Machine Learning-Driven Prediction of Fluorescent Probe Properties: Bridging the Gap between Prediction and Experimentation

by

Aisana Bolatbek

Submitted to the Department of Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Data Science

at the

NAZARBAYEV UNIVERSITY

April 2024

Author .....
Department of Computer Science
April 2024

Certified by .....
Siamac Fazli
Associate Professor
Thesis Supervisor

Certified by .....
Vsevolod Peshskov
Assistant Professor
Thesis Supervisor

Accepted by .....
Elizabeth Arkhangelsky
Acting Dean, School of Engineering and Digital Sciences

Machine Learning-Driven Prediction of Fluorescent Probe Properties: Bridging the Gap between Prediction and Experimentation

by

Aisana Bolatbek

Submitted to the Department of Computer Science
on April 2024, in partial fulfillment of the
requirements for the degree of
Master of Science in Data Science

Abstract

The development of organic fluorescent materials needs quick and precise predictions of photophysical characteristics for techniques like high-throughput virtual screening. However, there is a challenge caused by the constraints of quantum mechanical computations, experiments, and time. This thesis investigates the field of machine-learning-assisted fluorescence probe design to answer this difficulty. The main part of this investigation is the utilization of a substantial database of optical properties of organic compounds that was collected from various scientific papers. One of the complicating factors of this database is the presence of missing data which stems from the collection from various sources, and this inconsistency is examined with the use of a range of imputation methods. Furthermore, the thesis aims to construct predictive models that can forecast properties that are inherent to fluorescent compounds such as quantum yield, absorption and emission spectra, among others. This research aims to pave the way for a more efficient and targeted approach to fluorescent probe design.

Thesis Supervisor: Siamac Fazli
Title: Associate Professor

Thesis Supervisor: Vsevolod Peshskov
Title: Assistant Professor

Acknowledgments

The completion of this thesis is the culmination of efforts from various individuals whom I would like to express my sincere appreciation.

Firstly, I would like to express my appreciation to all who have directly made this research possible. I would like to express my sincere gratitude to my thesis supervisor Siamac Fazli, who has guided and encouraged me through this challenging yet rewarding journey. Siamac's mentorship has been invaluable, providing not only support but also inspiring new ideas and perspectives which enriched the depth and breadth of this research. Thank you, for your patience and belief in my abilities. Moving forward, I would like to share my heartfelt thanks to my colleagues, Miras Nurkin, Dias Kuatbekov, Kuanysh Akhmetzhanov, who worked alongside me on this project. I want to take this opportunity to wish each of you all the best. Last, but certainly not least, I would like to thank Vsevolod Peshkov, for his expertise in this field and guidance throughout the path of this journey.

I would like to thank my family and friends, who have supported and encouraged me. Thank you, mom, for always being there for me. Thank you, my dear friends and my brother, for being my pillars of strength. I would like to extend my heartfelt appreciation to my groupmates from Data Science, Class of 2024, with whom I have shared this remarkable journey during our master's program. Tomiris Zhaksylyk, Gaukhar Zhunussova, Batyr Arystanbekov, Zhansaya Maksut, Azamat Shora, and all, your companionship and support made this journey truly memorable. May you continue to follow your passions, seize opportunities, and achieve success in all your endeavors.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Fluorescence is a phenomenon when certain atoms and molecules absorb light at a specific wavelength and emit at a longer wavelength after a short period, known as the fluorescence lifetime [1]. The fluorescence process is influenced by three crucial events as shown in Figure 1-1. The initial event, where a molecule is excited by an incoming photon, takes place in femtoseconds (10^{-15} seconds). Subsequently, the vibrational relaxation of excited state electrons to the lowest energy level occurs at a slower pace, measured in picoseconds (10^{-12} seconds). The final step involves the emission of a longer wavelength photon, returning the molecule to the ground state, and unfolds over a longer time frame of nanoseconds (10^{-9} seconds).

Although the entire molecular fluorescence lifetime is measured in a mere billionth of a second, it is a remarkable interaction between light and matter. Fluorescent probes are essential tools in the fields of molecular biology, pharmacology, and cellular imaging. Appropriately constructed molecules emit fluorescent signals when bound to specific cellular or molecular targets. This fluorescence can be detected and used to study cellular processes, identify specific cellular structures, or monitor various biochemical reactions. Originally used to monitor protein dynamics, recent advances in fluorescent probes allow sophisticated measurements of protein instability and turnover. Moreover, fluorescent sensors are shedding light on the "dark matter" of the cellular milieu, visualizing small molecules, secondary metabolites, metals, and ions at the single-cell level [2].

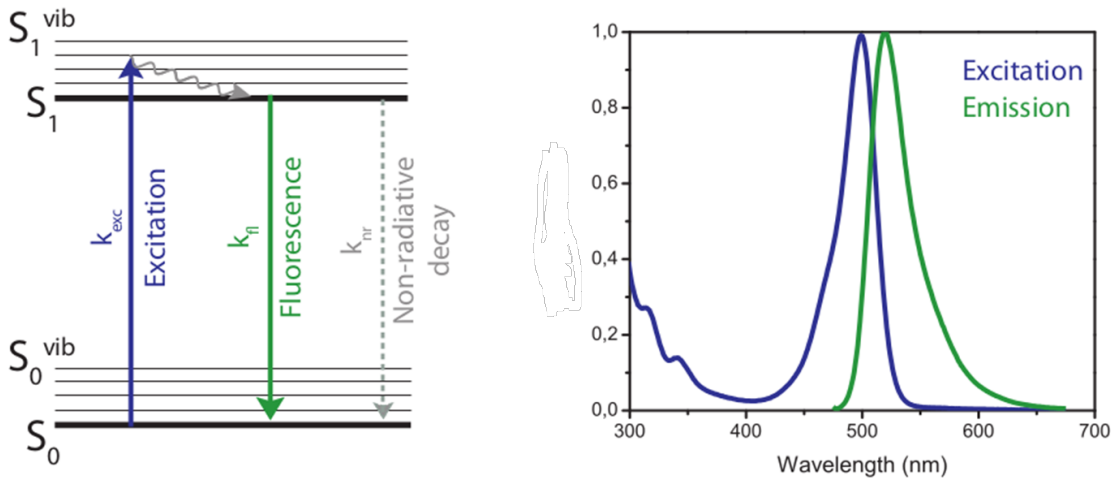

Figure 1-1: The diagram illustrating energy state transitions leading to fluorescence and excitation and emission profile

However, the accurate engineering of these probes is a difficult and time-consuming procedure that requires numerous trials of experiments. Theoretical calculations based on *ab initio* and density functional theory methods have been widely used to compute optical properties of designed chromophores (i.e. molecules that absorb light and emit colour as a result). Such theoretical calculations require high computational costs. Therefore, the development of advanced materials has entered a new era with the introduction of machine learning (ML) and artificial intelligence (AI), which promise to accelerate the discovery of novel compounds and simplify the design process. As a result of its strong data processing capability and relatively low research threshold, ML can significantly cut human and material expenses and expedite the research and development cycle. It is used to examine material structures and forecast material properties and has the potential to replace conventional research [3, 4]. In the context of predicting properties of fluorescent probes, the latest literature highlights notable progress in this area, indicating that the research goal is achievable. There are some findings on predicting properties such as emission wavelength or quantum yields with the use of ML approaches [4, 5, 6, 7, 8].

There are key prerequisites for an ML-based approach, which are data processing and feature engineering. Data-driven solutions like ML are determined by high-quality

data, by its reliability and extent. Another challenging task, which is essential for this specific topic, is the representation of chemical data, in a way, that is understandable by both humans and machines [9]. Different ways of representations, which are linear, structural, graphical, or fingerprints are reviewed in these papers [9, 10]. Molecular descriptors are information about molecule physicochemical characteristics, such as those related to constitution, structure, lipophilicity, electronics, geometry, hydrophobicity, solubility, quantum chemistry, and topology. Fingerprints are a binary (Yes/No), or count descriptors that show which functional groups, molecular topology, and physical properties, are present in the molecules. Thanks to the advancements in computing power, chemists may now evaluate the chemical space by utilizing fingerprints and large-scale molecular descriptors. However, rational feature selection is often expensive and difficult.

In the pursuit of unraveling the relationship between ML and the synthesis of fluorescent probes, this thesis utilizes comprehensive databases [6, 11]. The databases are a collection of of experimental data including various optical properties of fluorescent compounds. The primary focus lies in investigating the potential of AI to forecast the optical properties inherent in organic molecules. This research aims to illuminate how predictive modeling can be harnessed to anticipate the unique optical signatures exhibited by fluorescent compounds.

Beyond mere exploration, this thesis ventures to capture diverse molecular representations and generate novel features, thereby asses the influence of feature engineering within the context of predicting optical properties. Furthermore, an additional objective of this work is to extend the application of predictive modeling to address the issue of missing values within the existing databases. By leveraging ML techniques for imputation, the research aims to enhance the completeness and accuracy of the database, ultimately fostering a more robust foundation for the design and prediction of fluorescent probes. This thesis aims to contribute to the integration of AI and molecular design, advancing the frontier of fluorescent probe development for novel applications in biological research.

Finally, the research objectives consist of the following parts:

- Harness predictive capabilities of ML to utilize a database of organic compounds;
- Leverage molecular representations and feature extraction to enhance the usage of the database;
- Address another issue of the dataset – the presence of missing data by imputation.

Chapter 2

Related work

Fluorescent compounds are frequently used in a variety of settings. Given their photochemical properties, fluorescent molecules can be utilized as analytical and diagnostic instruments to study biological science and comprehend cell biology. Fluorescent compounds have been utilized to designate target cells, RNAs, DNAs, peptides, and live-cell pictures, as demonstrated by recent scientific advancements [11]. For a long time, chemists have been searching for the fluorescent core structures. Current research strategies rely primarily on scientific intuition and trial-and-error experimentation. In recent years, ML has shown great potential as a useful tool in many areas including material chemistry.

In [4], the authors investigate the relationship between the chemical structure of fluorescent dyes and their live-cell imaging properties. In this study, authors synthesize 1536 dyes to model ML-classifiers for assisting live-cell staining and endoplasmic reticulum judgment. They have generated more than 2000 molecular representations and reduced dimensions with Principal Component Analysis (PCA). However, the best performance is obtained from the model trained with features without PCA. Then, a multi-class classification task on cell-staining ability is performed. Moreover, they have examined another binary classification problem of whether dyes can target ER for imaging based on a subset of the initial dataset. Utilized ML models include K-Nearest Neighbours (KNN), Logistic Regression, Random Forest (RF), Gradient Boosting (GB), and Multilayer Perceptron (MLP). They achieved an accuracy of 84%,

a recall of 89% with an Area Under Curve of 0.9 with the GB model.

In [5], the authors discuss the significance of basic photophysical parameters by leveraging the large-scale (nearly 12,000 molecules) database to apply ML for emission wavelength predictions. They have introduced clustering and statistical approaches for predictive modeling. Firstly, they have extracted descriptors and reduced the descriptor dimension with several statistical indicators like variance threshold selection. Moreover, they have explored K-means to cluster 15 subgroups of molecular representations. It reduced the dimensionality of the feature space from 11411 to 6208. The Least Absolute Shrinkage and Selection Operator (Lasso) regression coupling in the ensemble with the RF model is reported as the best predictor. Lasso is used to extract dominant 480 descriptors. After all manipulations, they achieved $R^2 = 0.655$ with lower computational expenses. Finally, it is identified that four conjugated π - bonding related descriptors dominantly contribute to the target value.

In [6], authors establish a database covering more than 4300 solvated organic fluorescent dyes with 3000 distinct compounds and develop an ML approach to forecast emission and absorption wavelengths and photoluminescence quantum yield. They executed common procedure, where the first step includes descriptor generation to later use it as feature space. They have generated different descriptors like Morgan fingerprints, CDK fingerprints, MACCS keys, PubChem fingerprints, and their combinations with functionalized structure descriptors for chromophores, and extracted general experimental solvent descriptors. They have utilized different ML algorithms like MLP, Support Vector Machines (SVM), KNN, and various tree models. Along with regression models, the authors have built classification models to predict quantum yield. The best model achieved a Mean Absolute Error of 0.13, 11.1 and 14.6 for quantum yield, absorption and emission wavelengths, respectively. Authors have compared their estimates with time-dependent density functional theory calculations, and found out that few data augmentation improved their ML predictions. However, their suggestion of data augmentation involves adding similar to molecules to test set molecules, which contradicts ML rules. Particularly, this is called data leakage, phenomenon which occurs when information from the test set, which the model should

not have access to during training, is inadvertently included in the training process. This can lead to overly optimistic performance estimates during model training and evaluation because the model is effectively being trained on information that it will later need to predict.

In [7], authors predict photophysical properties, such as quantum yield, emission wavelength, and radioactive decay rate constant, of phosphorescent emitters with ML models. Phosphorescence is a process similar to fluorescence, the difference is that it involves a longer-lived excited state. Phosphorescent materials continue to emit light after the light source is removed, which is why they are often observed glowing in the dark. To implement the ML approach, the authors have established a dataset with 200 samples collected from the literature. As features, they have extracted up to 15 descriptors. This is rather a small dataset to adequately assess the validity of results from the prespective of ML. Nevertheless, they have implemented ML models like KNN, SVM, RF, Boosted Trees like LightGBM, AdaBoost, and XGBoost, with 10-fold cross-validation. The best results of the coefficient of determination R^2 are 0.96, 0.81, and 0.67 for the predictions of emission wavelength, photoluminescence quantum yield, and radioactive decay rate constant, respectively.

In [8], the authors predict the quantum yield of carbon quantum dots in biochar produced from 10 types of farm waste. Carbon quantum dots are a type of carbon-based nanomaterial with sizes typically ranging from a few to several tens of nanometers. The authors investigated the relationship between biochar preparation parameters (12 experimental descriptors) and quantum yield with prepared 480 samples. Their topic of interest is very specific with a focus on one type of nanomaterial, and the dataset is limited, thus, it is not generalizable to other cases. Their experiment involves 6 ML models. Models include KNN, tree-based models like Decision Tree (DT), RF, etc. The best result corresponds to the GB regression model with $R^2 > 0.9$. Moreover, they investigated feature importance of the best model.

The main course of this work is to predict the optical properties of fluorescent molecules based on available structural information. Tables 2.1 and 2.2 illustrate a comparison of this work and the above-stated related papers. Table 2.1 demonstrates

target values addressed in the related papers and in this thesis. Table 2.2 shows information regarding the approach and methodology.

The work covered in this thesis follows a similar procedure as in the above-stated papers. The datasets [6, 11] used in the work provide more information on optical properties, particularly, it is possible to predict emission wavelength λ_{emi}, absorption wavelength λ_{abs}, quantum yield Φ_{QY}, extinction coefficient $\log_{10}(\epsilon_{max})$, fluorescence lifetime τ_{flu}. The datasets are utilized with feature extraction techniques,i.e. descriptor generation with the help of sufficient libraries through molecular structure. One difference of this thesis is an attempt to further data augmentation through imputation. Ideally, we want to build a predictive model to forecast all available optical properties. However, the existence of missing values in the dataset presents another challenge. Thus, this thesis work focuses on another feature engineering approach, which is imputation. Regression models are utilized, and predictions are evaluated to assess not only model performance but also to validate data augmentation approaches.

