, the CRISPR site containing the sequence of the previous invaders is transcribed to create the pre-crRNA, and then a second trans-activating CRISPR RNA (tracrRNA) is produced from the region just after the CRISPR region. The tracrRNA is complementary to the repetitive CRISPR region. It then binds to the precrRNA. The resulting product is a double-stranded RNA which is cleaved with the help of an enzyme that recognizes double-stranded RNA(RNase III), thus creating the final crRNA.

Eventually the binding of the crRNA to the Cas9 protein produces a complex that locates the invading and cleaved sequences (Figure 2.3). The aforementioned complex recognizes invaders in the cell and more specifically detects a 20 nucleotide sequence followed by an adjacent pattern sequence (PAM) due to complementarity with the crRNA. Different Cas9 proteins recognize a different PAM sequence for this reason and the most widely used are Streptococcus pyogenes (SpCas9) or Staphylococcus aureus (SaCas9), the first recognizing the 5'-NGG-3 'PAM sequence and the second due to the fact that it is a smaller protein in size and is better packaged for viral delivery [12].

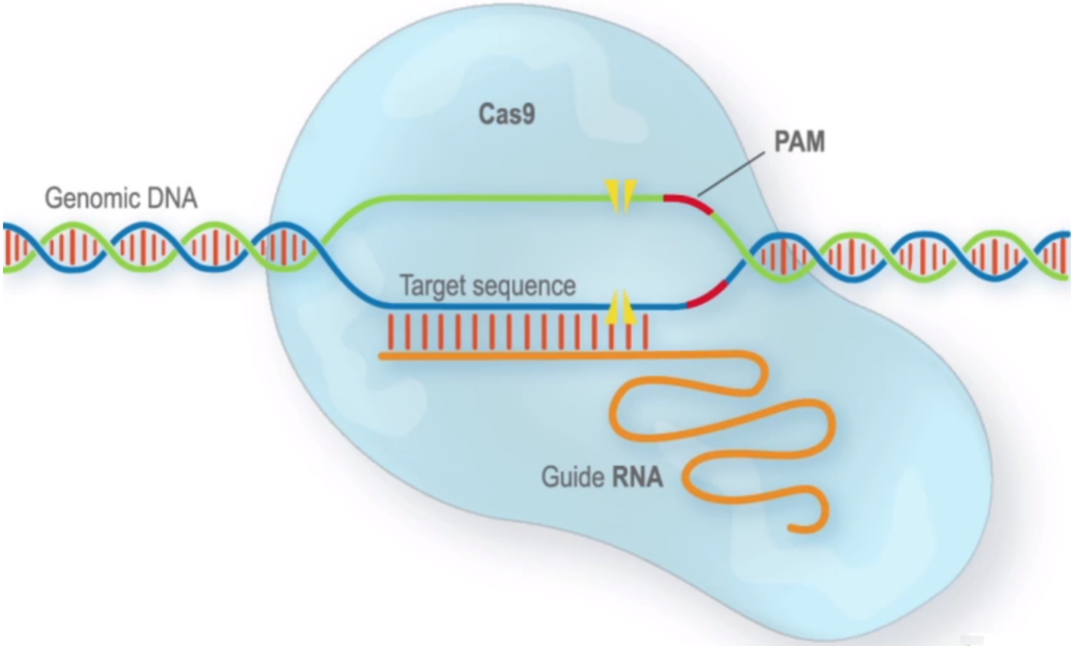

Figure 2.3: Binding of the CRISPR-Cas9 to the target [3])

2.2.2 Eukaryotic

Clustered regularly interspaced short palindromic repeats (CRISPR) and CRISPR - associated protein 9 (Cas9) system CRISPR-Cas9 is an effective mechanism for quick and easy application of genome engineering in eukaryotic cells[2]. CRISPR-Cas9 genome processing procedures use a non-specific endonuclease (Cas9) to cut the genome and a small RNA (gRNA) to guide this nuclease to a user-defined cut region. This procedure is one of the main tools for genome processing due to various options, such as manipulating, detecting, and displaying particular DNA and RNA sequences in the cell.

Double strand breaks in DNA created by the Cas9 protein are restored by the cell in two main ways, either by non-homologous end joining (NHEJ), which leads to random indels at the site of cleavage, or by homologous-directed repair (HDR).

2.2.3 CRISPR for DMD

Genes are the genetic directive that produces proteins and are composed of introns and exons. Exons are the pieces needed to produce the protein. As previously mentioned, the DMD gene is the most extensive in the human body containing 79 exons. It has been observed that patients with DMD have large mutations that lead to the deletion of much of the DMD gene, and a smaller percentage show duplications or point mutations. These mutations affect the production of the full length of the protein as they disrupt the DMD reading frame, typically creating a premature codon. In order to avoid the mutation that causes early termination, researchers developed an approach called exon skipping. Exon skipping is used to repair the reading frame by skipping one or more exons in which there is a mutation that affects the production of functional protein and can be achieved via either classical/indirect (splice site disruption) or direct (exon deletion) methods.

As mentioned previously, DSBs in DNA created by the Cas9 protein are repaired by the cell in two main ways, either by NHEJ, leading to random indels at the cleavage site, or by HDR where a DNA template is used to precisely edit the targeted site. Although HDR is more accurate it is not often applied to post-mitotic cells, such as skeletal muscle cells. There is also still a limit in the length of the template that can be used by HDR, which makes it impossible to correct large deletions that span multiple exons. For this reason, it is not considered as a potential method of repair for DMD where NHEJ is primarily applied and exon skipping (single-cut), exon deletion (double-cut), and exon reframing (single-cut) are some of the processes through which these indels might restore dystrophin production[4](Figure 2.4).

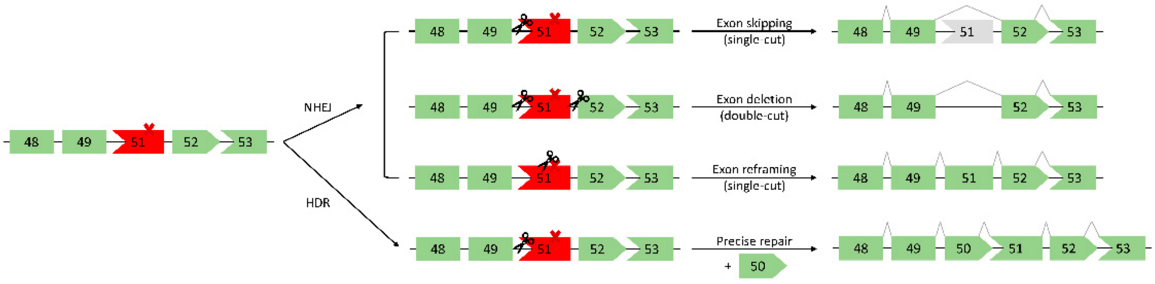

Figure 2.4: A diagram of DNA repair mechanisms for CRISPR/Cas9 gene editing in a hypothetical DMD patient with an exon 50 deletion mutation. [4])

2.3 Off-Target effects

One of the greatest impediments to CRISPR-Cas9 clinical translation is its off-target effects, which can have uncontrollable and unforeseen outcomes, including malignant transformation [13]. Off-target effects refers to mutagenesis at sites in the genome other than the desired on-target site (Figure 4.5). More specifically, CRISPR nucleases can identify DNA / RNA sequences in the genome with small mismatches or bulges and cleave these sequences, creating potentially undesirable mutations.

A major hurdle in selecting the right gRNA is the fact that it is quite difficult for most laboratories to have the gRNA genome due to the high cost [14]. For this reason the main purpose is to predict the CRISPR cutting specificity as well as the design of optimal gRNAs. The prediction of the CRISPR cutting specificity can be achieved in two ways either through a method based on alignment or through a method based on scoring [5]. The first alignment-based method uses conventional or specialized algorithms to align gRNA against a certain genome and as a result returns off-target sequences and positions. In the second scoring-based method, which is used to determine off-targets in silico, sgRNAs are scored and further classified based on the use of identified targets from the alignment procedure to pick the most suitable for experiments.The gRNA scoring approaches involve either hypothesis or Machine Learning [5].

Most off-target locations may be found using alignment-based approaches but this does not mean that there will be a split in all of these positions as other characteristics such as the position where the discrepancy is located have a significant impact[5]. In this method a huge number of off-target sequences can occur at the output which we can limit by setting a maximum mismatch value [5].

The approach we are interested in is Machine Learning and it is also the approach we have chosen to develop in the proposed tool. Based on the above, gRNAs are graded and predicted according to a Machine Learning model that takes into account many features beyond alignment.

In each of the two categories tools have been developed that are widely used to select the appropriate gRNA (Table 2.1).

Therefore, Machine Learning prediction of the best gRNAs is the ideal choice as it allows the cheap and fast investigation of off-target silicon effects for unexplored sequences [14]. Current genome editing techniques urge that researchers carefully choose guides to avoid potential off-target effects and test many to optimize on-target activity [15].

2.3.1 Off-target effects in DMD

Due to the capabilities of CRISPR-Cas9 in clinical applications, research interest in this field is particularly high, as are concerns about the safety and effectiveness of this technique. Research in a range of species, as well as non-human primates, has proved that CRISPR-

Table 2.1: CRISPR gRNA design services (including off-target scoring)

Shorthand	**Main Features**
Elevation and Azimuth	Machine-learning-based models, intergrates both CFD an epigenetic features
MIT server	Hand-Crafted rules
CCTop	Empirical score based on number of mismatches
CRISTA	Machine Learning, sequence composition and epigenetic features
CFD	20bp sgRNA without PAM

Table 2.2: Characteristics used by CRISTA to estimate cleavage propensity.

CRISTA Features	PAM type
	nucleotide composition
	GC content
	chromatin structure
	DNA methylation
	RNA secondary structure

Cas9 technology can be used to accurately modify the genome without serious mutations being observed. However, mutations can occur except the one we want to correct. Addressing these issues needs to be done in order to create large models of human diseases [16].

2.3.2 Off target prediction tools

CRISTA

The CRISPR Target Assessment (CRISTA) program employs the Burrows-Wheeler Aligner as an off-target search tool and uses a variety of characteristics (Table 2.2) to estimate cleavage propensity [6, 17]. CRISTA is based on a Random Forest approach for developing a regression model. All of this adds up to a complicated model that can estimate the likelihood of cleavage at a particular genomic location. As a result, one of the most valuable features of CRISTA's prediction framework is investigating the impact of various qualities. Their study

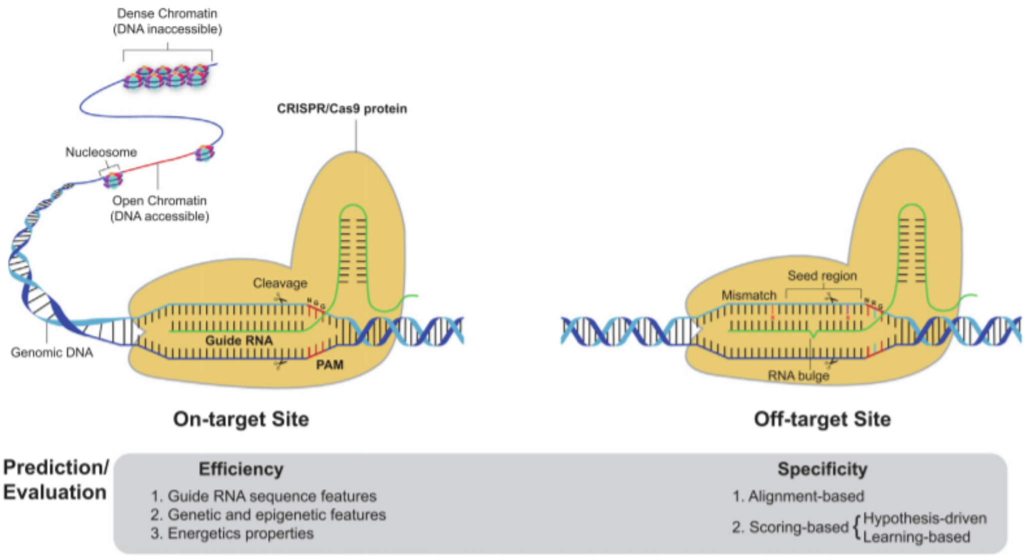

Figure 2.5: On target vs Off target effects [5]

also demonstrates that bulges are not uncommon and should be taken into account throughout the prediction process. CRISTA's estimated score indicates the frequency of genomic indels at a specific location compared to a highly effective sgRNA's on-target cleavage. Additionally, they assessed CRISTA's prediction performance in a leave-one-sgRNA-out cross-validation process, and compared it to competing techniques. They determined the squared Pearson correlation coefficient (r^2) between the experimentally observed and predicted cleavage frequencies over all the samples in their dataset. With a r^2 of 0.65, the predicted scores in cross-validation agreed with the observed values. In comparison, OptCD had a r^2 of 0.13, while CCTop scores had a r^2 of 0.23 and CFD score had a r^2 of 0.52 (Figure 2.7).

CRISPOR

CRISPOR [18] is a web tool where someone can conduct genome editing experiments. More specifically, given a input sequence it discovers all the suitable guide RNAs and evaluates them based on multiple scores like the possible off-targets and on-target efficiency. CRISPOR also appears to have a constantly growing number of genomes uploaded (more than 150 in the previous two years).

Additionally, CRISPOR is a web tool which provides a complete solution including gRNA selection, cloning, and expression, but also the primers that can be used for the possible off-targets and for assessing the guide activity.

CRISPOR examines the whole genome and locates regions similar to the input sequence

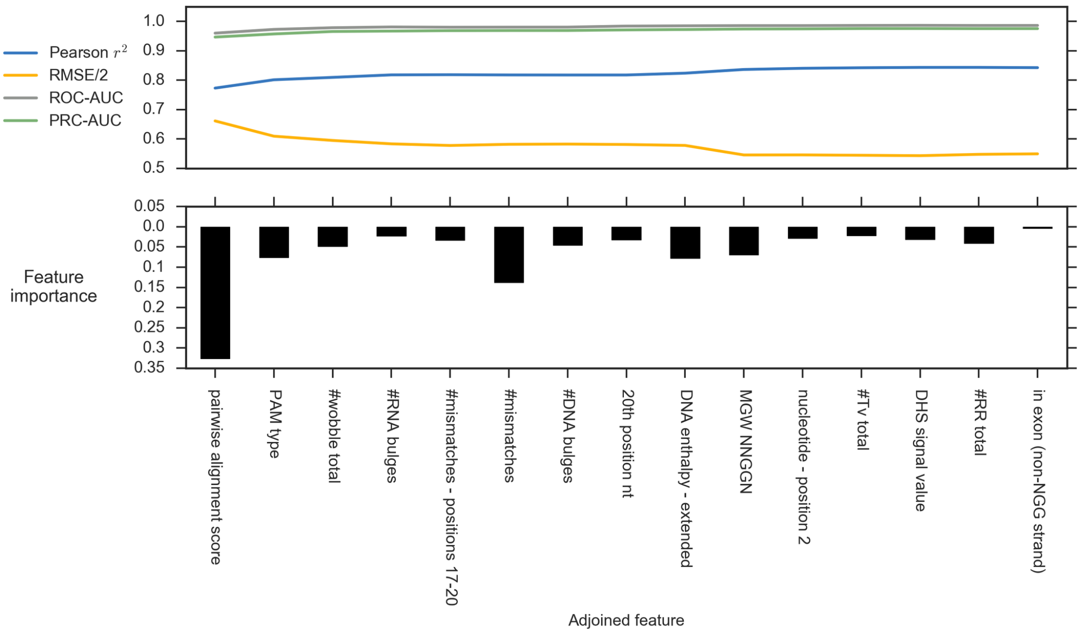

Figure 2.6: Upon the inclusion of each feature from left to right, the top figure shows the ROC-AUC, PRC-AUC, r^2, and root mean square error (RMSE). The bars show feature relevance, or how much each feature contributes to the prediction accuracy as calculated by the Random Forest algorithm [6].

(the potential off targets) with up to 4 mismatches tolerance. Consequently, they rate and rank the gRNAs based on their scores. For each gRNA the potential off-target sequences are ranked using the CFD score [19], which was found to be the more accurate in the comparison they made between four different scores. Finally, they proposed through their experiments that as the specificity increases the likelihood of significant off-targets effects decreases. They also demonstrate that while the forecast accuracy isn't outstanding, the existing predictions are useful, but there's no assurance that using CRISPOR alone would prevent off-target consequences.

Elevation

Elevation is a method for scoring individual guide–target pairs as well as aggregating them into a single, comprehensive summary guide score [14]. It is a machine-learning-based off-target summary model. Individual scores for each off-target location are predicted, as well as an overall score for gRNAs. As shown in the comparison from 5 independent datasets (Kleinstiver (Kleinstiver, et al., 2016), Listgarten (Listgarten, et al., 2018), Haeussler (Haeussler, et al., 2016), Tsai (Tsai, et al., 2015),Slaymaker (Slaymaker, et al., 2016)) the elevation model

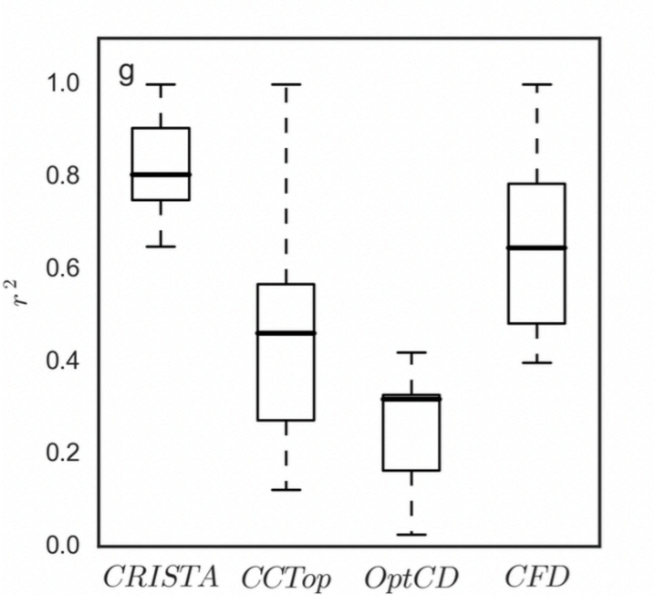

Figure 2.7: Pearson correlation coefficient computed for each sgRNA: CRISTA (averaged $r^2 = 0.80, sd = 0.13$), CCTop (averaged $r^2 = 0.46, sd = 0.22$), OptCD (averaged $r^2 = 0.32$, $sd = 0.28$), CFD score (averaged $r^2 = 0.65$, $sd = 0.28$) [6].

outperformed the MIT, CFD, and CCTop. (Figure 2.8) [7].

Elevation is suitable for human genome editing. It also considers both the sequence and DNA accessibility/epigenetic data to see whether there are any off-target consequences. It uses a two-layer regression model to estimate the off-target activity of a single mismatch first and integrate predictions for gRNA-target pairings with many mismatches.

2.4 Machine Learning

Definition 2.1. *"A computer program is said to learn from experience (E) with respect to some class of tasks (T) and performance measure (P), if its performance at tasks in T, as measured by P, improves with experience E", Tom Mitchell [20].*

Machine Learning is a subset of AI that uses sample data, referred as "training data", to create a mathematical model that can make exact predictions [21]. Electronic data gathered and prepared for analysis is commonly referred to as sample data. Machine Learning approaches have helped pattern recognition, computer vision, economics, computational biology, and biological and medicinal applications.

The Machine Learning algorithms are "soft programmed" in the notion that they automat-

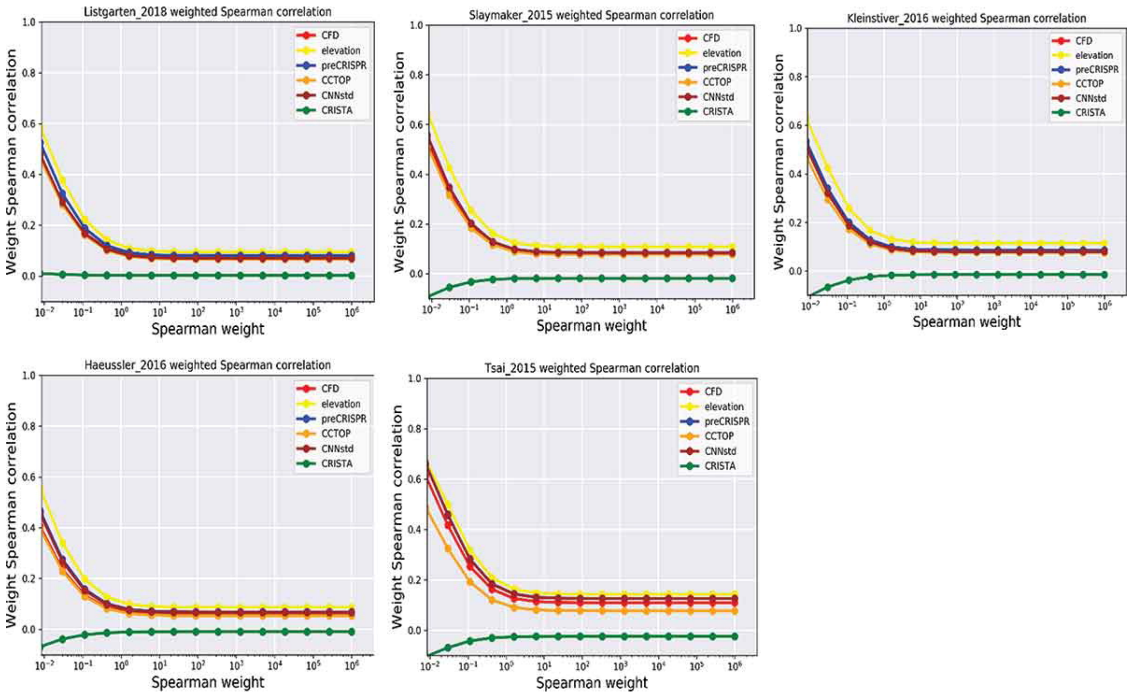

Figure 2.8: Comparison of different off-target prediction methods. The weighted Spearman correlation on the Y-axis is defined by the weight of the X-axis counterpart. The weight ranges from 10^{-2} to 10^{6}. The elevation model consistently outperforms the other models in this experiment [7].

ically change or adjust their design to complete the desired task. The process of adaptation is known as training, and it involves providing samples of input data along with desired outputs. The algorithm then optimizes its configuration such that it can not only provide the intended result when given the training inputs, but also adapt to achieve the desired result when given new, previously unknown data [22].