Table 2.1: Review of related papers and contribution of this work

Work	**Target variables**					
	Maximum absorption wavelength, λ_{abs}	Maximum emission wavelength , λ_{emi}	Quantum yield, Φ_{QY}	Cell staining ability	Lifetime, τ_{flu}	Extinction coefficient, $\log_{10}(\epsilon_{max})$
Yang et al. [4]				✓		
Ye et al. [5]		✓				
Ju et al. [6]	✓	✓	✓			
Wang et al. [7]		✓	✓			
Chen et al.[8]			✓			
Thesis	✓	✓	✓		✓	✓

Table 2.2: Review of related papers and contribution of this work (continued)

Work	**Data size**	**Feature engineering**	**Additional processing**	**ML task**	**Year**
Yang et al. [4]	1536	Descriptor generation	Dimensionality reduction	Classification	2023
Ye et al. [5]	11460	Descriptor generation through SMILES	Dimensionality reduction	Regression	2020
Ju et al. [6]	4300	Descriptor generation through SMILES	✗	Regression, Classification	2021
Wang et al. [7]	206	Descriptor computation through calculations based on density functional theory	✗	Regression	2023
Chen et al. [8]	480	Descriptor generation through experiments	✗	Regression	2023
Thesis	22907	Descriptor generation through SMILES	Imputation	Regression	2024

Chapter 3

Methodology

This section provides the overall methodology of this study. This methodology is divided into the following sections. The Dataset Collection and Preprocessing section holds detailed information on the dataset and its preparation part. The Data Augmentation approaches section explains the resampling procedure. The Feature Engineering section reports one of the main steps in modeling predictors. It explains various representations of molecules. The Predictive Modelling part dives into the ML part. It holds information on the problem description, selected models, and evaluation metrics. Finally, the Research workflow section concludes the research path.

3.1 Dataset Collection and Preprocessing

In this thesis work, a dataset [11] with optical properties of organic compounds is being utilized. To enhance model performance, we decided to include an additional dataset [6]. For simplicity, we call the first dataset *Experimental*, and the second one *Chemfluor*.

After data processing, we focused on five luminescence-related properties as targets. Table 3.1 shows the chosen fluorescent properties and their definitions.

Table 3.1: Target values

Property	Definition
Maximum absorption wavelength	The wavelength of light at which a fluorescent substance absorbs the highest intensity of electromagnetic radiation
Maximum emission wavelength	The wavelength of light at which a fluorescent substance emits the highest intensity of electromagnetic radiation
Quantum Yield	The ratio of the number of photons emitted to the number of photons absorbed
Extinction Coefficient	How strongly a fluorescent molecule absorbs light at a particular wavelength
Lifetime	The time a fluorophore spends in the excited State before emitting a photon and Returning to the ground State

3.1.1 Data Collection

Table 3.2 shows information on Experimental dataset attributes. This dataset is collected from different sources and contains different attributes as well as target variables such as absorption wavelengths, emission wavelengths, quantum yields, etc. The molecules are represented in the notation of a simplified molecular-input line-entry system (SMILES). Since the dataset was collected from different sources, it has an issue with missing data, which will be addressed in this thesis. Table 3.2 also shows information an available values for each column.

Table 3.3 shows information on the additional Chemfluor dataset. Similarly, it contains information about different properties. It was collected from different sources, thus, some missing values are also present.

3.1.2 Data Preprocessing

There are some cleaning steps applied to assemble a new dataset.

Two datasets, Experimental and ChemFluor datasets are combined into one. Both of them are collected from different sources, thus, there are missing values, and also duplicates, i.e. the same combinations of chromophores and solvents were presented in different papers by different authors. In order to keep only unique combinations,

Table 3.2: Experimental Dataset overview

Column name	**Data type**	**Description**	**Non-missing values**
Tag	Float	The numbering of data points	20236
Chromophore	String	SMILES of chromophore structure	20236
Solvent	String	SMILES of solvent structure	20236
Absorption max (nm)	Float	Maximum absorption wavelength, $\lambda_{abs,max}$	17295
Emission max (nm)	Float	Maximum emission wavelength, $\lambda_{emi,max}$	18142
Lifetime (ns)	Float	Fluorescence lifetime, τ_{flu}	6960
Quantum yield	Float	Photoluminescence quantum yield, Φ_{QY}	13837
$\log(e/mol^{-1} dm^3 cm^{-1})$	Float	Extinction coefficient at $\lambda_{abs,max}$, $\log_{10}(\epsilon_{max})$	8041
abs FWHM (cm^{-1})	Float	Absorption bandwidth (FWHM), σ_{abs}	747
emi FWHM (cm^{-1})	Float	Emission bandwidth (FWHM), σ_{emi}	627
abs FWHM (nm)	Float	Absorption bandwidth (FWHM), σ_{abs}	3592
emi FWHM (nm)	Float	Emission bandwidth (FWHM), σ_{emi}	7198
Molecular weight ($gmol^{-1}$)	Float	Molecular weight of chromophore	20236
Reference	String	Source document DOI	20236

Table 3.3: Chemfluor Dataset overview

Column name	Data type	Non-missing values
Absorption/nm	Float	4252
Emission/nm	Float	4386
PLQY	Float	3090
SMILES	String	4386
solvent	String	4386
Reference(doi)	String	4386
Et30	Float	4386
SP	Float	4386
SdP	Float	4386
SA	Float	4386
SB	Float	4386
Test method of Quantum Yield	Float	3207

aggregated values for different properties are stored.

Another issue is that not all data points correspond to solutions of fluorophores. Some compounds are fluorescent in solid state or thin films, meaning that no solvent is involved. Some compounds could be emissive in their specific states, such as solid, liquid, or gaseous. For the sake of uniformity, it is decided to drop cases where there is no combination of chromophore and solvent, rather just chromophore alone.

In addition, we are interested in keeping only canonical representations of molecules, therefore, SMILES notations are canonicalized.

In both datasets, more than five properties are presented. However, the available number is not sufficient for ML problems, thus, they are not considered in this work.

After all above-mentioned transformations, the dataset is formed, and Table 3.4 shows the information on the available size for each target property.

Table 3.4: Dataset after data cleaning

Column name	Non-missing values
Chromophore	22907
Solvent	22907
Absorption max (nm)	20471
Emission max (nm)	20924
Lifetime (ns)	6703
Quantum yield	15836
$\log(e/mol^{-1}\ dm^3 cm^{-1})$	7919

3.2 Data Augmentation approaches

We want to restore one value with another one. The issue is that most of the property columns consist of missing data, moreover, there is a correlation between some target values (see Figure 3-1). A high correlation coefficient means that there is a linear dependency between values. So there is an opportunity to perform data augmentation based on imputation with regression (maybe linear regression). Data augmentation is the process of artificially generating new data from existing data. It is one to boost model performance. Therefore, we want to analyze whether data augmentation affects predictions. There are different approaches to restoring the data for our case.

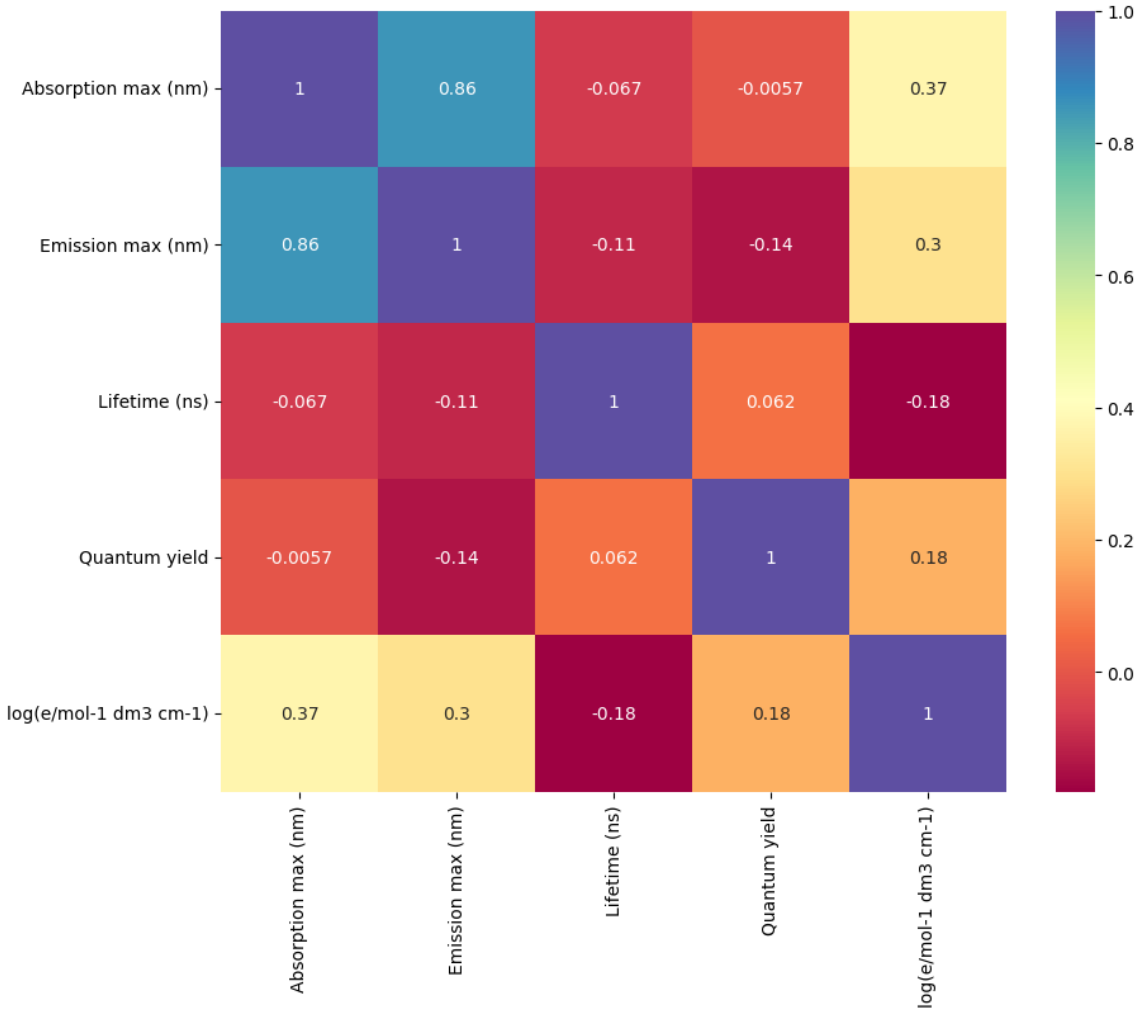

Figure 3-1: Correlation of target properties

3.2.1 Imputation approach

To analyze the effects of imputation let us consider the following task: predict one of the target variables (properties) based on the given information, i.e. other properties and molecular representations. To do so we should impute data, i.e. we will restore missing values in target columns. Fewer missing values mean better accuracy or reliability of the data, so consider columns with less missing data. Algorithm 1 describes two variants to impute data.

Algorithm 1 Data Preprocessing Steps

1: **Feature Imputation**:
2: - Delete rows where no label is available
3: - Split dataset: train (70%) - validation (15%) - test (15%)
4: - Impute missing values with some imputation model
5: - Train regression models on train dataset, use a validation set to choose hyperparameters
6: **Target Imputation**:
7: - Do not delete any rows
8: - Keep validation and test the same as in the previous set (meaning we won't have any imputed target values in evaluation sets)
9: - Add rows to the train dataset, where there are missing values for target variable
10: - Impute missing values with some imputation model
11: - Train regression models on new train dataset, use a validation set to choose hyperparameters
12: Compare the results of two approaches on the test set

By comparing the results of the two approaches and results, where no imputation is applied, we can analyze the effect of imputation on a particular property as well as the performance of ML models.

3.2.2 Imputation model

Imputation models considered in this work include **Multivariate imputation** [12] and **Nearest neighbors imputation** [13] algorithms. This section describes the first approach. The latter approach uses the basic K- Nearest Neighbours algorithm, which is described in detail in section 3.4.6.

To impute models target properties are used, also, some common descriptors are added. These include descriptors such as number of atoms, number of heavy atoms, number of hydrogen donors and acceptors, TPSA, Chi-indexes, Kappa-indexes. Descriptors are explained in detail in section 3.3.

Multivariate imputation algorithms

Multivariate imputation algorithms use the entire set of available feature dimensions to estimate the missing values. This approach is performed with the Iterative Imputer. The Iterative Imputer class, which approximates the missing values within each feature by leveraging the information contained in other features, operates through an iterative process. In each iteration, a particular feature column is designated as the dependent variable, denoted as y, while the remaining feature columns are treated as independent variables, represented collectively as X. Subsequently, a regression model is fitted on the available observed data (X, y) to estimate the missing values of y. This procedure is executed iteratively for each feature, following a round-robin fashion, and is repeated for a predetermined number of imputation rounds. Mathematically, the iterative imputation procedure can be represented as:

$$\hat{y}^{(t)} = \hat{f}_t(X_{\text{observed}}, y_{\text{observed}}),$$

where $\hat{y}^{(t)}$ denotes the predicted values of the feature y at iteration t, $\hat{f}_t$ represents the regression model trained at iteration t, and $(X_{\text{observed}}, y_{\text{observed}})$ denotes the subset of the data comprising observed values of both X and y.

Hyperparameters

There are different strategies to handle missing values. In this work mean of each column is used to replace missing values. Since we have missing values in different columns, we can define the order in which the features will be imputed, for example, in *ascending* order, from features with the fewest missing values to the most, or in *descending*. Moreover, the imputation model uses *estimators* at each step of the

round-robin imputation. Since a high correlation between target values is the motive to impute missing values, some linear regression algorithms, like Lasso and Ridge, are applied. Tree models tend to keep similar values and do not produce out-of-range values, thus, DT and boosted tree (XGBoost), are also considered. The first three models are described in sections 3.4.2, 3.4.3.

XGBoost (eXtreme GB) [14] is based on the principle of GB (see 3.4.5). XGBoost extends traditional GB by incorporating additional regularization terms in the objective function. The objective function of XGBoost is a sum of the loss function L and regularization terms Ω, which control the complexity of the model and help prevent overfitting:

$$\text{Objective} = \sum_{i=1}^{n} L(y_i, F_{m-1}(x_i) + h(x_i)) + \sum_{k=1}^{K} \Omega(f_k)$$

The regularization terms (Ω) can include penalties on the complexity of individual trees, such as L1 or L2 regularization on leaf weights, as well as penalties on the number of leaves or the depth of the trees.

XGBoost uses a different approach called "leaf-wise" tree growth. In leaf-wise growth, XGBoost grows the tree node-by-node, expanding the node that provides the maximum reduction in the objective function.

Unlike traditional GB algorithms, XGBoost leverages parallel processing techniques.

Evaluation

Evaluating imputation results is a challenging task. There are different criteria to assess imputation performance. In this work, **the sum of relative errors** is chosen as evaluation criterion.

Relative error is typically calculated as the absolute difference between the true value and the predicted value, divided by the true value. Mathematically, for a single

data point, the relative error (RE) is calculated as:

$$RE = \frac{|true\_value - predicted\_value|}{true\_value}$$

The sum of relative errors is then computed across all data points.

To ensure the robustness of the selected imputation model, a cross-validation procedure should be performed. In this work, the cross-validation involves randomly removing different amounts of data from the original dataset multiple times (e.g., removing 10, 100, 1000, and 10000 data points). For each iteration of the cross-validation, imputation is performed on the modified datasets using the chosen imputation model. This entire process is repeated multiple times (5) to obtain a robust estimate of the imputation model's performance across different subsets of the data. The imputation model that consistently yields the lowest sum of relative errors across the cross-validation iterations is selected as the best model for handling missing values in the dataset. In other words, at each iteration models are ranked by their evaluation results, and multi-ranking is performed with Order-weighted Average [15]. In this method, each ranking is assigned a weight. In our case, weights are determined by the degree of 10 (i.e., 1 for 10 data points removed, 2 for 100 data points removed, etc.). Then, the rankings are combined by taking a weighted average of the ranks for each item.