A computer algorithm can adapt in a variety of ways regarding to training. The input data will be chosen and weighted to produce the most conclusive results. Iterative optimization can be used to adjust the algorithm's variable numerical parameters. It may have a grid of potential computational paths that it arranges for the best outcomes. This could take the supplied data to generate probability distributions and use them to forecast outcomes [22].

Machine Learning aspires to replicate the way humans learn to analyze sensory (input) data in order to achieve a goal [22]. It is classified as supervised, unsupervised, or semi-supervised depending on the nature of the data labeling (Figure 2.9).

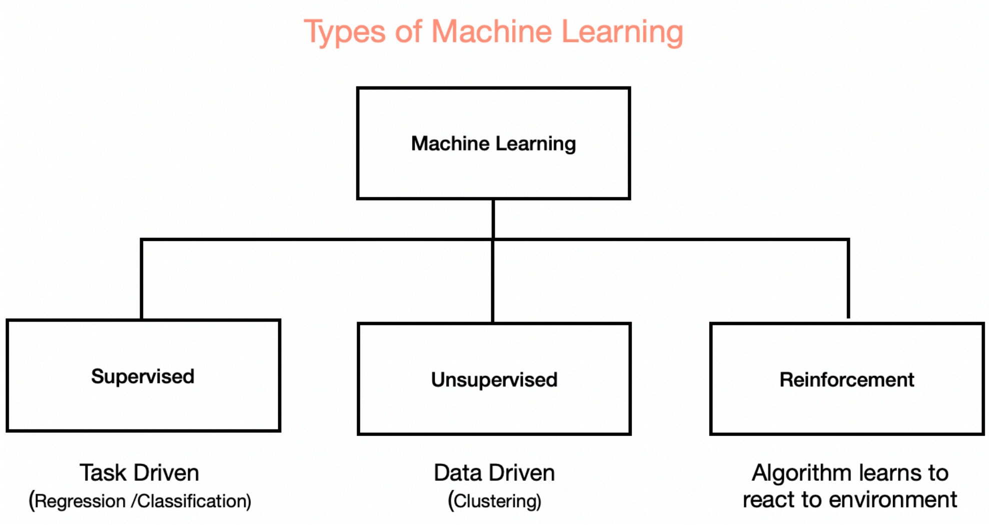

Figure 2.9: Categories of Machine Learning algorithms according to training data nature

2.4.1 Models

The subject of our research was formulated as a regression problem. The regression problem is a variation of the classification problem where the model outputs a continuous-valued result instead of a finite-valued one. In other terms, a regression model approximates a continuous-valued multivariate function [23].

Decision Tree Regressor

Non-parametric supervised Machine Learning approaches for creating prediction models from data include classification and regression trees. The models are created by recursively splitting the data space and fitting a basic prediction model to each division. Consequently, the partitioning may be graphically represented as a decision tree.

The basic algorithm of decision trees is:

1. Start at the root node as parent node.

2. Split the parent node at the feature a to minimize the sum of the child node impurities (maximize information gain).

3. Assign training samples to new child nodes.

4. Stop if leave nodes are pure or early stopping criteria is satisfied, else repeat steps 1 and 2 for each new child node.

In a decision tree, the attributes that carry the most information about the variable we want to predict are selected and placed as nodes in the tree. This results in only a few attributes participating in the procedure while solving the dimensionality reduction problem. The decision tree structure helps the analyst comprehend and interpret the given information at each level, in contrast to the black box process of the neural networks [24].

The algorithm that builds regression trees is CART (Classification and Regression Trees) [25]:

1. Find the appropriate split for each characteristic to reduce impurity.
2. Identify the characteristic that reduces impurity the most.
3. Split the node using the best split on that feature.
4. Repeat this process for each of the leaf nodes.

For continuous target variables, a reduction in variance approach is applied (regression problems). To find the optimal split, this method uses the usual variance formula. Then, the population is split according to the split with the lowest variance.

Variance is calculated in the following steps:

1. Calculate variance for each node.
2. As a weighted average of each node's variance, calculate variance for each split.

Finally, a tree is created with decision nodes and leaf nodes (Figure 2.10) [24]. The tree contains a root node, which corresponds to the best predictor and is the highest decision node.

XGBoost Regressor

XGBoost is a scalable tree-boosting Machine Learning method. The most crucial factor in XGBoost's effectiveness is its capability to scale in any situation. The system is ten times faster than previously commonly used approaches on a single machine, and it can handle billions of samples in distributed or memory-limited scenarios. Numerous key system and algorithmic innovations, such as sparse data management and parallel and distributed computing, contribute to XGBoost's scalability. However, the most significant characteristic is that it allows data scientists to handle hundreds of millions of instances on a single machine using out-of-core processing.

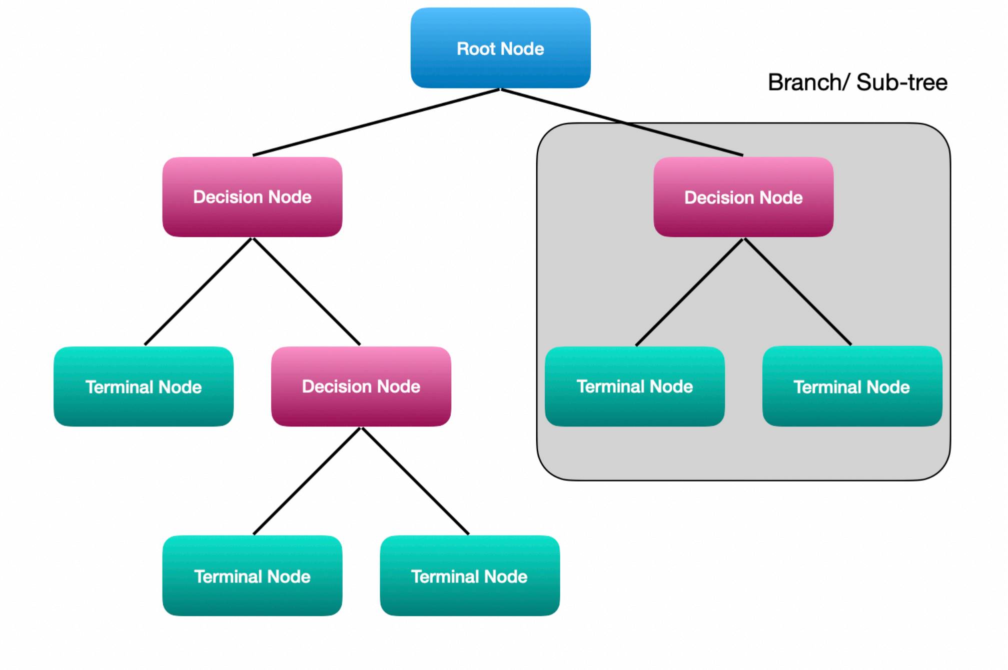

Figure 2.10: Decision Tree Structure

Gradient Boosting

Gradient boosting is among the most successful methods for developing predictive models. The notion of boosting emerged from the consideration of whether a poor learner could be improved. The objective was named the Hypothesis Boosting Problem by Michael Kearns [26]. A weak hypothesis, also known as a weak learner, is one where the results are at least marginally more suitable than random chance. The idea behind this hypothesis boosting was to filter observations, leaving only those that the weak learner could handle, and then focus on constructing additional weak learners to address the remaining challenging observations.

Gradient boosting is composed of three parts:

1. A loss function that has to be improved.

2. A weak learner that makes predictions.

3. A model that adds weak learners to reduce the loss function.

Decision regression trees are applied as the weak learner in gradient boosting because their continuous outputs can be combined, allowing for the addition of future model outputs and the adjustment of residuals in predictions.

Existing trees in the model are left unchanged, while new trees are added one after another. When adding trees, a gradient descent approach is utilized to minimise the loss. Gradient descent has typically been used to reduce a set of parameters, like the number of variables in a model or the coefficients in a regression equation.

Based on a dataset with n observations and m features $D = \{(x_i, y_i)\}, (|D| = n, x_i \in \mathrm{I\!R^m}, \mathrm{y_i} \in \mathrm{I\!R})$, we can represent a tree model with K additive functions for the output prediction and the space of the regression trees as follows [27]:

$$\hat{y}_i = \phi(x_i) = \sum_{k=1}^{K} f_k(x_i), f_k \in F, \text{ where } F = \{(f(x) = w_{q(x)}\}$$

and $q : \mathrm{I\!R^m} \rightarrow \mathrm{T}, \mathrm{w} \in \mathrm{I\!R^T}$ is the structure of each tree and T is the number of leaves in that tree that have weight w.

We minimize the following objective.

$$\mathcal{L}^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i) + \Omega(f_t)), \text{ where } \Omega(f) = \gamma T + \frac{1}{2}\lambda\|w\|^2$$

The difference between the prediction $\hat{y}_i$ and the target y_i is measured by l, which is a differentiable convex loss function.