3.3 Feature Engineering

A key part of the data preparation phase in ML is a feature engineering. During feature engineering, the features are extracted from the raw data. The selection of features is critical for building ML models and could affect the overall performance.

3.3.1 Molecular representation

In the case of chemistry-related tasks, there are various ways to define chemical compounds by machine-readable molecular representations. Representing chemical

data in an unambiguous ways, understandable both by humans and computers, is a challenging task. No representation is perfect for every circumstance, it depends on many factors as well as space constraints. This section provides some knowledge on available molecular representations used to build ML models.

The data is provided with SMILES notations both for chromophore and solvent compounds. SMILES, a Simplified molecular-input line-entry system represents organic molecules with a string of ASCII characters. String representations can be treated as words, and concepts from natural language processing can be applied to solve chemistry-related problems. This work does not dive into this field, rather the numerical data from molecular formula, SMILES, is extracted as in the traditional way of ML.

3.3.2 Molecular descriptors

One can represent chemical data through *molecular descriptors*. Molecular descriptors are numerical representations derived from the structural characteristics of molecules [9, 10]. These descriptors encompass a diverse array of properties, including constitutional, topological, geometrical, and electronic features. Careful selection of descriptors is crucial to balance informativeness with computational efficiency and model interpretability.

Molecular descriptors can be divided into two types: experimental and theoretical descriptors.

Experimental descriptors

Experimental descriptors are physical properties obtained by experimental observations or numerical simulations, such as $logP$, dipole moment, and, in general, additive physicochemical properties.

Theoretical descriptors

Theoretical representations, on the other hand, are generated computationally based on theoretical models and simulations. These representations leverage principles from quantum mechanics, molecular mechanics, or statistical mechanics to predict molecular structures and properties. Theoretical representations can be further categorized based on their dimensionality.

One-dimensional representations encapsulate molecular structures along a single axis, typically representing sequences of atoms or chemical bonds. Examples include SMILES (Simplified Molecular Input Line Entry System) notations, a count of different functional groups. 1D representations provide a compact and portable format for representing molecular structures but may lack spatial information.

Two-dimensional representations capture the planar connectivity of atoms within a molecule, preserving information about bond types, angles, and functional groups. Notable examples include molecular graphs, adjacency matrices, and molecular fingerprints such as Extended Connectivity Fingerprints (ECFP) and MACCS keys. 2D representations are more complex as well as still interpretable.

Three-dimensional representations encode the full spatial arrangement of atoms in three-dimensional space, accounting for bond angles, torsional angles, and interatomic distances. Molecular conformations, molecular mechanics force fields, and molecular docking poses are examples of 3D representations commonly used in computational chemistry. While 3D representations provide the most detailed insight into molecular structure, they are computationally expensive to generate and may require specialized modeling techniques.

3.3.3 Feature extraction

In this study, the **RDKit** library, and **PaDEL-Descriptor** software were employed to compute a diverse set of molecular descriptors from the molecular representations. RDKit, a popular open-source cheminformatics toolkit, provides a comprehensive suite of functions for molecular manipulation and descriptor calculation. PaDEL cal-

culates different kinds of molecular descriptors and fingerprints, using the Chemistry Development Kit (CDK). The total size of feature space is 13206 descriptors (6603 for each of chromophore and solvent in combination).

Overview of the extracted descriptors

This work leverages 1D and 2D descriptors and fingerprints. Generally speaking, the term "fingerprint" implies that the following representation is in the form of a numerical vector. If it is not indicated as a fingerprint or any other representation, then the descriptor means that it is a single numerical value. Next, overview of single descriptors is provided. Table 3.5 captures information about extracted descriptors from RDKit, their dimensionality, and brief definitions. Table 3.6 captures information about another set of fragmental (substructures in the graph) descriptors extracted in this work from RDKit [16]. Total number of descriptors is 189.

Table 3.5: Descriptors from RDKit

Descriptor	**Type**	**Definition**	**Paper**
MolWt	1D	Molecular Weight	
NumAtoms	1D	Number of Atoms	
NumHeavyAtoms	1D	Number of Heavy Atoms	
NumHeteroatoms	1D	Number of Heteroatoms	
NumAliphaticCarbocycles	1D	Number of AliphaticCarbocycles	
NumAliphaticHeterocycles	1D	Number of AliphaticHeterocycles	
NumAliphaticRings	1D	Number of Aliphatic Rings	
NumAmideBonds	1D	Number of Amide Bonds	
NumSpiroAtoms	1D	Number of SpiroAtoms	
NumAromaticCarbocycles	1D	Number of Aromatic Carbocycles	
NumAromaticHeterocycles	1D	Number of Aromatic Heterocycles	
NumAromaticRings	1D	Number of Aromatic Rings	
NumHeterocycles	1D	Number of Heterocycles	
NumRings	1D	Number of Rings	
NumRotatableBonds	1D	Number of Rotatable Bonds	
NumSaturatedCarbocycles	1D	Number of Saturated Carbocycles	
NumSaturatedHeterocycles	1D	Number of Saturated Heterocycles	
NumSaturatedRings	1D	Number of Saturated Rings	
NumHBA,	1D	Number of Hydrogen Bond Acceptors	

Descriptor	Type	Definition	Paper
NumLipinskiHBA	1D	Number of Hydrogen Bond Acceptors calculated according to Lipinski's Rule of Five	
NumHBD	1D	Number of Hydrogen Bond Donors	
NumLipinskiHBD	1D	Number of Hydrogen Bond Donors calculated according to Lipinski's Rule of Five	
Chi0v, Chi1v, Chi2v, Chi3v, Chi4v, Chi0n, Chi1n,Chi2n, Chi3n, Chi4n, HallKierAlpha , Kappa1, Kappa2, Kappa3	2D	Molecular Connectivity Chi Indexes and Kappa Shape Indexes	[17]
BalabanJ	2D	Balaban J index, Topological index, the degree of branching and cyclicity	[18]
BertzCT	2D	Bertz Chemical Topology index, the complexity or structural diversity of molecules	
IPC	2D	The information content of the coefficients of the characteristic polynomial of the adjacency matrix of a hydrogen-suppressed graph of a molecule	[19]
MolLogP, MolMR	2D	Wildman-Crippen logP and MR value	[20]
FpDensityMorgan	2D	Morgan fingerprint density	[21]
LabuteASA	1D	Labute's Approximate Surface Area	[22]
PEOE-VSA1 – PEOE-VSA14	2D	MOE Charge VSA Descriptor. The approximate accessible van der Waals surface area calculation for each atom along with the contribution to partial charge	[22]

Descriptor	Type	Definition	Paper
SlogP-VSA1 – SlogP-VSA12	2D	MOE logP VSA Descriptor. The approximate accessible van der Waals surface area calculation for each atom along with the contribution to the Log of the octanol/water partition coefficient	[22]
SMR-VSA1 – SMR-VSA10	2D	MOE SMR VSA Descriptor. The approximate accessible van der Waals surface area calculation for each atom along with the contribution to Molar Refractivity	[22]
TPSA	1D	Topological polar surface area	[23]
EState-VSA1 – EState-VSA11	2D	EState VSA Descriptor. MOE-type descriptors using electrotopological state indices and surface area contributions	
VSA-EState1 – VSA-EState10	2D	VSA EState Descriptor	

Table 3.6: Fragmental descriptors from RDKit

Fragment	Definition
fr-Al-COO	Number of aliphatic carboxylic acids
fr-Al-OH	Number of aliphatic hydroxyl groups
fr-Al-OH-noTert	Number of aliphatic hydroxyl groups excluding tert-OH
fr-ArN	Number of N functional groups attached to aromatics
fr-Ar-COO	Number of Aromatic carboxylic acids
fr-Ar-N	Number of aromatic nitrogens
fr-Ar-NH	Number of aromatic amines
fr-Ar-OH	Number of aromatic hydroxyl groups
fr-COO	Number of carboxylic acids
fr-COO2	Number of carboxylic acids
fr-C-O	Number of carbonyl
fr-C-O-noCOO	Number of carbonyl O, excluding COOH
fr-C-S	Number of thiocarbonyl
fr-HOCCN	Number of C(OH)CCN-Ctert-alkyl or C(OH)CCNcyclic
fr-Imine	Number of Imines
fr-NH0	Number of Tertiary amines
fr-NH1	Number of Secondary amines

Fragment	Definition
fr-NH2	Number of Primary amines
fr-N-O	Number of hydroxylamine groups
fr-Ndealkylation1	Number of XCCNR groups
fr-Ndealkylation2	Number of tert-alicyclic amines
fr-Nhpyrrole	Number of H-pyrrole nitrogens
fr-SH	Number of thiol groups
fr-aldehyde	Number of aldehydes
fr-alkyl-carbamate	Number of alkyl carbamates
fr-alkyl-halide	Number of alkyl halides
fr-allylic-oxid	Number of allylic oxidation sites
fr-amide	Number of amides
fr-amidine	Number of amidine groups
fr-aniline	Number of anilines
fr-aryl-methyl	Number of aryl methyl sites for hydroxylation
fr-azide	Number of azide groups
fr-azo	Number of azo groups
fr-barbitur	Number of barbiturate groups
fr-benzene	Number of benzene rings
fr-benzodiazepine	Number of benzodiazepines with no additional fused rings
fr-bicyclic	Number of bicyclic rings
fr-diazo	Number of diazo groups
fr-dihydropyridine	Number of dihydropyridines
fr-epoxide	Number of epoxide rings
fr-ester	Number of esters
fr-ether	Number of ether oxygens
fr-furan	Number of furan rings
fr-guanido	Number of guanidine groups
fr-halogen	Number of halogens
fr-hdrzine	Number of hydrazine groups
fr-hdrzone	Number of hydrazone groups
fr-imidazole	Number of imidazole rings
fr-imide	Number of imide groups
fr-isocyan	Number of isocyanates
fr-isothiocyan	Number of isothiocyanates
fr-ketone	Number of ketones
fr-ketone-Topliss	Number of ketones excluding diaryl, a,b-unsat.
fr-lactam	Number of beta lactams
fr-lactone	Number of cyclic esters
fr-methoxy	Number of methoxy groups -OCH3
fr-morpholine	Number of morpholine rings
fr-nitrile	Number of nitriles

Fragment	Definition
fr-nitro	Number of nitro groups
fr-nitro-arom	Number of nitro benzene ring substituents
fr-nitro-arom-nonortho	Number of non-ortho nitro benzene ring substituents
fr-nitroso	Number of nitroso groups, excluding NO2
fr-oxazole	Number of oxazole rings
fr-oxime	Number of oxime groups
fr-para-hydroxylation	Number of para-hydroxylation sites
fr-phenol	Number of phenols
fr-phenol-noOrthoHbond	Number of phenolic OH excluding ortho intramolecular Hbond substituents
fr-phos-acid	Number of phosphoric acid groups
fr-phos-ester	Number of phosphoric ester groups
fr-piperdine	Number of piperdine rings
fr-piperzine	Number of piperzine rings
fr-priamide	Number of primary amides
fr-prisulfonamd	Number of primary sulfonamides
fr-pyridine	Number of pyridine rings
fr-quatN	Number of quarternary nitrogens
fr-sulfide	Number of thioether
fr-sulfonamd	Number of sulfonamides
fr-sulfone	Number of sulfone groups
fr-term-acetylene	Number of terminal acetylenes
fr-tetrazole	Number of tetrazole rings
fr-thiazole	Number of thiazole rings
fr-thiocyan	Number of thiocyanates
fr-thiophene	Number of thiophene rings
fr-unbrch-alkane	Number of unbranched alkanes
fr-urea	Number of urea groups

Overview of the extracted fingerprints

Table 3.7 shows overview of molecular fingerprints utilized in this work, feature extraction tools and original papers.

Morgan fingerprints, also known as circular fingerprints, are introduced in [21]. Morgan fingerprints are based on identifying substructures, often called circular substructures, around each atom in the molecule. These circular substructures are formed by traversing the molecular graph from each atom, considering a certain radius or distance cutoff. Each circular substructure is converted into a unique identifier using

Table 3.7: Molecular fingerprints with source library

Source	Fingerprint name	Paper	Length
RDKit	Morgan fingerprints	[21]	2048
	Avalon fingerprints	[24]	512
	MACCSkeys fingerprints	[25]	167
	EState fingerprints	[26]	79
PaDEL	CDK fingerprints		1024
	CDK Extended fingerprints		1024
	Atom pairs fingerprints	[27]	780
	Atom pairs count fingerprints	[27]	780

hashing method. This work considers a fixed-length (2048) binary vector, where each element represents the presence or absence of a specific substructure.

Avalon fingerprints, first introduced in [24], are derived from the three-dimensional (3D) structures of molecules and capture information about their shape, size, and chemical properties. Multiple conformers (different spatial arrangements) of each molecule are generated to capture its flexibility and conformational variability.

MACCS keys, short for "Molecular ACCess System Keys", are the simplest and most restrictive fingerprint implemented in RDKit and introduced in [25]. It counts the occurrences of a collection of pre-defined, expert-derived substructures that are commonly used to quantify pharmacologically relevant molecular similarity. The fingerprints can only be accessed in dense bit vector format.

EState (Extended State) [26] is a method for encoding molecular structure information into numerical descriptors. EState descriptors are based on the concept of assigning partial charges to each atom in a molecule. These partial charges are calculated based on the electronegativity of the atoms and their bonding environment within the molecule.

Atom pair fingerprints were first introduced in [27]. These fingerprints capture pairwise interactions between atoms within a molecule and are widely used in similarity searching, virtual screening, and other molecular modeling applications. An atom pair substructure is defined as a triplet of two (non-hydrogen) atoms and their shortest path distance in the molecular graph, i.e. (atom type 1, atom type 2, geodesic distance). In the standard RDKit implementation, distinct atom types are defined

by tuples of atomic number, number of heavy atom neighbours, aromaticity, and chirality. All unique triplets in a molecule are enumerated and stored in sparse count or bit vector format.

CDK fingerprints, or CDK molecular fingerprints, are a type of molecular fingerprinting method implemented in the Chemistry Development Kit (CDK), an open-source Java library for cheminformatics. Molecular fingerprints are numerical representations of molecular structure and properties. These features can include functional groups, ring systems, etc. Moreover, CDK Extended fingerprints are also presented. They provide a more detailed representation of molecular structure and properties compared to simpler version. They capture a broader range of chemical features and interactions.

3.4 Predictive Modeling

After data processing steps models can be built to make predictions. Choosing the appropriate algorithm is essential for making sophisticated predictions. This section discusses algorithms leveraged in this work and shed light on some basic concepts behind them.

3.4.1 Common approach

Overall, ML can be classified as supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning. Making predictions refers to a supervised learning case since a target label is available. There are two tasks in supervised learning: regression and classification; they are divided based on whether the target is numerical (continuous) or categorical (discrete) respectively. This work consists of some widely used supervised learning algorithms like linear regression, tree-based models, and K- nearest neighbours (KNN).

Common ML workflow considers various steps starting from data collection to model deployment. After data processing one can start model configuration. This involves three steps: **training**, **hyperparameter tuning**, and **testing**.

Training includes model selection and training the selected model (with defined loss function and optimization) on some portion of available data, which is called *train data*. Then, one adjusts the hyperparameters of the selected model by assessing its performance on *validation data*, another portion of available data. To perform hyperparameter tuning grid search or random search of values can be used. Sometimes it is beneficial to perform *cross-validation*, and assess the performance across multiple subsets of the data. A separate portion of the data, known as a *test set*, not seen by the model during the first two steps, is used to evaluate the performance of the model. It is done to check how well the model generalizes the new, unseen data, ensuring that it does not overfit. Overfitting means that the model gives accurate predictions to the train data but not to unseen data, failing at capturing the general pattern as a result of relying on outliers in the train data.