By applying second order approximation we get:

$$\mathcal{L}^{(t)} \simeq \sum_{i=1}^{n}[l(y_i, \hat{y}_i^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2}h_i f_t^2(x_i)] + \Omega(f_t)$$

where

$$g_i = \vartheta_{\hat{y}^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})$$

and

$$h_i = \vartheta^2_{\hat{y}^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})$$

By removing the constant terms we get the following simplified objective at step t:

$$\tilde{\mathcal{L}^{(t)}} = \sum_{i=1}^{n}[g_i f_t(x_i) + \frac{1}{2}h_i f_t^2(x_i)] + \Omega(f_t)$$

$$= \sum_{i=1}^{n}[g_i f_t(x_i) + \frac{1}{2}h_i f_t^2(x_i)] + \gamma T + \frac{1}{2}\lambda\sum_{j=1}^{T} w_j^2 \text{ [27]}$$

where $I_j = \{i|q(x_i) = j\}$ is the instance set of leaf j.

We can calculate the optimal weight w_j^* for a fixed structure $q(x)$ as follows:

$$w_j^* = -\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda}$$

and the scoring function as a measure of a tree structure q:

$$\tilde{\mathcal{L}^{(t)}}(q) = -\frac{1}{2}\sum_{j=1}^{T}\frac{(\sum_{i\in I_j} g_i)^2}{\sum_{i\in I_j} h_i + \lambda} + \gamma T \text{ [27]}$$

The smaller the value is, the more suitable is the tree structure. As the enumeration of all tree topologies is difficult, a greedy approach is employed to repeatedly add branches to the tree. After splitting, the instance sets of the left and right nodes are IL and IR. The gain formula enumerates the possible segmentation points and picks the minimal target function and maximum gain partition.

$$G = \frac{1}{2}\left[\frac{(\sum_{i\in I_L} g_i)^2}{\sum_{i\in I_L} h_i + \lambda} + \frac{(\sum_{i\in I_R} g_i)^2}{\sum_{i\in I_R} h_i + \lambda} - \frac{(\sum_{i\in I} g_i)^2}{\sum_{i\in I} h_i + \lambda}\right] - \gamma \text{ [27]}$$

In practice, this method is commonly used to evaluate split candidates. During splitting, the XGBoost model generates a large number of simple trees that are utilized to score each leaf node. The equation's first, second, and third components indicate the left, right, and original leaf scores, respectively [28].

Random Forest Regressor

A random forest is a classifier composed of a number of tree-structured classifiers $h(x, k)$ $k = 1, ...,$ each of which votes for the most popular class at input x with a single unit vote [29]. Random forests are formed in regression by building trees from a random vector, with the tree predictor $h(x)$ utilizing numerical values instead of class labels. The output values are numerical, and the training set is selected separately from the random vector Y, X distribution. In addition to bagging, we apply random feature selection in regression forests [29].

Boosting and arcing algorithms can help minimize both bias and variance. The adaptive bagging approach in the regression was created to eliminate bias and works well in both classification and regression. However, it does vary the training set as it progresses, similar to arcing. Forests provide similar results to boosting and adaptive bagging, but the training set is not modified over time. Their precision indicates that they are working to minimize bias [29].

In contrast to a conventional tree, a random forest splits each node using the best group of predictors randomly selected at that node. Because it just needs a few parameters – the number of trees in the forest and variables at each node in the random subset – it's a simple approach to employ [30].

The steps of the Random Forest Regression method are as follows [31]:

1. Take an n_t bootstrap sample using the real data.
2. A regression tree must be created with some modifications for each bootstrap sample: At each node, randomly choose m_t of the predictors and determine the optimal split among the variables.
3. Compute the most recent data by adding the n_t tree predictions (average for regression).

Random forest= DT(base learner)+ bagging(Row sampling with replacement)+ feature bagging(column sampling) + aggregation(mean/median, majority vote) [32] (Figure 2.11)

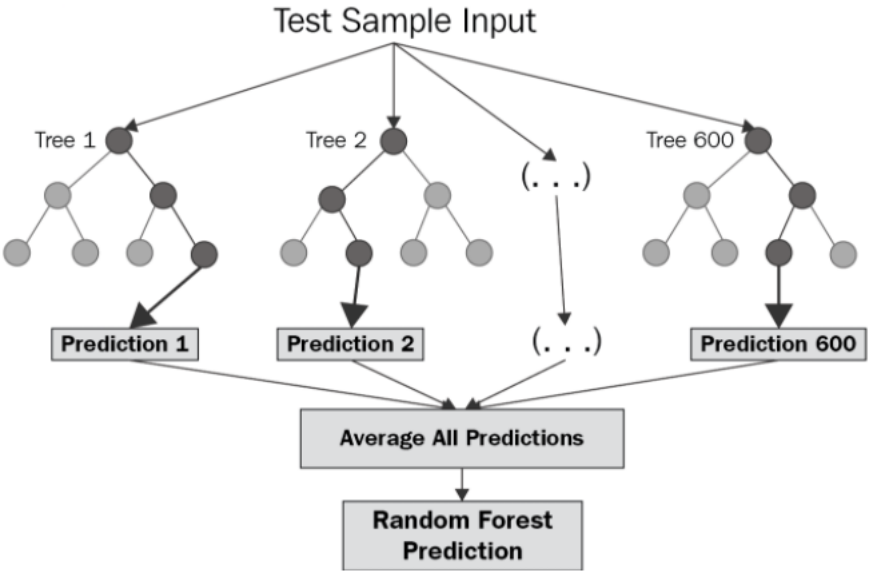

Figure 2.11: Random Forest Regressor [8]

Support Vector Regressor

SVMs are a form of pattern classifier developed by Vapnik and his colleagues based on a revolutionary statistical learning approach [33]. Through the increase of the margin between the separating hyperplane and the data, the SVM's optimization problem entails to reduce the upper bound of the generalization error. As a result, SVMs are better than the classic observed risk minimization approach most neural networks use in terms of generalizing successfully even in high dimensional spaces under few training sample conditions [34].

Empirical Risk Minimization is a term used to describe how traditional learning algorithms are meant to reduce error on the training dataset (ERM). SVMs, on the other hand, is based on the statistical learning theory's Structural Risk Minimization (SRM) concept. It improves generalization skills , and SRM is achieved by minimizing the upper bound of the generalization error [34].

SVR is based on a training set, $(x_1, y_1), ..., (x_i, y_i) \subset X \times R$ (X space of input patterns). In Support Vector regression, our aim is to find a fitting function $f(x)$ with a deviation less than the target (y_i) for the corresponding training data set. It's best if the function is rather flat. Alternatively, any mistake of less than ϵ is acceptable [34]. The linear function is:

$$f(x) =< w, x > +b = \sum_{j=1}^{M} w_j x_j + b, y, b \in R, x, w \in R^M$$

The dot product of X is represented by $< w, x >$, and flatness is given by w . To ensure this, we must limit the norm to a bare minimum. Because the SVM determines the hyperplane with the highest margin, it may be obtained by minimizing $\frac{1}{2}w^2$.

Chapter 3

Methodology

3.1 Data Collection

For the first step towards the implementation of this thesis, CRISPR-Cas9 off-target data for the DMD disease were collected. Due to the fact that there was not an available integrated dataset for the off-target effects we compiled our own dataset. Both research papers [35, 36, 37, 38, 39, 40, 41] and patents [42, 43, 44, 45] were reviewed and data for HEK293T cells (a derivative human cell) and mouse organisms were collected. The data collected contained information about the gRNA sequence, the corresponding off-target sequence, and the indels that were observed. In order for all sequences to be converted to sequences of the same length 23-nt, the Ensembl BLAST tool [46] was used (Table 3.1).

3.2 Data manipulation

In order to develop off-target prediction models, we need to manipulate the collected data to extract features that can be useful for off-target evaluation.

From the data we collected, we extracted some basic information that we used in the Machine Learning models. The DNA melting temperature prediction accuracy is critical to the experimental performance and outcome of numerous molecular biology procedures (T_m).

- The Wallace–Ikatura rule is frequently used as a guideline when estimating the Melting Temperature. The temperature at which the bonds between the chains are broken is calculated using the following formula:

$$T_{m_{wallace}} = 64.9 + 41 * (yG + zC - 16.4)/(wA + xT + yG + zC)$$

gRNA	off-target	indels (%)	organism
TCTTTGAAAGAGCAACAAAATGG	TTTTTGAAAGAGCAACAATAAGG	0.25	mouse
TCTTTGAAAGAGCAACAAAATGG	TTTTTGAAAGATCAACAAAATAG	0.68	mouse
TCTTTGAAAGAGCAACAAAATGG	TCTGTGAAAGAGTAACAAAATGG	0.27	mouse
GATACTAGGGTGGCAAATAGAGG	GATACTAGTGTGGCTCATAGAGG	0.19	mouse
GATACTAGGGTGGCAAATAGAGG	GATACGATGGTGGCAAATCGTGG	0.28	mouse
GATACTAGGGTGGCAAATAGAGG	GATACTAGGGTGGGGAATAAAGG	0.44	mouse
GCCTACTCAGACTGTTACTCTGG	TCCTACTCACACTGTTACTCAGG	9.3	HEK293T cells
GCCTACTCAGACTGTTACTCTGG	ACCTGCTCACACTGTTACTCCAG	0	HEK293T cells
GCCTACTCAGACTGTTACTCTGG	GCATTCTCAAACTGTTACTCAGG	0	HEK293T cells
GATTGGCTTTGATTTCCCTAGGG	AATTGGCATTGATTTCCCTAGAG	0.8	HEK293T cells
GATTGGCTTTGATTTCCCTAGGG	CATTGGCTTTAATTTCCCTATAG	0	HEK293T cells
GATTGGCTTTGATTTCCCTAGGG	GATAGGCTGTGATTTCCCTAGAG	0	HEK293T cells

Table 3.1: Sample of the data collected

for oligos > 13 [47].