3.4.2 Linear regression: Lasso and Ridge

Linear regression [28] is a statistical method used for modeling the relationship between target values and features. The goal is to find the best-fitting linear line that predicts the target labels.

Linear regression makes several assumptions, including:

1. The relationship between the target and features is linear.

2. The residuals, the differences between predicted and true values, are independent

3. The variance of the residuals is constant across all levels of the independent variables.

4. The residuals are normally distributed.

if the assumptions are violated, it might affect the accuracy of the model.

Given a data set $\{y_i, x_{i1}, ..., x_{ip}\}_{i=1}^{n}$ of n data points, model assumes relation between target (dependent variable) y and features (independent variables) x^p linear

and takes the following form:

$$y_i = \beta_0 + \beta_1 x_{i1} + ... + \beta_p x_{ip} + \epsilon_i = \vec{x_i^T}\vec{\beta} + \epsilon_i,$$

where, $\beta = [\beta_0, \beta_1, ..., \beta_p]$ is a parameter vector (regression bias and coefficients), ϵ is an error term, ^{T} denotes transpose, and $\vec{x_i}$ is extended to $\vec{x_i} = [1, x_{i1}, ...x_{ip}]$ to get a dot product.

Fitting a linear model requires estimating regression coefficients such that the error term is minimized. It is common to use $||\epsilon||_2^2$ as a minimization measure. This method is called **Ordinary Least squares**. The objective is to solve

$$\min_{\beta} L = \min_{\beta} \sum_{i=1}^{n} (y_i - -\vec{x_i^T}\vec{\beta})^2$$

Putting the independent and dependent variables in matrices X and Y respectively the loss function can be rewritten as:

$$L = ||Y - X\vec{\beta}||^2 = (Y - X\vec{\beta})^T(Y - X\vec{\beta}) = Y^TY - Y^TX\vec{\beta} - \vec{\beta}^TX^TY + \vec{\beta}^Tx^TX\vec{\beta}$$

The loss is convex, and according to finding optimum solution after setting the gradient to zero, the optimum parameter is produced as follows:

$$\vec{\beta} = (X^TX)^{-1}X^TY.$$

Lasso Regression

A common problem of the linear regression model is that it might overfit and multicollinearity (high correlation between independent variables). To address this issue we introduce two regularization techniques Lasso and Ridge.

Lasso, short for Least Absolute Shrinkage and Selection Operator, also known as L1 regularization, is a regularization technique [29]. It introduces a penalty term to the loss function as follows:

$$\min_{\beta}(L + \lambda \sum_{j=1}^{p} |\beta_j|),$$

where λ is the regularization parameter, controlling the penalty term.

L1 norm added to loss functions tends to set some coefficients to zero, subsequently, excluding less relevant features. Thus, it is used for feature selection.

The choice of the regularization parameter is critical. A larger λ means stronger regularization, setting more coefficients to exact zero. Cross-validation is often used to find the optimal value of λ and it is considered as a hyperparameter.

Ridge Regression

Ridge Regression, also known as Tikhonov Regularization or L2 regularization, is another regularization technique used in linear regression [30]. A penalty term is added to the OLS objective function as follows:

$$\min_{\beta}(L + \lambda \sum_{j=1}^{p} \beta_j^2),$$

where λ is the regularization parameter, a hyperparameter addressed during cross-validation stage.

Unlike Lasso, Ridge does not set coefficients to exact zero, making different features contribute to the prediction.

3.4.3 DT

The DT is a typical classification model, also it might be used for regression tasks [31].

DT works by recursively partitioning the data into subsets based on the features of the dataset. The algorithm makes decisions at each node of the tree based on the values of the features ultimately leading to a predicted outcome.

- **Roof Node:** Topmost node in the tree, which represents the complete dataset, starting point of the decision-making process
- **Splitting:** Process of dividing a node into two or more child nodes

- **Decision Node:** Child node that can be further divided into other child nodes
- **Leaf Node:** Terminal node without any child node, represents the final outcome
- **Pruning:** Process of removing branches from a tree, used to prevent overfitting

Simple Algorithm

DT algorithm works as follow:

Algorithm 2 DT Construction

1: Begin the tree with the root node
2: Find the best feature based on certain criteria
3: Split data into smaller subsets based on the chosen feature
4: For each subset, repeat steps 2-3 until the stopping criterion is met

Feature Selection Measures

For regression task there are some common splitting criteria. Criteria such as Squared Error, Friedman Mean Squared Error, and Poisson are considered in this work.

Squared Error (Mean Squared Error - MSE) measures the average squared difference between the actual target values (y_i) and the predicted values ($\hat{y}_i$). For a dataset with n instances, MSE is calculated as:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2,$$

where y_i represents the actual target value of the i-th instance, and $\hat{y}_i$ represents the predicted target value of the i-th instance.

Friedman Mean Squared Error (Friedman MSE), introduced by Jerome H. Friedman, extends the concept of MSE with an improvement score, making it particularly suitable for GB regression algorithms. It aims to optimize the mean squared error while considering the improvement gained by splitting at a node. Mathematically, it's quite similar to MSE, but with adjustments for the improvement score.

The **Poisson** criterion is used when the target variable follows a Poisson distribution. The Poisson deviance is minimized, which is analogous to minimizing the mean squared error in ordinary least squares regression but tailored for Poisson-distributed data. The Poisson deviance is calculated as:

$$\text{Poisson Deviance} = 2\sum_{i=1}^{n}\left(y_i \log\left(\frac{y_i}{\hat{y}_i}\right) - (y_i - \hat{y}_i)\right),$$

where y_i represents the actual target value of the i-th instance, and $\hat{y}_i$ represents the predicted target value of the i-th instance.

Stopping conditions

Stopping conditions used in this work consist of setting maximum depth and minimum leaf samples. Maximum depth is the limit of the depth of the tree to avoid overfitting. Minimum leaf samples is a minimum number of samples required to be a leaf node, which may be treated as smoothing criteria

3.4.4 Ensemble method: Random Forest

Ensemble learning methods use multiple learning algorithms to achieve better predictive performance. One such ensemble learning model is **RF** [32].

Core of Random Forest: Decision Trees

RF operates by constructing a multitude of DT (or forests) during training and an outputting the average of predictions of the individual trees. Individual trees tend to grow very deep and learn irregular patterns, and fail at generalization. RF is a way of averaging multiple DT. Moreover, when building each DT, RF consider only a random subset of the features at each split. This helps to overcome the issue of overfitting since it has added benefit of randomness to the mode.

Bagging

The training algorithm involves bootstrap aggregating, or bagging, to tree learners. It utilizes bootstrap sampling to create multiple subsets of the training data. This involves randomly sampling the training data with replacement. Each subset, called a bootstrap sample, is used to train a separate DT.

Algorithm 3 Bagging

Input: Training set X with labels Y, number of trees N
for $i = 1 \rightarrow N$ **do**
    Sample, with replacement, n training examples from X, Y; call these X_i, Y_i
    Train a classification or regression tree f_i on X_i, Y_i
Output: RF model consisting of N trees $\{f_1, \ldots, f_N\}$
Average predictions from all individual trees:

$$\hat{f} = \frac{1}{N} \sum_{i=1}^{N} f_i().$$

Basic Algorithm

Algorithm 4 represents the simple algorithm of RF.

Algorithm 4 RF Algorithm

Input: Training dataset with features X and labels Y, number of trees to grow N, number of features to consider at each split m
for $i = 1 \rightarrow N$ **do**
    Create a bootstrap sample by randomly sampling n examples with replacement from the training data.
    Randomly select m features from the total p features.
    Train a DT using the bootstrap sample and the selected features:
    - At each node, choose the best split based on a chosen criterion (e.g., Gini impurity for classification, mean squared error for regression).
    - Continue recursively partitioning the data until a stopping criterion is met (e.g., maximum depth, minimum samples per leaf).
    Store the trained tree.
Output:
    For classification: Aggregate predictions by majority voting among all trees.
    For regression: Aggregate predictions by averaging the outputs of all trees.

3.4.5 Ensemble method: Gradient Boosting

GB is another powerful ensemble learning method [33]. The general idea is that it works by sequentially adding weak learners, like DTs, to the ensemble, where each new learner is trained to correct the errors made by the previous learners.

Model Training Process

The GB process involves the following steps:

1. **Initialization:** A simple model is initialized to make initial predictions. For regression, this could be the mean of the target variable, and for classification, it could be the mode.

2. **Sequential Learning:** Additional weak learners (trees) are sequentially added to the ensemble. Each new learner focuses on minimizing the errors made by the ensemble so far.

3. **Gradient Descent Optimization:** At each iteration, the new learner is trained to minimize the residual errors of the ensemble by adjusting its predictions using gradient descent.

4. **Shrinkage (Learning Rate):** To control the contribution of each new learner and prevent overfitting, a shrinkage parameter, also known as the learning rate, is introduced.

Algorithm 5 GB Training

Input: Training dataset with features X and labels Y, number of trees N, shrinkage parameter η
Initialize model: $\hat{f}_0(x) = \text{initial\_model}(X, Y)$
for $i = 1 \rightarrow N$ **do**
    Compute residuals: $r_i = Y - \hat{f}_{i-1}(X)$
    Fit base learner h_i to residuals: $h_i = \text{base\_learner}(X, r_i)$
    Update model: $\hat{f}_i(x) = \hat{f}_{i-1}(x) + \eta \cdot h_i(x)$
Output: GB model $\hat{f}(x) = \sum_{i=0}^{N} \eta \cdot h_i(x)$

Hyperparameters

Along with hyperparameters of core trees such as depth of tree, samples per leaf, also number of trees, there are hyperparameters that determines sequential manner. In this work, learning rate and loss functions are optimized. Here are explanations for some commonly used loss functions in GB.

Squared Error, also known as Mean Squared Error (MSE), is a widely used loss function in GB regression. It measures the average squared difference between the actual target values (y_i) and the predicted values ($\hat{y}_i$). Mathematically, MSE is defined as:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2,$$

where n is the number of instances, y_i represents the actual target value of the i-th instance, and $\hat{y}_i$ represents the predicted target value of the i-th instance.

GB with MSE loss typically results in predictions that converge towards the mean of the target values, as the algorithm minimizes the squared differences between the actual and predicted values.

Absolute Error, also known as Mean Absolute Error (MAE), is another loss function used in GB regression. It measures the average absolute difference between the actual target values (y_i) and the predicted values ($\hat{y}_i$)). Mathematically, MAE is defined as:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|.$$

MAE is less sensitive to outliers compared to MSE because it doesn't square the errors. Therefore, GB with MAE loss might be more robust to outliers in the data.

Huber Loss is a hybrid loss function that combines the characteristics of MSE and MAE. It behaves like MSE for small errors but like MAE for large errors, making it less sensitive to outliers while still being differentiable everywhere. The Huber Loss function is defined as:

$$\text{Huber}(y, \hat{y}) = \begin{cases} \frac{1}{2}(y - \hat{y})^2 & \text{if } |y - \hat{y}| \leq \delta \\ \delta(|y - \hat{y}| - \frac{1}{2}\delta) & \text{otherwise} \end{cases},$$

where δ is a threshold parameter that determines when the loss transitions from quadratic to linear. Typically, δ is set based on heuristics or cross-validation.

Huber Loss provides a compromise between the robustness of MAE and the efficiency of MSE, making it suitable for scenarios where the data may contain outliers but still benefit from leveraging squared errors for small deviations.

3.4.6 K-Nearest Neighbours

KNN [34] is a simple ML algorithm, belongs to lazy learning algorithm. Lazy learning means that it processes train data only to make. The idea behind KNN to make predictions for a new data point based on the majority label or the average of the K-nearest data points for classification or regression tasks respectfully.

Algorithm

The simple algorithm of KNN is as follows.

Algorithm 6 K-Nearest Neighbors (KNN)

Require: Training data set: *train_data*, Test data set: *test_data*, Value of K: k
Ensure: Predicted labels for test data set
1: Select value of K (number of nearest neighbors).
2: **for** each data point in *test_data* **do**
3: Calculate distance between the current test data point and all training data points using some distance metric.
4: Find the K data points among *train_data* with the smallest distance based on the chosen distance measure. These are the nearest neighbors.
5: **if** Classification task **then**
6: Determine the class labels of the K nearest neighbors by majority voting.
7: The class with the highest occurrence becomes the predicted class for the current test data point.
8: **if** Regression task **then**
9: Calculate the class label for the current test data point by taking the average of the target values of the K nearest neighbors.
return Predicted labels for the entire test data set.

Hyperparameters

The choice of distance and average measures affect the performance of the model.

Euclidean distance is the cartesian distance between the two points which are hyperplane, and calculated as follows:

$$d(x, y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}.$$

Manhattan Distance, also known as Taxicab distance, is the distance between two points measured along axes at right angles, and calculated as follows:

$$d(x, y) = \sum_{i=1}^{n} |x_i - y_i|.$$

Both Euclidean distance and Manhattan distance are special cases of the **Minkowski distance**:

$$d(x, y) = (\sum_{i=1}^{n}(x_i - y_i)^p)^{\frac{1}{p}},$$

with $p = 2$ and $p = 1$ respectfully.

Predicting target value is performed by majority rule. All points in each neighborhood can be used in prediction with the same weight, or with weight by the inverse of their distances, i.e. closer neighbours will have a greater influence.

3.4.7 Evaluation metrics

During hyperparameter tuning and testing when the evaluation is performed, evaluation metrics (loss functions) are considered. There are different common metrics for regression tasks [35]. **Mean Squared Error (MSE)** is the average of squared differences between predicted and actual values, and calculated as follows:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2,$$

where y_i is a true value, $\hat{y}_i$ is a predicted value, and n is a number of data points.

Sometimes the square root of the MSE or **Root Mean Squared Error (RMSE)** is considered.

Mean Absolute Error (MAE) is the average of absolute differences between true and predicted values, and calculated as follows:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|.$$

Additionally, the mean of absolute difference divided to the true value, or **Mean Absolute Percentage Error (MAPE)** can be considered as a loss funtion.

R-squared or Coefficient of the determination (R^2) represents the proportion of variance in the dependent variable explained by the independent variables, and is calculated as follows:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2},$$

where $\bar{y}$ is the mean of true values.

3.5 Research workflow

Fig. 3-2 illustrates the general workflow of the work. In general, raw data is being handled with the help of some imputation and feature engineering approaches to produce imputed data. Imputed data as well as original data is utilized to train ML regression models. The process is repeated for different combinations of models. Testing evaluates not only predictions but also the success of imputation and feature engineering.