- The biopython method [48], which calculates the melting temperature using empirical formulas based on GC content, is:

$$T_{m_{GC}} = 4GC + 2TA$$

- The nearest neighbors method [49] is the principal technique we employ to determine T_m for oligonucleotides with sequence lengths ranging from 15 to 120 bases (upper length limit of our standard DNA oligos offering). This approach is the most accurate since it considers the oligonucleotide sequence rather than simply the base composition like the other way. In addition, the nearest neighbours technique considers thermodynamic and other influences on T_m, such as oligonucleotide and monovalent cation concentrations. The formula used is:

$$T_m = \frac{\Delta H}{A + \Delta S + R \ln \frac{C}{4}} - 273.15 + 16.6 \log[Na^+]$$

where:

  - Tm = melting temperature in °C
  - ΔH = enthalpy change in kcal mol^{-1}
  - A = constant of -0.0108 kcal $K^{-1} * mol^{-1}$

gRNA	off-target	indels(%)	Tm_Wallace	Tm_GC	Tm_NN	GC_content
TCTTTGAAAGAGCAACAAAATGG	TTTTTGAAAGAGCAACAATAAGG	0.25	52.0	41.954892	45.955119	0.30
TCTTTGAAAGAGCAACAAAATGG	TTTTTGAAAGATCAACAAAATAG	0.68	52.0	41.954892	45.955119	0.30
TCTTTGAAAGAGCAACAAAATGG	TCTGTGAAAGAGTAACAAAATGG	0.27	52.0	41.954892	45.955119	0.30

Table 3.2: Part 1 of the processed data

  - ΔS = entropy change in kcal $K^{-1} * mol^{-1}$
  - R = gas constant of 0.00199 kcal $K^{-1} * mol^{-1}$
  - C = oligonucleotide concentration in M or mol L^{-1}
  - -273.15 = conversion factor to change the expected temperature in Kelvins to °C
  - $[Na+]$ = sodium ion concentration in M or mol L^{-1}

- The paired G and C nucleotides in double-stranded DNA are connected by three hydrogen bonds, while the A and T nucleotides have just two. As a result, GC pairings are “stronger” than AT pairs, and the GC/AT ratio of a DNA sequence has a significant impact on its physical characteristics (such as its “melting point”) [50]. The method to calculate the GC frequency is:

$$GC_{frequency} = float(C_{count} + G_{count})/length$$

- The amount of mismatches between the target sequence and the prospective off-target sequence is thought to have a major impact on the incidence of off-target effects. The number of mismatches is calculated using the hamming distance method.

  We applied the aforementioned methods to the collected data and transformed the dataset to include numerical data. More specifically, in order to apply the melting temperature methods, we only kept a sub-string with size 20-nt from the 23-nt gRNA and 23-nt off-target sequence. Additionally, we computed the mismatches, and they appear to have values in the range $(2, 17)$ as shown in the Figure 3.1. The mean value of the mismatches is 4.8, and the standard deviation is equal to 2.13. A sample of the processed data is presented in the Table 3.2 and 3.3.

3.3 Evaluation metrics

For the evaluation of the methods, we chose the Spearman Correlation coefficient. Spearman’s rank correlation coefficient, named after Charles Spearman, is a nonparametric mea-

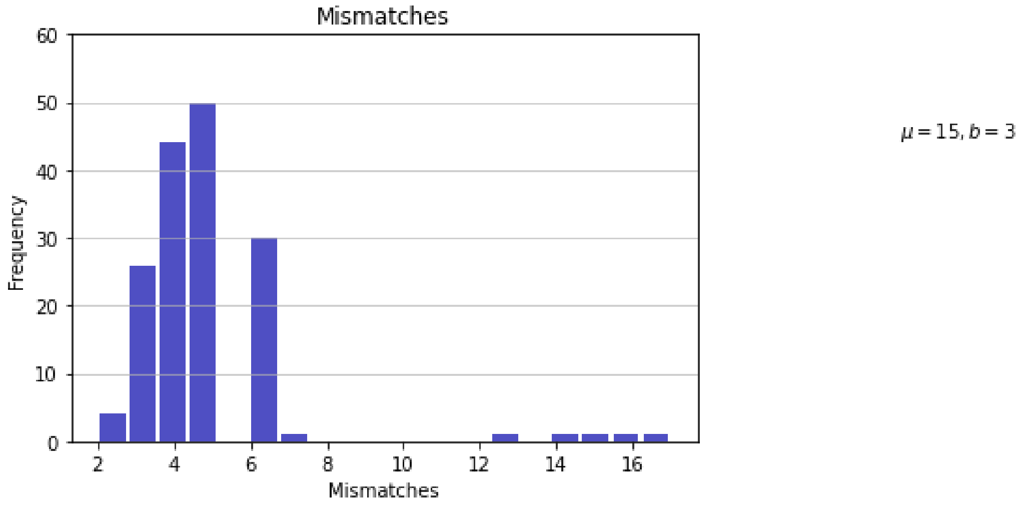

Figure 3.1: Frequency of mismatches on the collected data.

gRNA	off-target	Tm_Wallace_off	Tm_GC_off	Tm_NN_off	GC_content_off	mismatches
TCTTTGAAAGAGCAACAAAATGG	TTTTTGAAAGAGCAACAATAAGG	50.0	39.904892	43.566767	0.25	3
TCTTTGAAAGAGCAACAAAATGG	TTTTTGAAAGATCAACAAAATAG	48.0	37.854892	41.929952	0.20	3
TCTTTGAAAGAGCAACAAAATGG	TCTGTGAAAGAGTAACAAAATGG	52.0	41.954892	44.378333	0.30	2

Table 3.3: Part 2 of the processed data

sure of rank correlation usually represented in statistics by the Greek letter ρ or as r. It assesses the ability of a monotonic function to describe the relationship between two variables.

Spearman correlation calculates correlation in the same way that Pearson correlation does, with the only difference being that Pearson evaluates linear correlations as opposed to Spearman, which evaluates monotonic ones between the rank variables (whether linear or not) (Figure 3.2) .

When one variable is the ideal monotone function of the other, and there are no repeated values, a perfect Spearman correlation returns the values +1 or -1. Spearman's coefficient is suitable for continuous as well as discrete ordinal variables. The n raw scores X_i, Y_i are transformed to ranks for a sample of size n, $\mathrm{R}(X_i), \mathrm{R}(Y_i)$, and r_s is computed as:

$$r_s = \rho_{\mathrm{R}(X),\mathrm{R}(Y)} = \frac{\operatorname{cov}(\mathrm{R}(X), \mathrm{R}(Y))}{\sigma_{\mathrm{R}(X)}\sigma_{\mathrm{R}(Y)}}$$

where ρ is the standard Pearson correlation coefficient applied to rank variables, $\sigma_{\mathrm{R}(Y)}$ are the standard deviations of the rank variables, and $\operatorname{cov}(\mathrm{R}(X), \mathrm{R}(Y))$ is the covariance of the rank variables [51].

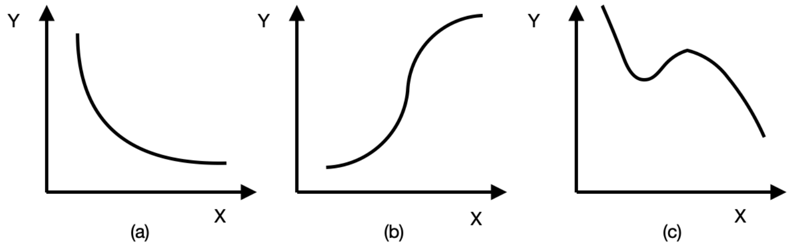

Figure 3.2: (a) Monotonically decreasing, (b) Monotonically increasing, (c) Not monotonic

3.4 Implementation

In the implementation we propose, the sequences we collected start from the first stage of preprocessing. The 23-nucleotide guide RNA sequence and the corresponding 23-nucleotide off-Target sequence are converted to numerical data using the methods of the Biopython library [52]. More specifically the sequences are converted to 20-nt sequence and then the methods Tm_Wallace, Tm_GC, Tm_NN and the getGCFreq function from the CCTOP tool [53] are applied to them. The mismatches between the two initial strings are then calculated using the hamming distance function.

In the next step, we apply the Machine Learning algorithms we chose to the processed data. More specifically, as the available data is limited in number, we decided to use the implementations provided by the sklearn library[54] combined with a Nested Cross-Validation approach in order to select the most suitable model.

Finally having come up with the model which has the best Spearman correlation using the original data we compare our results with those of three tools designed to evaluate off-targets (CRISTA[6], CRIPSOR[18], Elevation[14]).

Chapter 4

Results

4.1 Evaluation with Cross Validation – Kfold

Cross-validation is a statistical method for estimating and comparing learning algorithms that splits data into two parts: training and validation sets[55]. K-fold cross-validation is the most fundamental type of cross-validation. Both training and validation sets must pass through in successive rounds in traditional cross-validation so that each data point can be validated against the other.

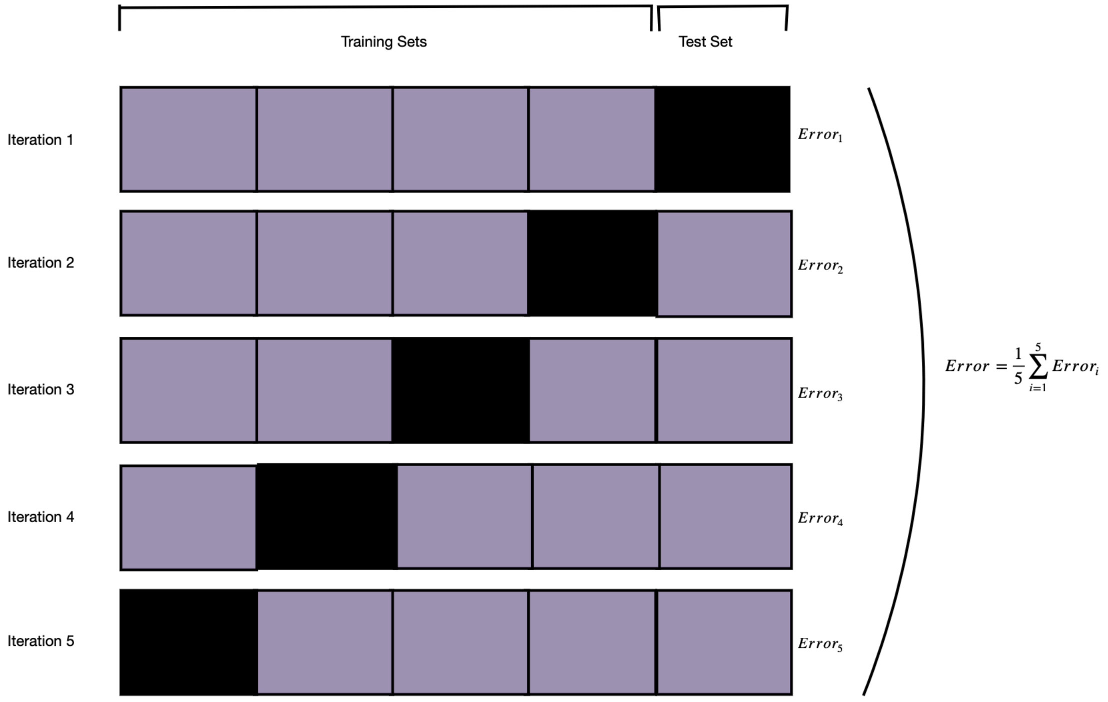

Figure 4.1: 5-Fold Cross Validation

More specifically, in k-fold cross-validation, the data is initially partitioned into k equally sized segments or folds. Subsequently, k iterations of training and validation are run, with each iteration holding out a different fold of the data for validation and the rest k - 1 folds being used for learning (Fig. 4.1). Ten fold cross-validation (k = 10) is the most standard cross-validation used in Machine Learning.[55]

When we utilize data to develop an ML model, we often split it into training and validation/test sets. The validation/test set is employed to validate the model on previously unseen data, whereas the training set is used for training. A straightforward 80 % -20 % split is the standard method. K-fold cross-validation produces a less biased model than other approaches, such as a simple train/test split.

By using cross-validation, we create K distinct models, allowing us to make predictions on all of our data. In addition, Cross-Validation can be used to select the best parameters of a Machine Learning model. Nested cross-validation is a method for optimizing model hyperparameters and determining models that aim to solve the problem of overfitting the training dataset. Hyperparameter search is less prone to overfitting the dataset in this approach since it is only exposed to a subset of the dataset supplied by the outer cross-validation phase.

A major drawback is that the number of model assessments conducted dramatically increases when using nested cross-validation as the number of the fits gets from $n*k$ to $k*n*k$ since the method is repeated k times for each fold.

4.2 Comparative analysis

Because the data set is small enough to apply deep learning algorithms, we apply Machine Learning and cross validation to the data. We use k-fold cross-validation with 10 folds and apply a XGBoost Regressor, a Decision Tree Regressor, a Support Vector Regressor and a Random Forest Regressor. For the evaluation we use the Spearman scorer we defined, which calculates spearman correlation between the real observed indels and the predicted ones.

After completing the preprocessing stage follows the stage of the cross validation and the training of the models. As mentioned above the split of the dataset into train and test sets is achieved with the use of k-fold cross-validation where in my implementation $k = 10$ so we have $(k - 1) = 9$ training sets and 1 test set. As long as all sets have been used as test sets the procedure stops and the final result is the average Spearman score of all iterations.

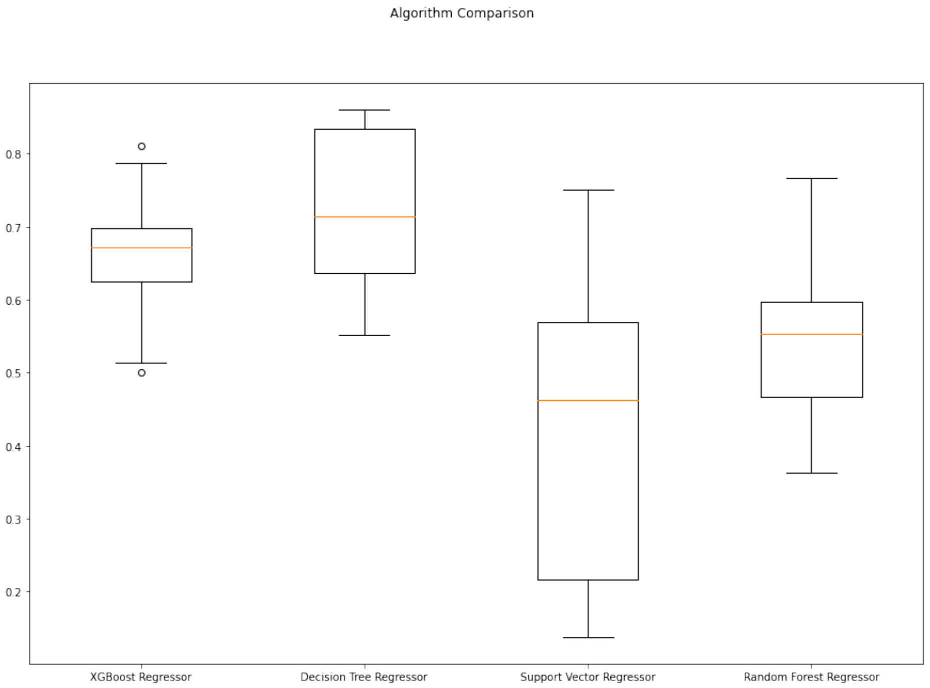

Figure 4.2: Comparison of the models based on the nested cross validation score

Initially for each of the models we applied cross validation with a base model, which means that we only used some default parameters. In order to apply hyper-parameter tuning we split our dataset in training and test subsets based on the $train\_test\_split$ function with $test\_size = 0.2$ and $training\_size$ equal to 0.8. For each method we applied the Hyper Parameter Tuning function we defined we used Grid Search to find the best hyper parameters. We used the 80% training set to fit the grid search algorithm and the evaluated the result using the 20% test set using the Spearman Score. Subsequently we applied nested cross validation in order to find the best combination of hyperparameters that provide the best Spearman corellation and additionally to see the average Spearman score for each one of them in order to compare their performance. The nested-cross validation provides a less biased result in comparison to the hyper parameter tuning. The following tables show the results we got for each one of the models.

For the XGBoost Regressor from the Table 4.1 we observe that the baseline model does not provide a satisfying performance. Additionally we see that the average score of the nested cross validation (0.659) is good enough in comparison to the hyper-parameter tuning score (0.616). We primarily focus on the nested cross validation score for the evaluation of the models, because as mentioned before this score is not prone to overfitting.

For the Decision Tree Regressor from the Table 4.2 we observe that the baseline model

Table 4.1: XGBoost Regressor results

Model	XGBoost Regressor		
	Baseline Model	Hyper Parameter Tuning	Nested Cross Validation

Parameters	Default Parameters	learning_rate = 0.01,
		max_depth = 5,
		n_estimators = 500,
		subsample = 0.5
			Best Parameters:
			learning_rate = 0.01,
			max_depth = 10,
			n_estimators = 1000,
			subsample = 0.799

Spearman Correlation	Mean = 0.443	0.616	Average Score = **0.659**

Table 4.2: Decision Tree Results

Model	Decision Tree Regressor		
	Baseline Model	Hyper Parameter Tuning	Nested Cross Validation

Parameters	Default Parameters	max_features = auto,
		max_depth = 10,
		min_samples_split = 2,
		min_samples_leaf = 2
			Best Parameters:
			max_features = auto,
			max_depth = 110,
			min_samples_split = 2,
			min_samples_leaf = 1

Spearman Correlation	Mean = 0.544	0.628	Average Score = **0.724**

Table 4.3: Support Vector Regressor results

Model	Support Vector Regressor	
	Baseline Model	Nested Cross Validation

Parameters	Default Parameters	Best Parameters:
		kernel = linear,
		C = 1.5,
		gamma = 0.0001,
		epsilon = 0.1

Spearman Correlation	Mean = 0.365	Average Score = **0.416**

does not provide a satisfying performance. Additionally we see that the average score of the nested cross validation (0.724) is good enough in comparison to the Hyper parameter tuning score (0.628). We can conclude that the initial observation is that the Decision tree Regressor outperforms the other methods as shown in the Figure 4.2.

For the Support Vector Regressor from the Table 4.3 we observe that the baseline model has the worst performance in comparison with the other models. Additionally, we see that the average score of the nested cross validation (0.416) is also not good enough as it is less than 0.5. We can confirm these results from the Figure 4.2.

For the Random Forest Regressor from the Table 4.4 we observe that the baseline model does not provide a satisfying performance. Additionally we see that the average score of the nested cross validation (0.546) is not as good as the Hyper Parameter Tuning (0.65). Although we use the nested cross validation as a metric for comparison with the rest models.

4.2.1 Comparison with state-of-the-art tools

Based on the previous research we conducted, we observed that Elevation outperforms the other off-target evaluation tools (i.e., CRISTA, CRISPOR) and this is also evident in Table 4.5, as the Spearman correlation is close to the average Spearman correlation of my Decision Tree Regressor. On the other hand, CRISTA and CRISPOR performed poorly on the DMD dataset with scores 0.196 and 0.095, respectively.

Table 4.4: Random Forest Regressor Results

Model	Random Forest Regressor		
	Baseline Model	Hyper Parameter Tuning	Nested Cross Validation

Parameters	Default Parameters	max_depth = 90,
		max_features = 80,
		min_samples_leaf = 3,
		min_samples_split = 12,
		n_estimators = 30
			Best Parameters:
			max_depth = 90,
			max_features = 2,
			min_samples_leaf = 3,
			min_samples_split = 8,
			n_estimators = 10,
			bootstrap = True

Spearman Correlation	Mean = 0.441	0.65	**Average Score = 0.546**

Table 4.5: Spearman Corellation between the original indels and the tools' predictions.

Off-target tool	CRISTA	CRISPOR	ELEVATION	DMD_DTR
Spearman Correlation	0.196	0.095	0.6	0.724

Table 4.6: Average importance of the dataset features based on the Decision Tree Regressor

Features	Average Importance
Tm_Wallace	0.0157
mismatches	**0.3246**
Tm_NN_off	0.0930
Tm_Wallace_off	0.0733
Tm_GC	**0.3178**
GC_content_off	0.0263
GC_content	0.0021
Tm_NN	**0.1371**
Tm_GC_off	0.0097

4.3 Feature Importance and selection of the best model

Comparing the performance of all models based on the nested cross validation score we conclude that the model with the best performance is the Decision Tree Regressor with the following parameters max_features = auto, max_depth = 110, min_samples_split = 2, min_samples_leaf = 1.

In the next step, we applied once more a nested cross validation in order to check which features of the dataset are the most important features for predicting the output. We observe that the column ‘mismatches’ (0.3246) is the most significant followed by the ‘Tm_GC’ (0.3178) and ‘Tm_NN’ (0.1371).