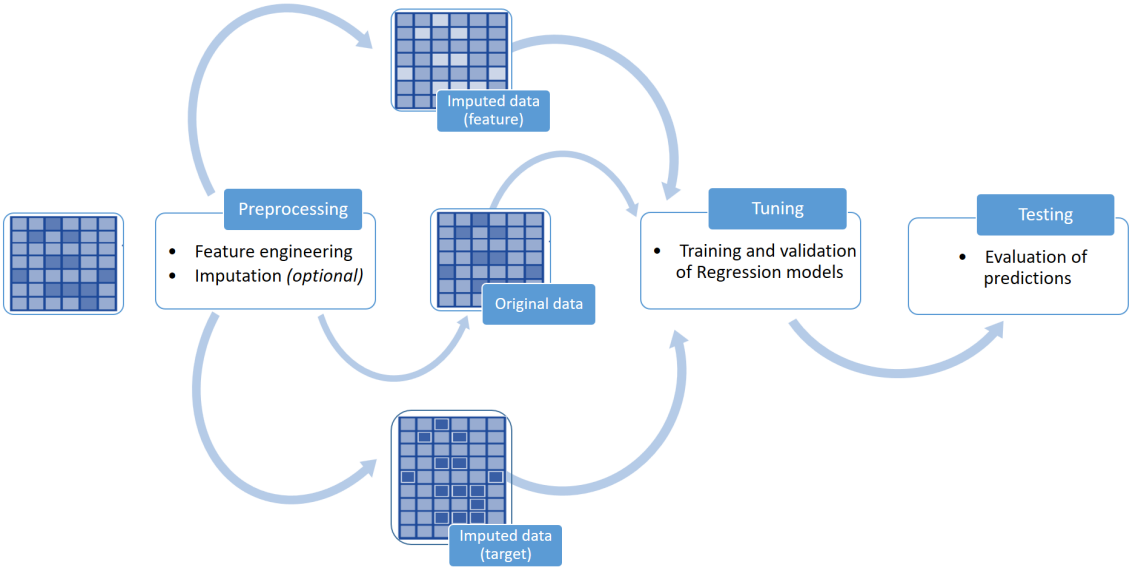

Figure 3-2: Research workflow diagram

Chapter 4

Results

4.1 Imputation

Before training and hyperparameter selection, one should process data. There are missing values in the dataset, thus, there is an opportunity to expand the dataset to provide a more complete information analysis. Therefore, we perform imputation, and "augment" the dataset by filling in those missing values. Imputation methodology is described previously in section 3.2.2, this section covers the results after the selection of the best imputation models.

4.1.1 Hyperparameter tuning of imputers

As mentioned before, in this work imputation algorithms such as Iterative Imputer and Nearest Neighbours are utilized. Iterative Imputer uses Decision Tree, XGBoost, Linear, and Ridge regression models as core estimators. During hyperparameter tuning, hyperparameters of estimators, like maximum depth of tree, or regularization strength, are also selected along with imputation strategies.

Table 4.1 shows the hyperparameter set of each model used as an imputer, and also the strategy regarding imputation strategy.

Table 4.1: Imputation strategy: hyperparameter tuning

Model	Hyperparameter	Definition	Parameter grid
Iterative Imputer	imputation_order	order in which the features will be imputed	ascending, descending
Iterative Imputer: Linear			
Iterative Imputer : Ridge	alpha	regularization strength	$1, 10, 10^2, 10^3, 10^4, 10^5$
Iterative Imputer: XGBoost	eta	step size shrinkage	0.1, 0.3, 0.5
	max_depth	maximum depth of tree	3, 6, 10
	gamma	minimum loss reduction	0, 0.1, 1
Iterative Imputer: Decision Tree	criterion	function to measure the quality of a split	squared error, friedman mse, poisson
	max_depth	maximum depth of tree	5, 10, 50
	min_samples_leaf	minimum number of samples required to be a leaf node	5, 10, 15
Nearest Neighbours Imputer	n_neigbours	number of neighbours	5, 10, 15, 25
	weights	weight function used in prediction	uniform, distance

4.1.2 Evaluation of imputation models

As described before, for the validation purposes random points from original data are being removed. Then, each imputer is used repeatedly on each produced data, and the sum of relative errors is calculated. Imputers are ranked at each step, then ranks are aggregated, and imputer with consecutively better results is chosen as the best.

Table 4.2 shows best selected models ranking at each step of data removal.

Figure 4-1 shows growth of the sum of relative errors with less available true data. Each model is chosen with the best hyperparamaters. As a result, imputation with XGBoost shows best results. Original data is imputed with this selected model.

Table 4.2: Ranking of imputation models based on the sum of relative errors

Number of missing points **Model**	**10**	**100**	**1000**	**10000**	**Overall**
Nearest Neighbours Imputation, n_neighbours = 5, weights = distance properties with descriptors	3	**1**	**1**	5	2
Nearest Neighbours Imputation, n_neighbours = 5 ,weights = distance	10	7	5	3	5
Iterative Imputer with Ridge, ascending order, alpha = 10^5 properties with descriptors	4	4	7	6	6
Iterative Imputer with Ridge, ascending order, alpha = 10^5	5	5	8	7	7
Iterative Imputer with XGBoost, descending order, eta=0.5, gamma=0, max_depth=6 properties with descriptors	**1**	3	2	2	**1**
Iterative Imputer with XGBoost, descending order, eta=0.3, gamma=1, max_depth=6	7	2	6	**1**	3
Iterative Imputer with Decision Tree, descending order, criterion=squared_error, min samples for leaf=5, max_depth=50 properties with descriptors	2	6	3	4	4
Iterative Imputer with Decision Tree, descending order, criterion = squared_error, min_samples_leaf=10, max_depth=5	9	8	4	8	8
Iterative Imputer with Linear Regression, descending order properties with descriptors	6	10	10	10	10
Iterative Imputer with Linear Regression, descending order	8	9	9	9	9

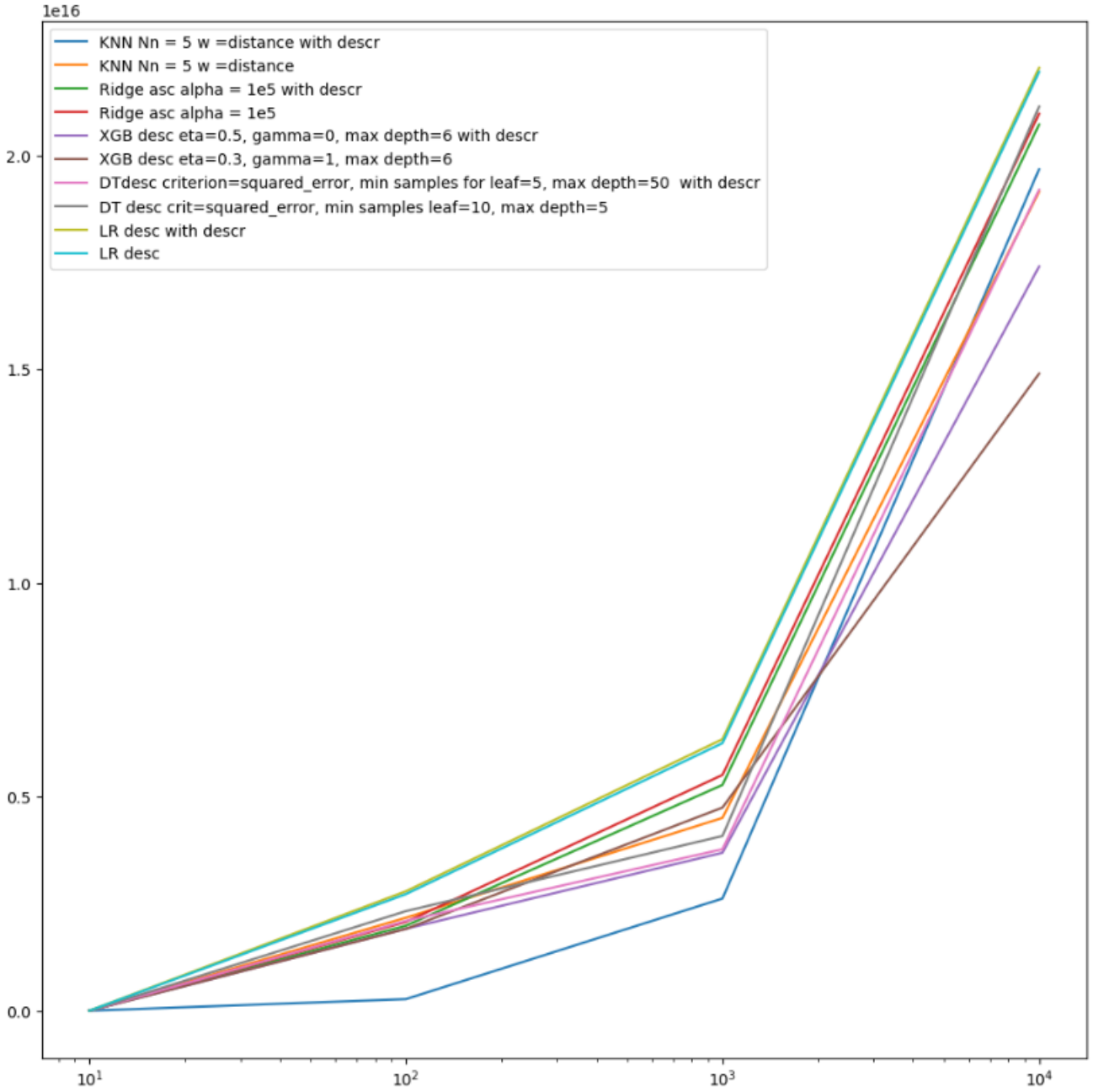

Figure 4-1: Imputation model results : averaged sum of relative errors against number of removed data points

4.2 Hyperparameter tuning of predictors on original data

As mentioned before, to predict 5 target properties number of estimators, such as Lasso, Ridge, DT, and KNN, are used. Moreover, number of descriptors, such as Avalon, Morgan, and more, are utilized as feature space for predictiv models. Overall their performance has showed acceptable results. For some properties, like *Quantum yield*, and *Lifetime*, ensemble methods, such as RF and GB are utilized. Before constructing models, data is split into the train, validation, and test sets with a ratio of 70% - 15% - 15% for each property on validation set hyperparameter tuning with grid search is performed. This section captures the information on hyperparameters of each model to be tuned and the results of the validation. Results include the overall information on selected models to evaluate on test set, and their evaluation scores on test set.

4.2.1 Hyperparameter set

Table 4.3 shows hyperparameter set used for tuning each model predicting each of 5 target variables. In other words, for each property the same hyperparameters from same parameter grid are tuned.

Table 4.3: Hyperparameter set for validation

Model	**Hyperparameter**	**Definition**	**Parameter grid**
Lasso	alpha	regularization strength	$10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 10^2, 10^3, 10^4, 10^5$
Ridge	alpha	regularization strength	$10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 10^2, 10^3, 10^4, 10^5$
DT	criterion	function to measure the quality of a split	squared error, friedman mse, poisson
	min_samples_leaf	minimum number of samples required to be a leaf node	5, 10, 20, 50
	max_depth	maximum depth of tree	50, 100, 200
KNN	weights	weight function used in prediction	uniform, distance
	p	p in Minkowski distance	1, 2
	n_neigbours	number of neighbours	5, 10, 15
RF	criterion	function to measure the quality of a split	squared error, friedman mse, poisson
	min_samples_leaf	minimum number of samples required to be a leaf node	5, 10, 20, 50
	max_depth	maximum depth of individual tree	50, 100, 200
	n_estimators	number of estimators	50, 100, 200
GB	loss	loss function to be optimized	squared error, absolute error, huber
	learning_rate	learning rate or shrinkage	0.05, 0.1, 0.5
	criterion	function to measure the quality of a split	squared error, friedman mse
	max_depth	maximum depth of individual tree	3, 6, 10

4.2.2 Selected models after validation

This section provides the information on selected models to predict each of target variables after validation stage.

Table 4.4 provides hyperparameters of the selected models to predict *Maximum absorption wavelength* after validation.

Table 4.4: Selected models to predict *Maximum absorption wavelength* after validation

Model	**Lasso**	**Ridge**	**Decision Tree**			**KNN**		
Hyperp. Repr.	alpha	alpha	criterion	min samples leaf	max depth	weights	p	n neigbours
Morgan	0.1	1000	friedman mse	5	100	distance	1	5
EState	0.00001	0.01	squared error	5	50	distance	1	5
MACCS	0.01	10	poisson	5	50	distance	1	5
Avalon	0.01	100	poisson	5	50	distance	1	5
CDK	0.1	1000	friedman mse	5	50	distance	1	5
CDK Extended	0.1	1000	poisson	5	50	distance	1	5
Atom Pairs	0.01	100	friedman mse	5	50	distance	1	5
Atom Pairs Count	0.01	100	squared error	5	50	distance	1	5
Descriptors	0.01	1	poisson	5	50	distance	1	5

Table 4.5 provides hyperparameters of the selected models to predict *Maximum emission wavelength* after validation.

Tables 4.6, 4.7 provide hyperparameters of the selected models to predict *Quantum yield* after validation.

Table 4.8 provides hyperparameters of the selected models to predict *Extinction coefficient* after validation.

Tables 4.9, 4.10 provide hyperparameters of the selected models to predict *Lifetime* after validation.

Table 4.5: Selected models to predict *Maximum emission wavelength* after validation

Model	**Lasso**	**Ridge**	**Decision Tree**			**KNN**		
Hyperp. Repr.	alpha	alpha	criterion	min samples leaf	max depth	weights	p	n neigbours
Morgan	0.1	1000	squared error	5	100	distance	1	5
EState	0.1	10	squared error	5	50	distance	1	5
MACCS	0.01	10	poisson	5	50	distance	1	5
Avalon	0.1	100	poisson	5	100	distance	1	5
CDK	0.1	1000	poisson	5	50	distance	1	5
CDK Extended	0.1	1000	poisson	5	50	distance	1	10
Atom Pairs	0.01	10	friedman mse	5	50	distance	1	5
Atom Pairs Count	0.1	100	poisson	5	50	distance	1	5
Descriptors	0.01	1	poisson	5	50	distance	1	5

Table 4.6: Selected models to predict *Quantum yield* after validation

Model	**Lasso**	**Ridge**	**Decision Tree**			**KNN**		
Hyperp. Repr.	alpha	alpha	criterion	min samples leaf	max depth	weights	p	n neigbours
Avalon	0.001	1000	poisson	10	50	distance	1	10
Morgan	0.001	1000	friedman mse	5	100	distance	1	5
MACCS	0.0001	100	poisson	5	50	distance	1	5
Atom Pairs Count	0.001	1000	squared error	5	50	distance	1	10
CDK	0.001	1000	friedman mse	10	50	distance	1	10
Atom Pairs	0.001	100	squared error	5	50	distance	1	15
CDK Extended	00.001	1000	poisson	10	50	distance	1	10
EState	0.001	1000	squared error	5	50	distance	1	10
Descriptors	0.001	100000	poisson	10	50	distance	1	5

Table 4.7: Selected models to predict *Quantum yield* after validation (continued)

Model	**RF**				**GB**			
Hyperp. Repr.	criterion	min samples leaf	max depth	n est.	criterion	learning rate	loss	max depth
Avalon	poisson	50	5	200	squared error	0.1	squared error	10
Morgan	poisson	200	5	200	squared error	0.5	squared error	10
MACCS	poisson	50	5	50	squared error	0.1	squared error	10
Atom Pairs Count	poisson	50	5	200	squared error	0.1	squared error	10
CDK	poisson	50	5	200	squared error	0.1	squared error	10
Atom Pairs	poisson	100	5	100	squared error	0.1	squared error	10
CDK Extended	poisson	50	5	200	friedman mse	0.05	squared error	10
EState	poisson	50	5	200	squared error	0.1	squared error	10
Descriptors	poisson	50	5	200	squared error	0.1	huber	10

Table 4.8: Selected models to predict *Extinction coefficient* after validation

Model	**Lasso**	**Ridge**	**Decision Tree**			**KNN**		
Hyperp. Repr.	alpha	alpha	criterion	min samples leaf	max depth	weights	p	n neigbours
Descriptors	0.001	10	friedman mse	5	50	distance	1	5
EState	0.001	100	poisson	5	50	distance	1	5
MACCS	0.0001	10	squared error	10	50	distance	1	5
Morgan	0.001	1000	poisson	5	50	distance	1	5
Avalon	0.001	1000	squared error	10	50	distance	1	5
CDK	0.001	1000	poisson	5	50	distance	1	5
CDK Extended	0.001	1000	friedman mse	5	50	distance	1	5
Atom Pairs	0.001	1000	poisson	5	50	distance	1	5
Atom Pairs Count	0.001	1000	friedman mse	5	50	distance	1	5