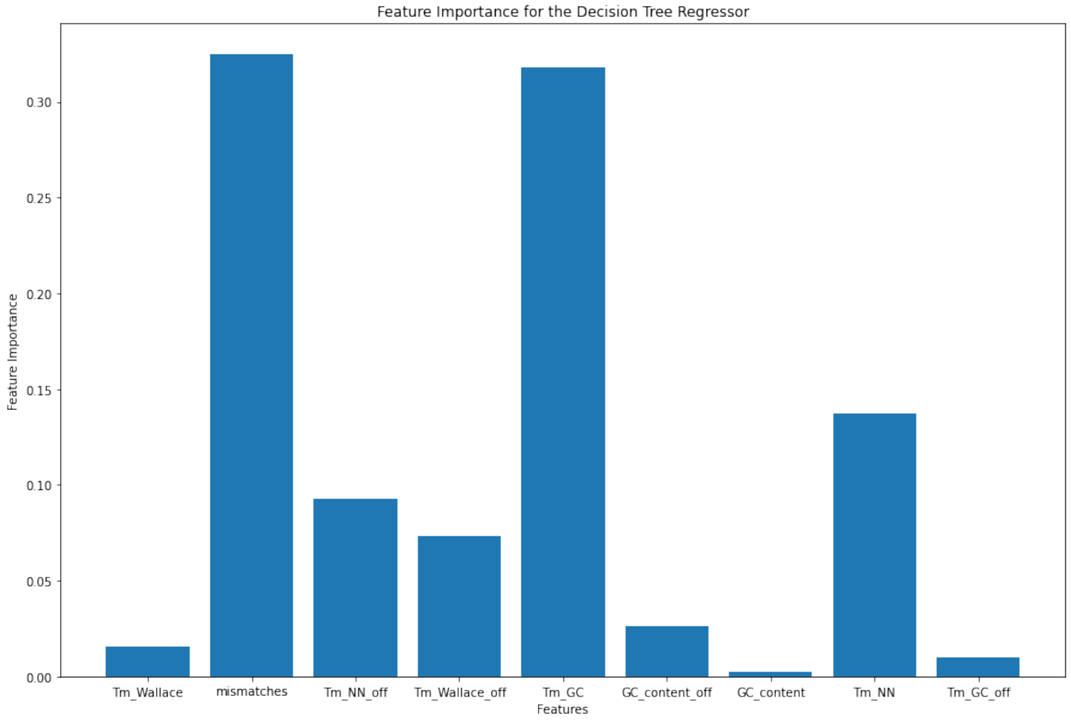

Figure 4.3: Average importance of the dataset features based on the Decision Tree Regressor

Chapter 5

Conclusions

In this thesis, we presented a solution for evaluating off-targets for the DMD disorder using regression, a field of Machine Learning that has been one of the most basic and researched tools. In particular, we collected all the available until now data for the off-targets that have been observed for DMD using the CRISPR-Cas9 gene-editing approach. Subsequently, we prepared the data to get processed by the Machine Learning models and defined the scoring function (Spearman Correlation). From the models we explored (XGBoost Regressor, Decision Tree Regressor, Random Forest Regressor, Support Vector Regressor) the Decision Tree Regressor seemed to predict the indels that occur from the off-targets more accurately than the other. Additionally, we compared the performance of the cross-validation applied to Decision Tree Regressor with three more generally designed tools(CRISTA,CRISPOR,Elevation) specified in the evaluation of the off-targets. The Decision Tree Regressor with cross-validation performed better than the other tools, and Elevation was the only one that performed comparatively satisfactorily.

Due to the small amount of data available (161 entries from 15 distinct gRNAs) , we cannot apply deep learning techniques because it would most certainly overfit. A further approach would include gathering additional data resulting from new research to develop more complex models. Additionally, off-target cleavage sites should be examined much more thoroughly and rigorously to get information about the genetic features that could assist in a much more efficient approach.

Finally, the Decision Tree Regressor has been integrated into an into an online tool for in-silico evaluation of experiments for DMD therapies. The user inserts the guide RNA they want to evaluate, and subsequently, the algorithm scans the whole human genome and finds potential off-target sequences with up to 4 mismatches (using the FlashFry tool [56]). The possible off-target sequences are evaluated using the Decision Tree Regressor model we developed in this thesis, and an aggregated score for the single guide RNA input is produced, that could be useful for estimating the performance of new CRIPSR-Cas9 experiments for DMD therapies.

Bibliography

[1] Michael Sinnreich and George Karpati. *Dystrophinopathies*, page 205–229. Cambridge University Press, 8 edition, 2010.

[2] Deborah M. Thurtle-Schmidt and Te-Wen Lo. Molecular biology at the cutting edge: A review on crispr/cas9 gene editing for undergraduates. *Biochemistry and Molecular Biology Education*, 46(2):195–205, 2018.

[3] Professor David Hornby. Molecular surgery with crispr-cas9.

[4] Esra Erkut and Toshifumi Yokota. Crispr therapeutics for duchenne muscular dystrophy, 2022.

[5] Guanqing Liu, Yong Zhang, and Tao Zhang. Computational approaches for effective crispr guide rna design and evaluation. *Computational and structural biotechnology journal*, 18:35–44, Nov 2019. 31890142[pmid].

[6] Shiran Abadi, Winston X. Yan, David Amar, and Itay Mayrose. A machine learning approach for predicting crispr-cas9 cleavage efficiencies and patterns underlying its mechanism of action. *PLoS computational biology*, 13(10):e1005807–e1005807, Oct 2017. 29036168[pmid].

[7] Jun Wang, Xiuqing Zhang, Lixin Cheng, and Yonglun Luo. An overview and metanalysis of machine and deep learning-based crispr grna design tools. *RNA Biology*, 17(1):13–22, 2020. PMID: 31533522.

[8] Random forest regression. `https://levelup.gitconnected.com/random-forest-regression-209c0f354c84`. Accessed: 2021-12-1.

[9] Nigel G. Laing, Mark R. Davis, Klair Bayley, Sue Fletcher, and Steve D. Wilton. Molecular diagnosis of duchenne muscular dystrophy: past, present and future in relation to

implementing therapies. *The Clinical biochemist. Reviews*, 32(3):129–134, Aug 2011. 21912442[pmid].

[10] Partha S Ghosh Basil T Darras, David K Urion. *Dystrophinopathies*. GeneReviews(®).

[11] Eppie M Yiu and Andrew J Kornberg. Duchenne muscular dystrophy. *Journal of Paediatrics and Child Health*, 51(8):759–764, 2015.

[12] Kenji Rowel Q. Lim, Chantal Yoon, and Toshifumi Yokota. Applications of crispr/cas9 for the treatment of duchenne muscular dystrophy. *Journal of Personalized Medicine*, 8(4), 2018.

[13] Wei-Jing Dai, Li-Yao Zhu, Zhong-Yi Yan, Yong Xu, Qilong Wang, and Xiao-Jie Lu. Crispr-cas9 for in vivo gene therapy: Promise and hurdles. *Molecular Therapy. Nucleic Acids*, 5, 08 2016.

[14] Jennifer Listgarten, Michael Weinstein, Benjamin P. Kleinstiver, Alexander A. Sousa, J. Keith Joung, Jake Crawford, Kevin Gao, Luong Hoang, Melih Elibol, John G. Doench, and Nicolo Fusi. Prediction of off-target activities for the end-to-end design of crispr guide rnas. *Nature Biomedical Engineering*, 2(1):38–47, Jan 2018.

[15] F. A. Ran, P. D. Hsu, J. Wright, V. Agarwala, D. A. Scott, and F. Zhang. Genome engineering using the CRISPR-Cas9 system. *Nat Protoc*, 8(11):2281–2308, Nov 2013.

[16] Yongchang Chen, Yinghui Zheng, Yu Kang, Weili Yang, Yuyu Niu, Xiangyu Guo, Zhuchi Tu, Chenyang Si, Hong Wang, Ruxiao Xing, Xiuqiong Pu, Shang-Hsun Yang, Shihua Li, Weizhi Ji, and Xiao-Jiang Li. Functional disruption of the dystrophin gene in rhesus monkey using CRISPR/Cas9. *Human Molecular Genetics*, 24(13):3764–3774, 04 2015.

[17] Guanqing Liu, Yong Zhang, and Tao Zhang. Computational approaches for effective crispr guide rna design and evaluation. *Computational and Structural Biotechnology Journal*, 18:35–44, Jan 2020.

[18] Jean-Paul Concordet and Maximilian Haeussler. Crispor: intuitive guide selection for crispr/cas9 genome editing experiments and screens. *Nucleic acids research*, 46(W1):W242–W245, Jul 2018. 29762716[pmid].

[19] John G. Doench, Nicolo Fusi, Meagan Sullender, Mudra Hegde, Emma W. Vaimberg, Katherine F. Donovan, Ian Smith, Zuzana Tothova, Craig Wilen, Robert Orchard, Herbert W. Virgin, Jennifer Listgarten, and David E. Root. Optimized sgrna design to maximize activity and minimize off-target effects of crispr-cas9. *Nature biotechnology*, 34(2):184–191, Feb 2016. 26780180[pmid].

[20] Tom M. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997.

[21] Xian-Da Zhang. *Machine Learning*, pages 223–440. Springer Singapore, Singapore, 2020.

[22] Issam El Naqa and Martin J. Murphy. *What Is Machine Learning?*, pages 3–11. Springer International Publishing, Cham, 2015.

[23] Mariette Awad and Rahul Khanna. *Support Vector Regression*, pages 67–80. Apress, Berkeley, CA, 2015.

[24] Min Xu, Pakorn Watanachaturaporn, Pramod K. Varshney, and Manoj K. Arora. Decision tree regression for soft classification of remote sensing data. *Remote Sensing of Environment*, 97(3):322–336, 2005.

[25] Leo Breiman. *Classification and regression trees*. CRC Press, 1984.

[26] Michael Kearns. 1988.

[27] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 785–794, New York, NY, USA, 2016. Association for Computing Machinery.

[28] Minghua Chen, Qunying Liu, Shuheng Chen, Yicen Liu, Chang-Hua Zhang, and Ruihua Liu. Xgboost-based algorithm interpretation and application on post-fault transient stability status prediction of power system. *IEEE Access*, 7:13149–13158, 2019.

[29] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, Oct 2001.

[30] Andy Liaw and Matthew C. Wiener. Classification and regression by randomforest. 2007.

[31] Rishabh Choudhary and Hemant Kumar Gianey. Comprehensive review on supervised machine learning algorithms. In *2017 International Conference on Machine Learning and Data Science (MLDS)*, pages 37–43, 2017.

[32] Rana singh. Mathematics behind random forest and xgboost, Nov 2019. Accessed: 2021-12-1.

[33] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, Sep 1995.

[34] Hyeran Byun and Seong-Whan Lee. Applications of support vector machines for pattern recognition: A survey. In Seong-Whan Lee and Alessandro Verri, editors, *Pattern Recognition with Support Vector Machines*, pages 213–236, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg.

[35] Hao Yin, Chun-Qing Song, Joseph R. Dorkin, Lihua J. Zhu, Yingxiang Li, Qiongqiong Wu, Angela Park, Junghoon Yang, Sneha Suresh, Aizhan Bizhanova, Ankit Gupta, Mehmet F. Bolukbasi, Stephen Walsh, Roman L. Bogorad, Guangping Gao, Zhiping Weng, Yizhou Dong, Victor Koteliansky, Scot A. Wolfe, Robert Langer, Wen Xue, and Daniel G. Anderson. Therapeutic genome editing by combined viral and non-viral delivery of crispr system components in vivo. *Nature Biotechnology*, 34(3):328–333, Mar 2016.

[36] Christopher E. Nelson, Yaoying Wu, Matthew P. Gemberling, Matthew L. Oliver, Matthew A. Waller, Joel D. Bohning, Jacqueline N. Robinson-Hamm, Karen Bulaklak, Ruth M. Castellanos Rivera, Joel H. Collier, Aravind Asokan, and Charles A. Gersbach. Long-term evaluation of aav-crispr genome editing for duchenne muscular dystrophy. *Nature Medicine*, 25(3):427–432, Mar 2019.

[37] Annalisa Lattanzi, Stephanie Duguez, Arianna Moiani, Araksya Izmiryan, Elena Barbon, Samia Martin, Kamel Mamchaoui, Vincent Mouly, Francesco Bernardi, Fulvio Mavilio, and Matteo Bovolenta. Correction of the exon 2 duplication in dmd myoblasts by a single crispr/cas9 system. *Molecular therapy. Nucleic acids*, 7:11–19, Jun 2017. 28624187[pmid].

[38] Chengzu Long, John R. McAnally, John M. Shelton, Alex A. Mireault, Rhonda Bassel-Duby, and Eric N. Olson. Prevention of muscular dystrophy in mice by crispr/cas9-

mediated editing of germline dna. *Science (New York, N.Y.)*, 345(6201):1184–1188, Sep 2014. 25123483[pmid].

[39] Hongmei Lisa Li, Naoko Fujimoto, Noriko Sasakawa, Saya Shirai, Tokiko Ohkame, Tetsushi Sakuma, Michihiro Tanaka, Naoki Amano, Akira Watanabe, Hidetoshi Sakurai, Takashi Yamamoto, Shinya Yamanaka, and Akitsu Hotta. Precise correction of the dystrophin gene in duchenne muscular dystrophy patient induced pluripotent stem cells by talen and crispr-cas9. *Stem cell reports*, 4(1):143–154, Jan 2015. 25434822[pmid].

[40] Chengzu Long, Leonela Amoasii, Alex A. Mireault, John R. McAnally, Hui Li, Efrain Sanchez-Ortiz, Samadrita Bhattacharyya, John M. Shelton, Rhonda Bassel-Duby, and Eric N. Olson. Postnatal genome editing partially restores dystrophin expression in a mouse model of muscular dystrophy. *Science (New York, N.Y.)*, 351(6271):400–403, Jan 2016. 26721683[pmid].

[41] Christopher E. Nelson, Chady H. Hakim, David G. Ousterout, Pratiksha I. Thakore, Eirik A. Moreb, Ruth M. Castellanos Rivera, Sarina Madhavan, Xiufang Pan, F. Ann Ran, Winston X. Yan, Aravind Asokan, Feng Zhang, Dongsheng Duan, and Charles A. Gersbach. In vivo genome editing improves muscle function in a mouse model of duchenne muscular dystrophy. *Science (New York, N.Y.)*, 351(6271):403–407, Jan 2016. 26721684[pmid].

[42] Hilton Isaac B. (Durham NC) Perez-Pinera Pablo (Lynn MA) Kabadi Ami M. (Durham NC) Thakore Pratiksha I. (Durham NC) Ousterout David G. (Atlanta GA) Black Joshua B. (Durham NC) Gersbach Charles A. (Durham, NC). RNA-GUIDED GENE EDITING AND GENE REGULATION, April 26, 2020.

[43] LONG Chengzu (New York NY) OLSON Eric N. (Dallas, TX). COMPOSITIONS AND METHODS FOR CORRECTING DYSTROPHIN MUTATIONS IN HUMAN CARDIOMYOCYTES, November 26, 2020.

[44] Bengtsson Niclas (Seattle WA) Hauschka Stephen D. (Seattle WA) Chamberlain Jeffrey S. (Seattle, WA). MUSCLE-SPECIFIC CRISPR/CAS9 EDITING OF GENES, December 21, 2017.

[45] LONG Chengzu (Dallas TX) MCANALLY John R. (Dallas TX) SHELTON John M. (Dallas TX) BASSEL-DUBY Rhonda (Dallas TX) OLSON Eric N. (Dallas, TX). PRE-

VENTION OF MUSCULAR DYSTROPHY BY CRISPR/CAS9-MEDIATED GENE EDITING, March 3, 2016.

[46] Kevin L. Howe, Premanand Achuthan, James Allen, Jamie Allen, Jorge Alvarez-Jarreta, M. Ridwan Amode, Irina M. Armean, Andrey G. Azov, Ruth Bennett, Jyothish Bhai, Konstantinos Billis, Sanjay Boddu, Mehrnaz Charkhchi, Carla Cummins, Luca Da Rin Fioretto, Claire Davidson, Kamalkumar Dodiya, Bilal El Houdaigui, Reham Fatima, Astrid Gall, Carlos Garcia Giron, Tiago Grego, Cristina Guijarro-Clarke, Leanne Haggerty, Anmol Hemrom, Thibaut Hourlier, Osagie G. Izuogu, Thomas Juettemann, Vinay Kaikala, Mike Kay, Ilias Lavidas, Tuan Le, Diana Lemos, Jose Gonzalez Martinez, José Carlos Marugán, Thomas Maurel, Aoife C. McMahon, Shamika Mohanan, Benjamin Moore, Matthieu Muffato, Denye N. Oheh, Dimitrios Paraschas, Anne Parker, Andrew Parton, Irina Prosovetskaia, Manoj P. Sakthivel, Ahamed I Abdul Salam, Bianca M. Schmitt, Helen Schuilenburg, Dan Sheppard, Emily Steed, Michal Szpak, Marek Szuba, Kieron Taylor, Anja Thormann, Glen Threadgold, Brandon Walts, Andrea Winterbottom, Marc Chakiachvili, Ameya Chaubal, Nishadi De Silva, Bethany Flint, Adam Frankish, Sarah E. Hunt, Garth R. IIsley, Nick Langridge, Jane E. Loveland, Fergal J. Martin, Jonathan M. Mudge, Joanella Morales, Emily Perry, Magali Ruffier, John Tate, David Thybert, Stephen J. Trevanion, Fiona Cunningham, Andrew D. Yates, Daniel R. Zerbino, and Paul Flicek. Ensembl 2021. *Nucleic Acids Research*, 49(D1):D884–D891, Jan 2021.

[47] R. B. Wallace, J. Shaffer, R. F. Murphy, J. Bonner, T. Hirose, and K. Itakura. Hybridization of synthetic oligodeoxyribonucleotides to phi chi 174 DNA: the effect of single base pair mismatch. *Nucleic Acids Res*, 6(11):3543–3557, Aug 1979.

[48] J. Marmur and P. Doty. Determination of the base composition of deoxyribonucleic acid from its thermal denaturation temperature. *Journal of Molecular Biology*, 5(1):109–118, 1962.

[49] John SantaLucia. A unified view of polymer, dumbbell, and oligonucleotide dna nearest-neighbor thermodynamics. *Proceedings of the National Academy of Sciences*, 95(4):1460–1465, 1998.

[50] Peter Cock. Molecular organisation and assembly in cells. Accessed: 2021-12-1.

[51] *Spearman Rank Correlation Coefficient*, pages 502–505. Springer New York, New York, NY, 2008.

[52] Peter J. A. Cock, Tiago Antao, Jeffrey T. Chang, Brad A. Chapman, Cymon J. Cox, Andrew Dalke, Iddo Friedberg, Thomas Hamelryck, Frank Kauff, Bartek Wilczynski, and Michiel J. L. de Hoon. Biopython: freely available python tools for computational molecular biology and bioinformatics. *Bioinformatics*, 25(11):1422–1423, Jun 2009.

[53] Manuel Stemmer, Thomas Thumberger, Maria del Sol Keyer, Joachim Wittbrodt, and Juan L. Mateo. Correction: Cctop: An intuitive, flexible and reliable crispr/cas9 target prediction tool. *PLOS ONE*, 12(4):e0176619, Apr 2017.

[54] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[55] Sylvain Arlot and Alain Celisse. A survey of cross-validation procedures for model selection. *Statistics Surveys*, 4(none):40–79, Jan 2010.

[56] Aaron McKenna and Jay Shendure. Flashfry: a fast and flexible tool for large-scale crispr target design. *BMC Biology*, 16(1):74, Jul 2018.

APPENDICES

Appendix A

Code

The code which was created for this thesis is available in the following link: https://github.com/elenikou/Off_Target_Evaluation_DMD

```
’’’Calculation of Melting Temperature using the ‘Wallace ’rule,
using empirical formulas based on GC content, using nearest neighbor
thermodynamics and of the GC frequence.’’’

for i in range(len(df)):
    df_tm[’23nt gRNA’][i] = df_tm[’23nt gRNA’][i][:20]

Tm_Wallace=[]
Tm_GC=[]
Tm_NN=[]
GC_content=[]
for i in range(len(df)):
    mystring=df_tm[’23nt gRNA’][i]
    myseq = Seq(mystring)
    Tm_Wallace.append(mt.Tm_Wallace(myseq))
    Tm_GC.append(mt.Tm_GC(myseq))
    Tm_NN.append(mt.Tm_NN(myseq))
    GC_content.append(getGCFreq(df_tm[’23nt gRNA’][i]))

df_tm[’Tm_Wallace’]=Tm_Wallace
```

```
df_tm['Tm_GC']=Tm_GC
df_tm['Tm_NN']=Tm_NN
df_tm['GC_content']=GC_content

#Spearman Scorer
def Spearman(X, Y):
    Spearman, p_value = stats.spearmanr(X, Y)
    return Spearman

spearman = make_scorer(Spearman)

#Nested Cross−Validation

#Grid Creation for Decision Tree Regressor
max_depth = [int(x) for x in np.linspace(10, 110,
            num = 11)]
max_depth.append(None)
criterion = spearman
min_samples_split = [2, 5, 10]
min_samples_leaf = [1, 2, 4]
max_features = ['auto', 'sqrt']
grid = {'max_features': max_features,
        'max_depth': max_depth,
        'min_samples_split': min_samples_split,
        'min_samples_leaf': min_samples_leaf}
print(grid)



#Inner cross validation
grid_cross_validation = KFold(n_splits=3, shuffle=True,
                              random_state=1)
#model
dtr = DecisionTreeRegressor(random_state=1)
```

```
#Grid search for the hyperparameters grid
#using the inner cross validation
grid_search = GridSearchCV(dtr, grid, scoring=spearman, n_jobs=1,
                           cv=grid_cross_validation, refit=True)
#Outer cross validation
final_cross_validation = KFold(n_splits=10, shuffle=True,
                               random_state=1)
#Nested cross validation search
scores = cross_val_score(grid_search, df, target, scoring=spearman,
                         cv=final_cross_validation, n_jobs=−1)
#Average Spearman Score
print(’Average Spearman Score: %.3f (%.3f)’ %
        (mean(scores), std(scores)))
```