Table 4.9: Selected models to predict *Lifetime* after validation

Model	**Lasso**	**Ridge**	**Decision Tree**			**KNN**		
Hyperp. Repr.	alpha	alpha	criterion	min samples leaf	max depth	weights	p	n neigbours
Avalon	0.001	100	poisson	5	50	distance	1	5
Morgan	0.01	1000	poisson	5	100	distance	1	5
MACCS	0.0001	100	poisson	5	50	distance	1	5
Atom Pairs Count	0.001	100	friedman mse	5	50	distance	1	10
CDK	0.001	100	friedman mse	10	50	distance	1	5
Atom Pairs	0.001	100	poisson	5	50	distance	1	10
CDK Extended	0.001	100	poisson	5	50	distance	1	10
EState	1e-7	1e-5	poisson	5	50	distance	1	10
Descriptors	0.001	1	poisson	10	50	distance	1	5

Table 4.10: Selected models to predict *Lifetime* after validation (continued)

Model	**RF**				**GB**			
Hyperp. Repr.	criterion	min samples leaf	max depth	n est.	criterion	learning rate	loss	max depth
Avalon	friedman mse	50	5	200	squared error	0.1	huber	10
Morgan	squared error	100	5	50	squared error	0.5	huber	10
MACCS	poisson	200	5	50	squared error	0.05	squared error	10
Atom Pairs Count	friedman mse	50	5	100	squared error	0.1	squared error	10
CDK	friedman mse	50	5	50	friedman mse	0.05	squared error	10
Atom Pairs	squared error	50	5	50	squared error	0.1	squared error	10
CDK Extended	friedman mse	100	5	50	friedman mse	0.1	huber	6
EState	friedman mse	50	5	100	squared error	0.1	squared error	10
Descriptors	friedman mse	50	5	100	squared error	0.1	squared error	10

4.3 Hyperparameter tuning of predictors on augmented data

After collecting imputed data, one follows feature or target imputation process as described in section 3.2.1. Then hyperparameter tuning is performed to train models on imputed data. This section shows the results of validating predictors on augmented data. The hyperparameter set is kept the same as described previously in section 4.2.1.

4.3.1 Selected models after validation with feature imputation

This section provides the information on selected models to predict each of target variables with feature imputation after validation stage. Feature imputation means that in the features space along with extracted descriptors other target properties (imputed, if not provided) are utilized.

Table 4.11 provides hyperparameters of the selected models to predict *Maximum absorption wavelength* after validation.

Table 4.11: Selected models to predict *Maximum absorption wavelength* after validation of augmented data with feature imputation

Model	**Lasso**	**Ridge**	**Decision Tree**			**KNN**		
Hyperp. Repr.	alpha	alpha	criterion	min samples leaf	max depth	weights	p	n neigbours
Morgan	0.1	100	squared error	5	100	distance	1	5
EState	0.1	100	poisson	5	50	distance	1	5
MACCS	0.001	10	poisson	10	50	distance	1	5
Avalon	0.01	100	poisson	5	50	distance	1	5
CDK	0.01	100	squared error	5	50	distance	1	5
CDK Extended	0.1	1000	poisson	10	50	distance	1	5
Atom Pairs	0.01	100	poisson	10	50	distance	1	5
Atom Pairs Count	0.1	10	poisson	5	50	distance	1	5
Descriptors	0.001	1	poisson	5	50	distance	1	5

Table 4.12 provides hyperparameters of the selected models to predict *Maximum emission wavelength* after validation.

Table 4.12: Selected models to predict *Maximum emission wavelength* after validation of augmented data with feature imputation

Model	**Lasso**	**Ridge**	**Decision Tree**			**KNN**		
Hyperp. Repr.	alpha	alpha	criterion	min samples leaf	max depth	weights	p	n neigbours
Morgan	0.1	100	poisson	5	50	distance	1	5
EState	0.1	100	squared error	5	50	distance	1	5
MACCS	0.0	100	friedman mse	5	50	distance	1	5
Avalon	0.01	100	poisson	5	50	distance	1	5
CDK	0.01	100	squared error	10	50	distance	1	5
CDK Extended	0.1	100	poisson	10	50	distance	2	5
Atom Pairs	0.01	10	squared error	10	50	distance	1	5
Atom Pairs Count	0.1	1000	squared error	10	50	distance	1	5
Descriptors	0.00001	1	poisson	5	50	distance	1	5

Tables 4.13, 4.14 provide hyperparameters of the selected models to predict *Quantum yield* after validation.

Table 4.15 provides hyperparameters of the selected models to predict *Extinction coefficient* after validation.

Tables 4.16, 4.17 provide hyperparameters of the selected models to predict *Lifetime* after validation.

Table 4.13: Selected models to predict *Quantum yield* after validation of augmented data with feature imputation

Model	**Lasso**	**Ridge**	**Decision Tree**			**KNN**		
Hyperp. Repr.	alpha	alpha	criterion	min samples leaf	max depth	weights	p	n neigbours
Avalon	0.001	100	squared error	10	50	distance	1	5
Morgan	0.001	1000	poisson	5	100	distance	1	5
MACCS	0.0001	10	poisson	5	50	distance	1	10
Atom Pairs Count	0.001	1000	poisson	20	50	distance	1	5
CDK	0.001	1000	poisson	10	50	distance	1	10
Atom Pairs	0.001	100	poisson	5	50	distance	1	10
CDK Extended	0.001	1000	poisson	5	50	distance	1	10
EState	0.001	100	poisson	10	50	distance	1	5
Descriptors	0.01	10^6	poisson	10	50	distance	1	5

Table 4.14: Selected models to predict *Quantum yield* after validation of augmented data with feature imputation (continued)

Model	**RF**				**GB**			
Hyperp. Repr.	criterion	min samples leaf	max depth	n est.	criterion	learning rate	loss	max depth
Avalon	poisson	50	5	200	friedman mse	0.1	huber	10
Morgan	poisson	100	5	200	squared error	0.1	squared error	10
MACCS	poisson	50	5	200	squared error	0.1	squared error	10
Atom Pairs Count	poisson	50	5	200	squared error	0.1	squared error	10
CDK	poisson	100	5	200	friedman mse	0.1	huber	10
Atom Pairs	poisson	50	5	200	squared error	0.1	huber	10
CDK Extended	poisson	50	5	200	squared error	0.05	squared error	10
EState	poisson	50	5	200	friedman mse	0.1	squared error	10
Descriptors	poisson	50	5	200	squared error	0.1	squared error	10

Table 4.15: Selected models to predict *Extinction coefficient* after validation of augmented data with feature imputation

Model	**Lasso**	**Ridge**	**Decision Tree**			**KNN**		
Hyperp. Repr.	alpha	alpha	criterion	min samples leaf	max depth	weights	p	n neigbours
Descriptors	0.001	10	poisson	5	50	distance	1	5
EState	0.001	100	friedman mse	5	50	distance	1	5
MACCS	0.0001	10	poisson	5	50	distance	1	5
Morgan	0.001	1000	poisson	5	50	distance	1	5
Avalon	0.001	1000	poisson	5	50	distance	1	5
CDK	0.001	1000	poisson	10	50	distance	1	5
CDK Extended	0.001	1000	squared error	10	50	distance	1	5
Atom Pairs	0.001	1000	friedman mse	5	50	distance	1	5
Atom Pairs Count	0.001	1000	poisson	5	50	distance	1	5

Table 4.16: Selected models to predict *Lifetime* after validation of augmented data with feature imputation

Model	**Lasso**	**Ridge**	**Decision Tree**			**KNN**		
Hyperp. Repr.	alpha	alpha	criterion	min samples leaf	max depth	weights	p	n neigbours
Avalon	0.001	100	friedman mse	5	50	distance	1	5
Morgan	0.001	100	poisson	5	50	distance	1	5
MACCS	0.001	10	poisson	10	50	distance	1	10
Atom Pairs Count	0.001	100	poisson	5	50	distance	1	5
CDK	0.001	100	poisson	5	50	distance	1	5
Atom Pairs	0.01	100	squared error	20	50	distance	1	5
CDK Extended	0.001	100	poisson	5	50	distance	2	5
EState	10^{-5}	10^{-2}	poisson	10	50	distance	1	5
Descriptors	0.001	1	friedman mse	5	50	distance	1	5

Table 4.17: Selected models to predict *Lifetime* after validation of augmented data with feature imputation (continued)

Model	**RF**				**GB**			
Hyperp. Repr.	criterion	min samples leaf	max depth	n est.	criterion	learning rate	loss	max depth
Avalon	squared error	200	5	200	squared error	0.1	huber	10
Morgan	poisson	50	5	50	friedman mse	0.1	huber	10
MACCS	friedman mse	100	5	200	squared error	0.05	squared error	10
Atom Pairs Count	squared error	50	5	100	squared error	0.1	squared error	10
CDK	squared error	50	5	200	friedman mse	0.1	huber	10
Atom Pairs	poisson	50	5	100	squared error	0.1	squared error	10
CDK Extended	friedman mse	50	5	200	friedman mse	0.1	squared error	6
EState	friedman mse	20	5	200	friedman mse	0.1	huber	10
Descriptors	friedman mse	50	5	200	friedman mse	0.1	squared error	10

4.3.2 Selected models after validation with target imputation

This section provides the information on selected models to predict each of target variables with target imputation after validation stage. Target imputation means that along with other target properties, imputed data points are added to the train data.

Table 4.18 provides hyperparameters of the selected models to predict *Maximum absorption wavelength* after validation.

Table 4.18: Selected models to predict *Maximum absorption wavelength* after validation of augmented data with target imputation

Model	**Lasso**	**Ridge**	**Decision Tree**			**KNN**		
Hyperp. Repr.	alpha	alpha	criterion	min samples leaf	max depth	weights	p	n neigbours
Morgan	1	100	friedman mse	20	50	distance	1	15
EState	1	100	poisson	50	50	distance	1	15
MACCS	1	100	poisson	10	50	distance	1	15
Avalon	1	1000	poisson	20	50	distance	1	15
CDK	1	100	squared error	50	50	distance	1	15
CDK Extended	1	100	squared error	50	50	distance	1	15
Atom Pairs	1	1000	poisson	20	50	distance	1	15
Atom Pairs Count	1	1000	squared error	50	50	distance	1	15
Descriptors	0.1	10^6	squared error	50	50	distance	1	15

Table 4.19 provides hyperparameters of the selected models to predict *Maximum emission wavelength* after validation.

Tables 4.20, 4.21 provide hyperparameters of the selected models to predict *Quantum yield* after validation.

Table 4.22 provides hyperparameters of the selected models to predict *Extinction coefficient* after validation.

Tables 4.23, 4.24 provide hyperparameters of the selected models to predict *Lifetime* after validation.

Table 4.19: Selected models to predict *Maximum emission wavelength* after validation after validation of augmented data with target imputation

Model	**Lasso**	**Ridge**	**Decision Tree**			**KNN**		
Hyperp. Repr.	alpha	alpha	criterion	min samples leaf	max depth	weights	p	n neigbours
Morgan	1	100	squared error	20	50	distance	1	15
EState	1	1000	squared error	20	50	distance	2	5
MACCS	1	100	squared error	50	50	distance	1	10
Avalon	1	100	poisson	20	50	distance	2	15
CDK	0.1	1000	poisson	50	50	distance	2	10
CDK Extended	1	1000	poisson	20	50	distance	2	5
Atom Pairs	1	1000	poisssn	50	50	distance	2	15
Atom Pairs Count	0.1	1000	poisson	50	50	distance	2	15
Descriptors	1	1000	squared error	50	50	distance	1	5

Table 4.20: Selected models to predict *Quantum yield* after validation after validation of augmented data with target imputation

Model	**Lasso**	**Ridge**	**Decision Tree**			**KNN**		
Hyperp. Repr.	alpha	alpha	criterion	min samples leaf	max depth	weights	p	n neigbours
Avalon	0.01	10^4	poisson	50	50	distance	2	15
Morgan	0.01	10^4	poisson	50	50	distance	1	15
MACCS	0.01	10^4	squared error	50	50	distance	1	15
Atom Pairs Count	0.01	10^4	poisson	50	50	distance	1	15
CDK	0.01	10^4	poisson	50	50	distance	2	15
Atom Pairs	0.01	10^4	squared error	50	50	distance	1	15
CDK Extended	0.01	10^4	squared error	50	50	uniform	2	15
EState	0.01	1000	poisson	50	50	distance	1	15
Descriptors	0.01	10^6	poisson	50	50	distance	1	15

Table 4.21: Selected models to predict *Quantum yield* after validation after validation of augmented data with target imputation (continued)

Model	**RF**				**GB**			
Hyperp. Repr.	criterion	min samples leaf	max depth	n est.	criterion	learning rate	loss	max depth
Avalon	friedman mse	50	5	200	friedman mse	0.1	squared error	6
Morgan	poisson	100	5	200	squared error	0.05	squared error	10
MACCS	friedman mse	50	5	200	squared error	0.05	huber	10
Atom Pairs Count	squared error	50	5	200	squared error	0.1	squared error	10
CDK	poisson	100	5	200	squared error	0.1	squared error	6
Atom Pairs	squared error	50	5	100	squared error	0.1	huber	10
CDK Extended	friedman mse	50	5	200	squared error	0.05	squared error	6
EState	squared error	100	5	200	friedman mse	0.05	squared error	10
Descriptors	squared error	50	5	200	friedman mse	0.1	huber	10

Table 4.22: Selected models to predict *Extinction coefficient* after validation after validation of augmented data with target imputation

Model	**Lasso**	**Ridge**	**Decision Tree**			**KNN**		
Hyperp. Repr.	alpha	alpha	criterion	min samples leaf	max depth	weights	p	n neigbours
Descriptors	1	10^6	squared error	50	50	distance	1	15
EState	0.01	1000	friedman mse	50	50	distance	1	15
MACCS	0.1	10^4	squared error	50	50	distance	2	15
Morgan	0.1	10^5	poisson	50	50	distance	2	10
Avalon	0.1	10^4	poisson	50	50	distance	2	15
CDK	0.1	10^4	poisson	50	50	distance	2	15
CDK Extended	0.1	10^5	poisson	50	50	distance	2	15
Atom Pairs	0.1	10^5	friedman mse	50	50	distance	2	15
Atom Pairs Count	0.1	10^5	squared error	50	50	distance	1	15

Table 4.23: Selected models to predict *Lifetime* after validation after validation of augmented data with target imputation

Model	**Lasso**	**Ridge**	**Decision Tree**			**KNN**		
Hyperp. Repr.	alpha	alpha	criterion	min samples leaf	max depth	weights	p	n neigbours
Avalon	1	10^6	poisson	50	50	uniform	2	15
Morgan	1	10^6	poisson	50	50	uniform	2	15
MACCS	1	10^6	poisson	50	50	uniform	1	15
Atom Pairs Count	0.001	100	poisson	50	50	uniform	1	15
CDK	1	10^6	poisson	50	50	uniform	2	15
Atom Pairs	1	10^6	squared error	50	50	uniform	1	15
CDK Extended	1	10^6	poisson	50	50	uniform	2	15
EState	1	10^6	poisson	50	50	distance	1	15
Descriptors	1	10^6	squared error	50	50	distance	1	15

Table 4.24: Selected models to predict *Lifetime* after validation after validation of augmented data with target imputation (continued)

Model	**RF**				**GB**			
Hyperp. Repr.	criterion	min samples leaf	max depth	n est.	criterion	learning rate	loss	max depth
Avalon	poisson	100	50	100	squared error	0.05	absolute error	3
Morgan	friedman mse	50	50	50	friedman mse	0.1	absolute error	10
MACCS	friedman mse	200	50	50	squared error	0.05	absolute error	3
Atom Pairs Count	friedman mse	100	50	50	squared error	0.05	absolute error	6
CDK	friedman mse	200	50	100	friedman mse	0.05	absolute error	3
Atom Pairs	poisson	100	50	200	friedman mse	0.05	absolute error	10
CDK Extended	friedman mse	50	50	50	squared error	0.05	absolute error	3
EState	friedman mse	50	50	200	friedman mse	0.05	absolute error	3
Descriptors	squared error	200	50	50	friedman mse	0.05	absolute error	3

4.4 Evaluation

After performing data augmentation and hyperparameter tuning, one can assess performance of the selected models on the test set. This section holds the information of the performance results of all models covered in this thesis.

4.4.1 Evaluating models trained on original data

After hyperparameter tuning, models are trained on train and validation set of original data to capture all available information. Following figures show results of evaluating models on each test set corresponding to 5 target values.

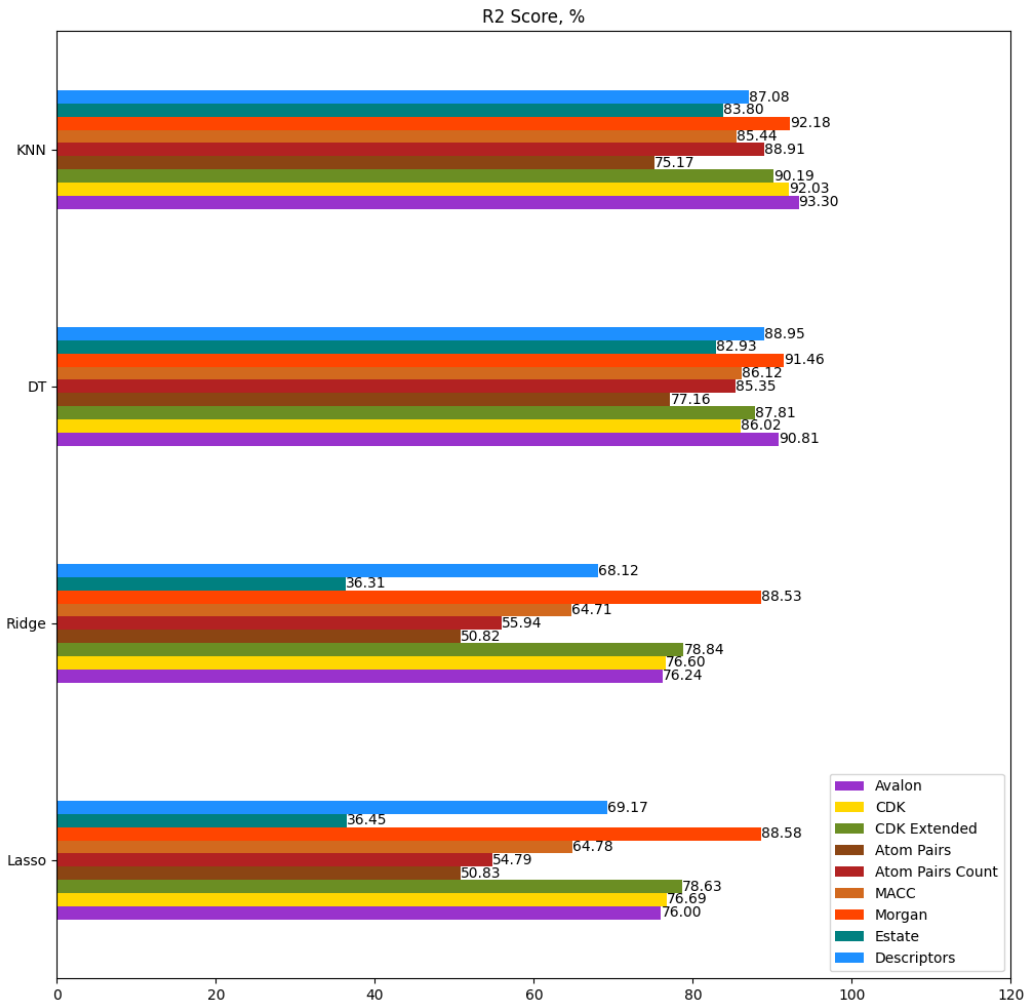

Figure 4-2: R^2 scores of models, trained on original data, for *Maximum absorption wavelength*

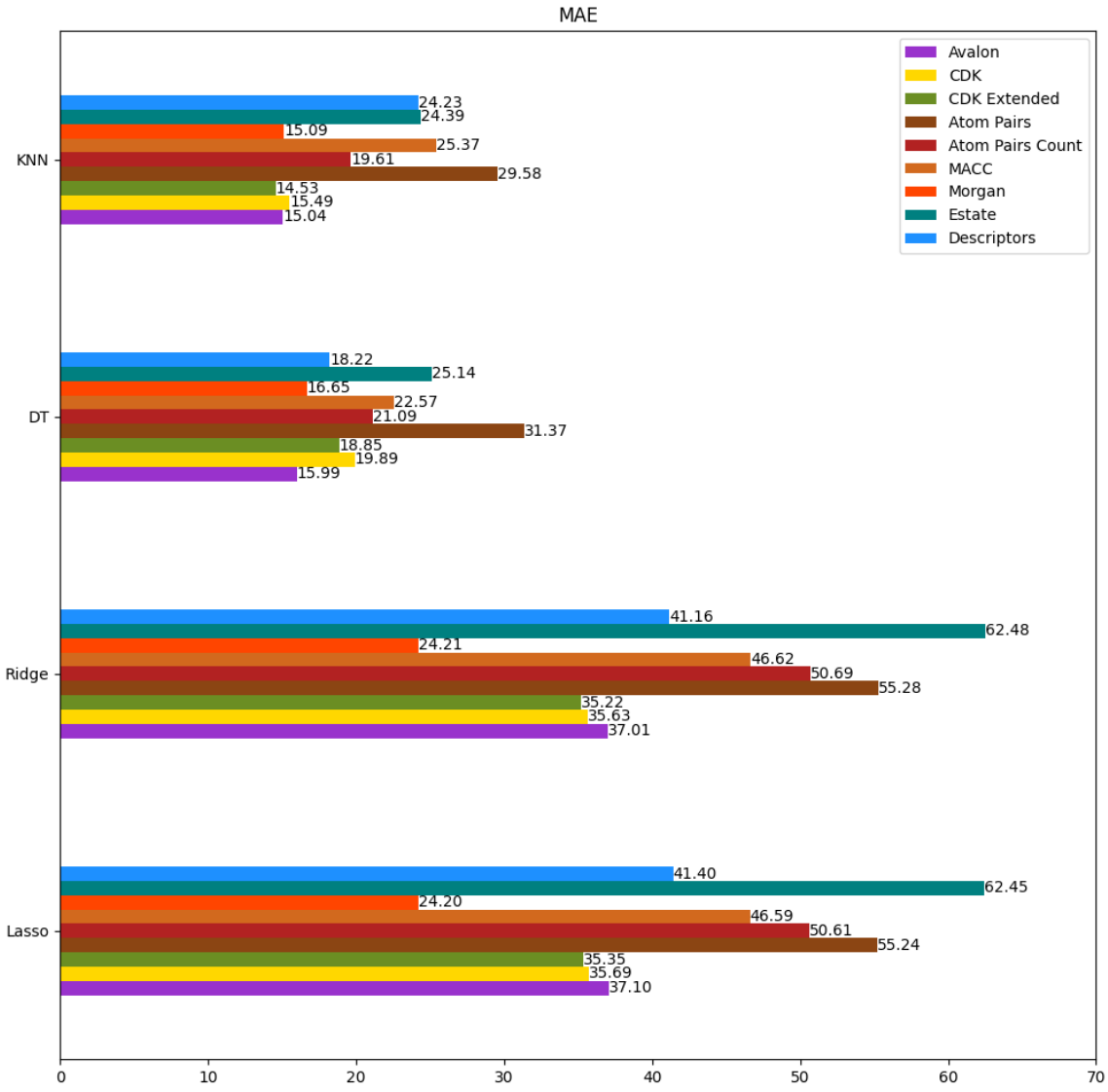

Figure 4-3: MAE scores of models, trained on original data, for *Maximum absorption wavelength*

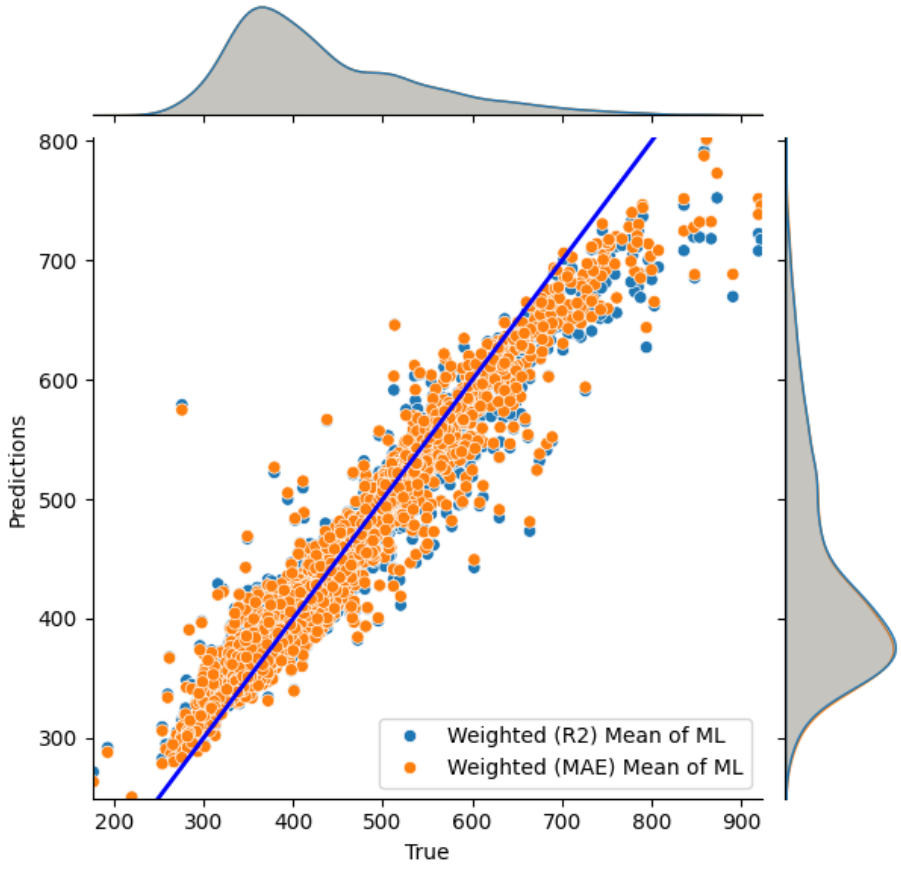

Figure 4-4: True and predicted (aggregated) with models, trained on original data, values of *Maximum absorption wavelength*

Figures 4-2 and 4-3 show R^2 and MAE scores of selected models for *Maximum absorption wavelength*. All models then are aggregated with weights corresponding to their validation results, and Figure 4-4 displays plot of true against aggregated predicted values. Aggregated predictions with MAE scores from the validation stage show $R^2 = 0.933, MAE = 18.104$, on the other hand, aggregated with R^2 show $R^2 = 0.9217, MAE = 20.015$.

As can be seen, all models with Morgan fingerprints as a feature space achieved the highest results of $R^2 \approx 0.9$. Overall, linear models are worse than KNN or DT, which might lead to the conclusion that this ML task is on non-linearity. The best MAE is achieved with KNN model with CDK-Extended fingerprints, $MAE = 14.53$. The best $R2$ is achieved with KNN model with Avalon fingerprints, $R^2 = 0.933$.

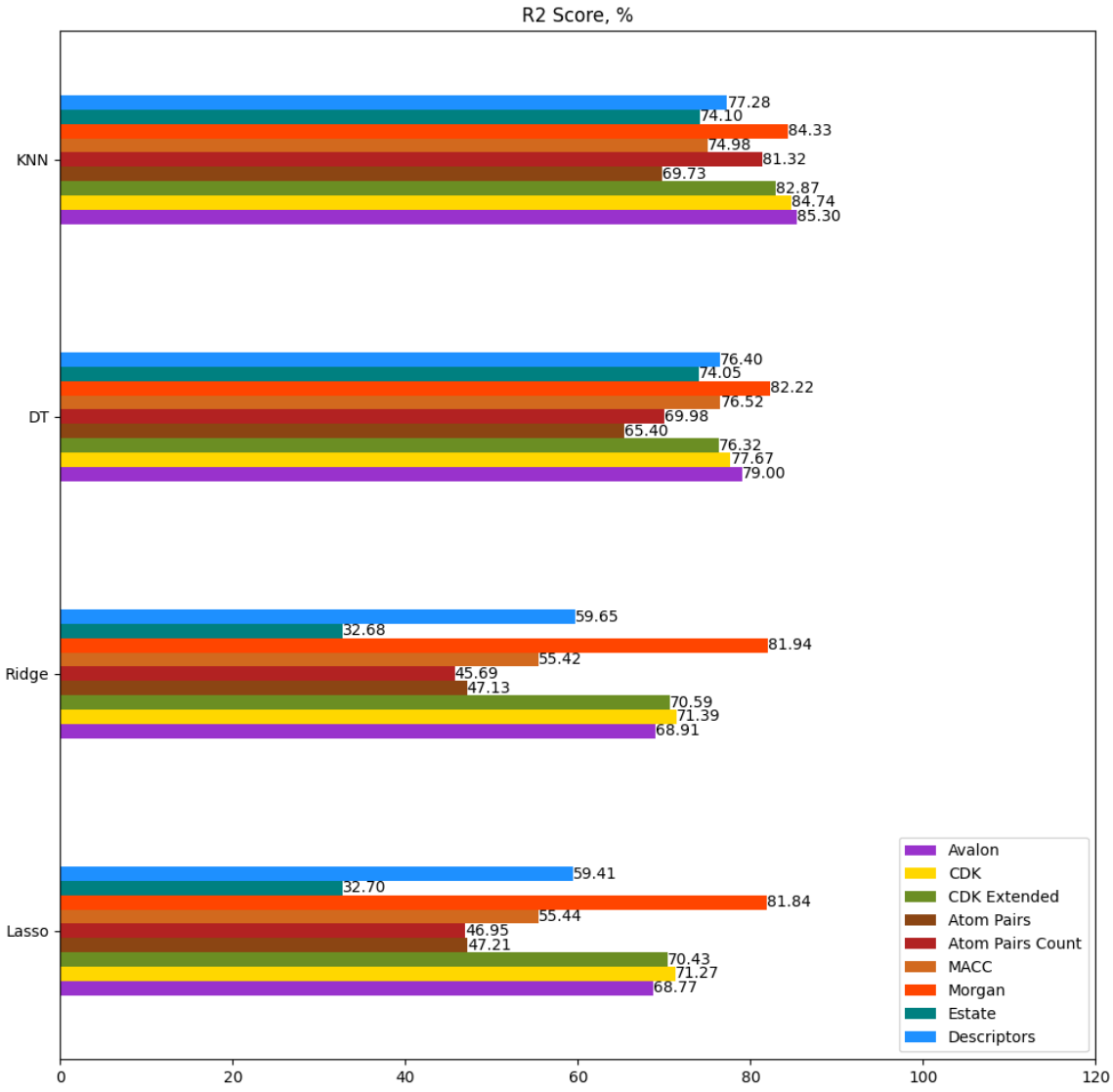

Figure 4-5: R^2 scores of models, trained on original data, for *Maximum emission wavelength*

Figures 4-5 and 4-6 show R^2 and MAE scores of selected models for *Maximum emission wavelength*.

All models then are aggregated with weights corresponding to the validation results, and Figure 4-7 displays plot of true against aggregated predicted values. Aggregated predictions with MAE scores from validation stage show $R^2 = 0.8756, MAE = 24.159$, on the other hand, aggregated with R^2 show $R^2 = 0.8556, MAE = 23.025$.

As can be seen, the results are quite similar to results of predicting *Maximum absorption wavelength*. With Morgan fingerprints as a feature space models achieved

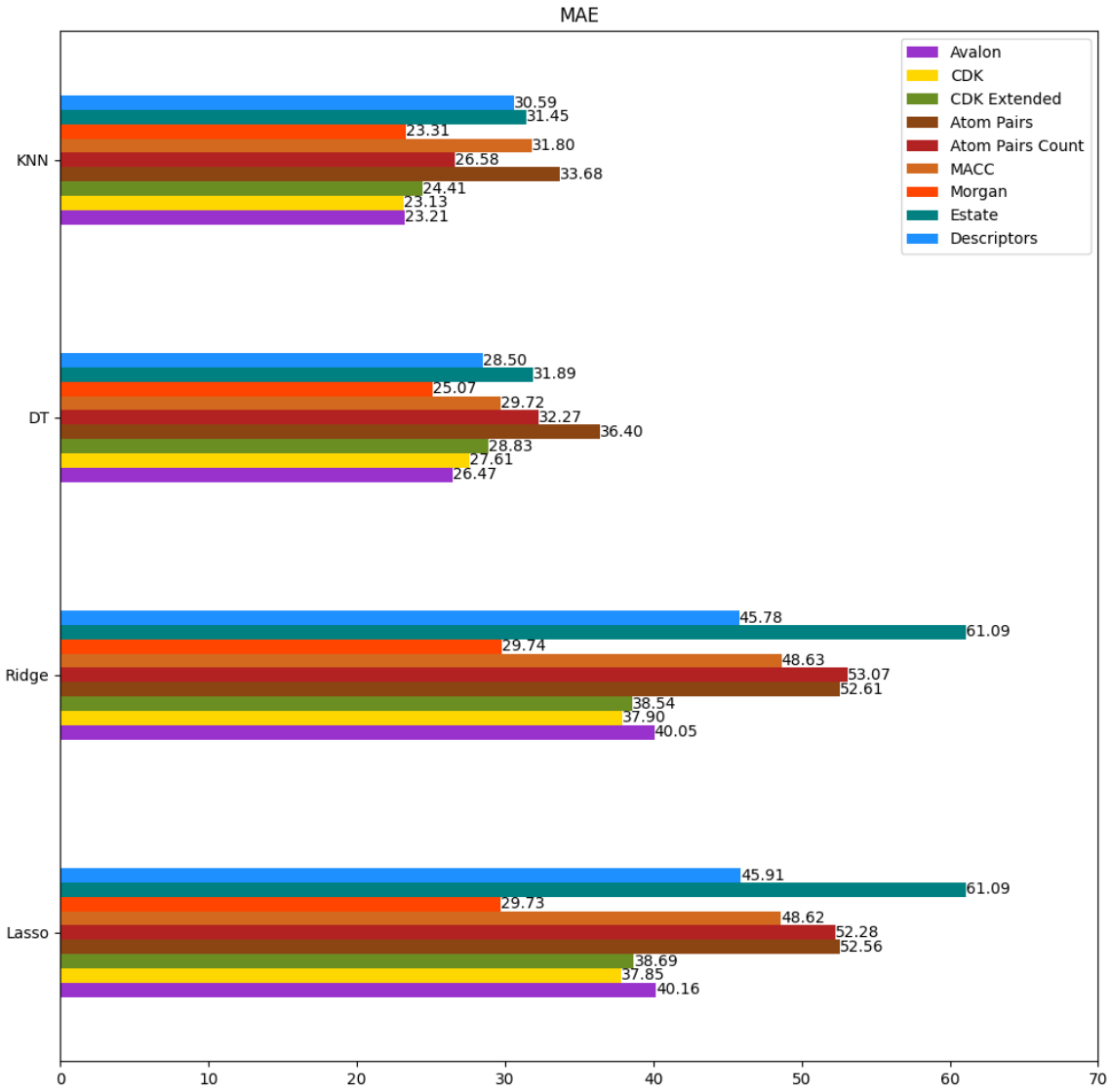

Figure 4-6: MAE scores of models, trained on original data, for *Maximum emission wavelength*

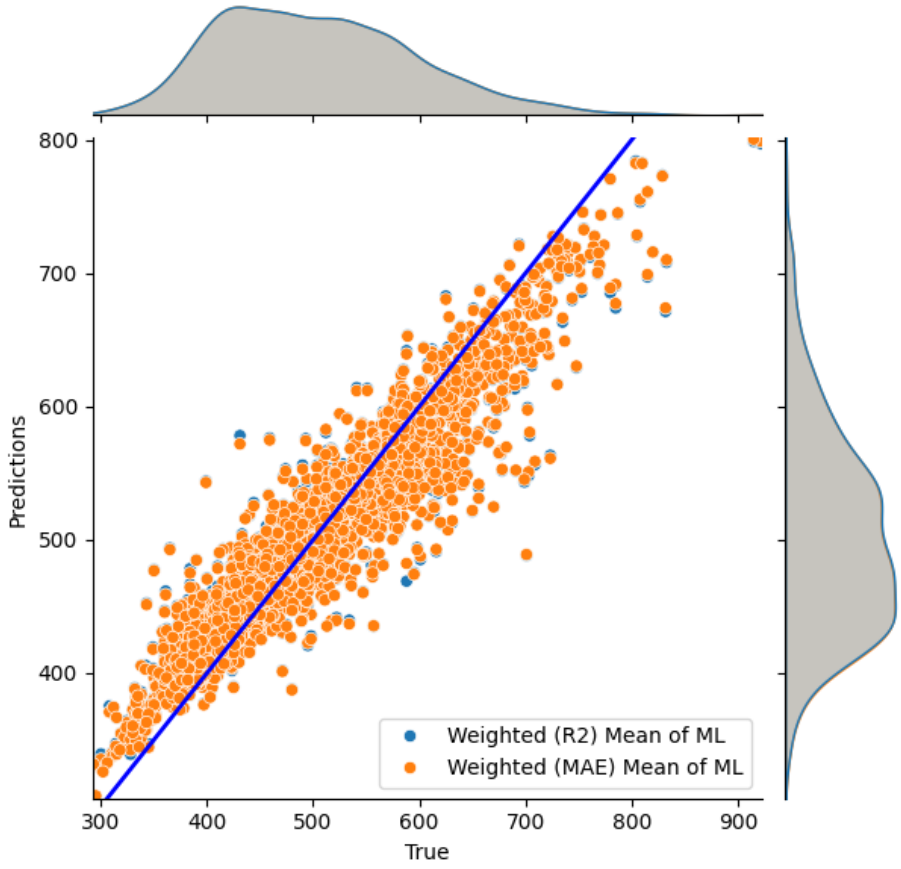

Figure 4-7: True and predicted (aggregated) with models, trained on original data, values of *Maximum emission wavelength*

$R^2 > 0.8, MAE < 30$. Avalon fingerprint as a feature and KNN model as a predictor achieved the best results among all models with $R^2 = 0.853, MAE = 23.21$. Nevertheless, as can be noticed, aggregated results are the leading overall.

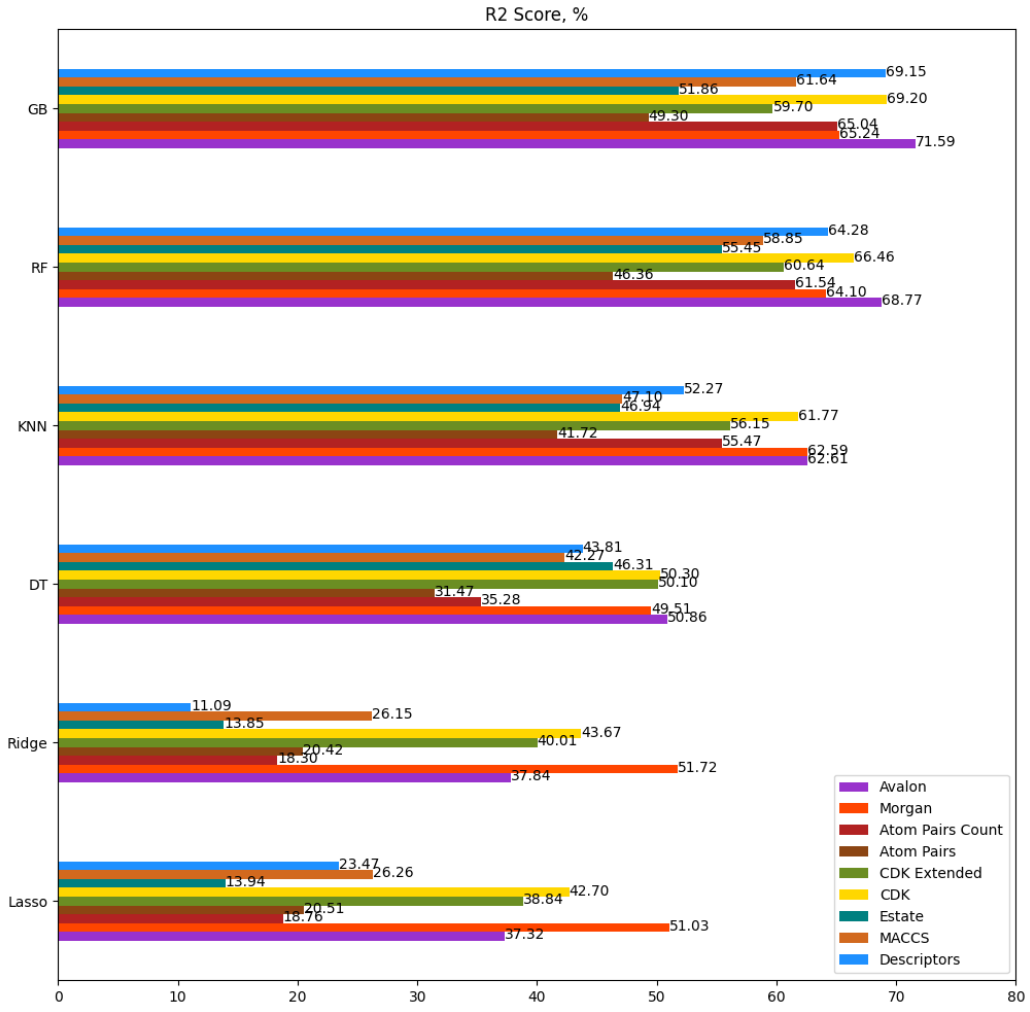

Figure 4-8: R^2 scores of models, trained on original data, for *Quantum yield*

Figures 4-8 and 4-9 show R^2 and MAE scores of selected models for *Quantum yield*.

All models then are aggregated with weights corresponding to the validation results, and Figure 4-10 displays plot of true against aggregated predicted values. Aggregated predictions with MAE scores from validation stage show $R^2 = 0.6491, MAE = 0.146$, on the other hand, aggregated with R^2 show $R^2 = 0.6653, MAE = 0.141$.

As can be seen, the results are worse than above stated two properties. That is why, ensemble models such as GB and RF are added to predictors. Avalon fingerprint as a feature space and GB as a predictor achieved the best results among all models with $R^2 = 0.7159, MAE = 0.12$.

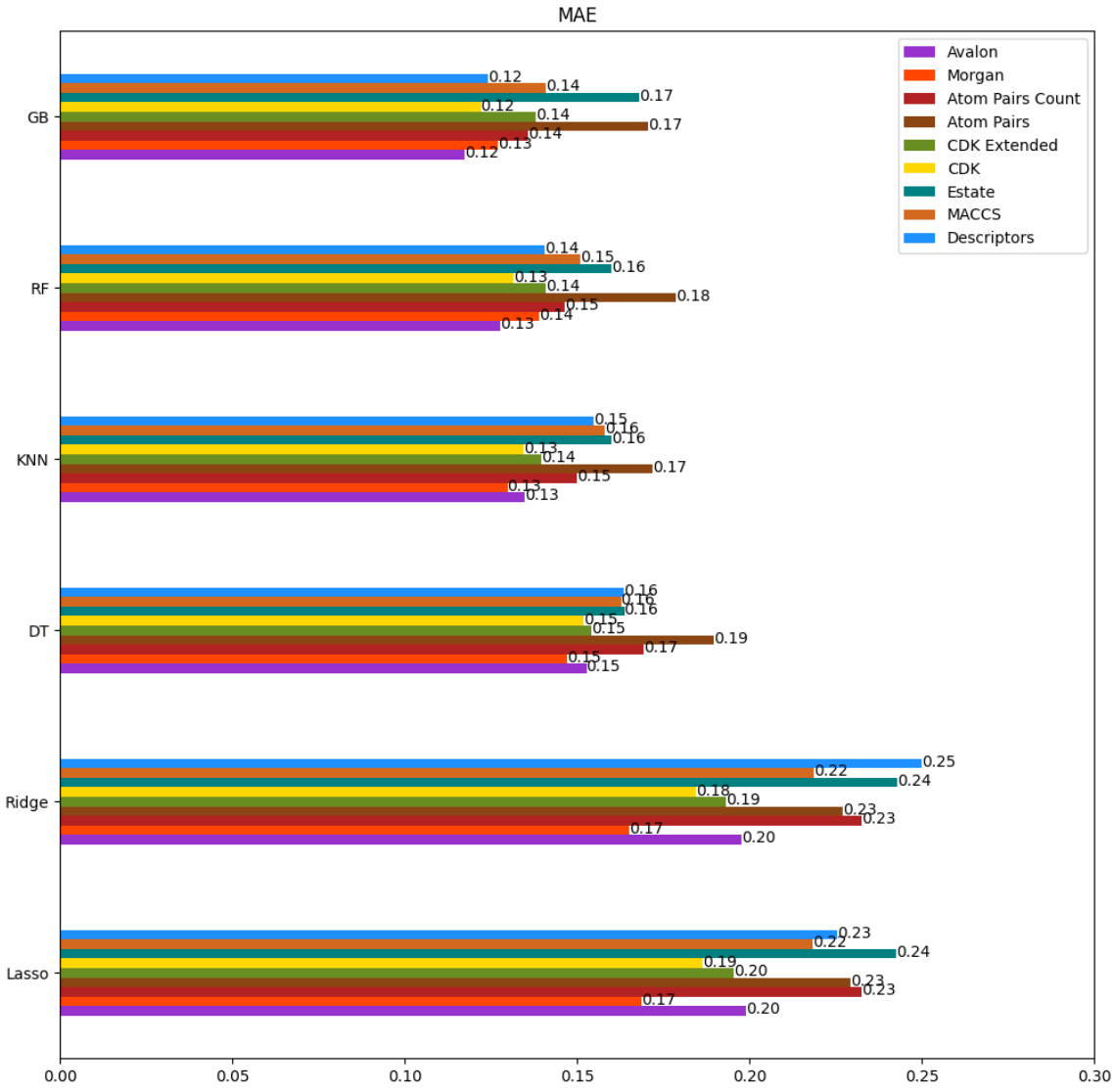

Figure 4-9: MAE scores of models, trained on original data, for *Quantum yield*

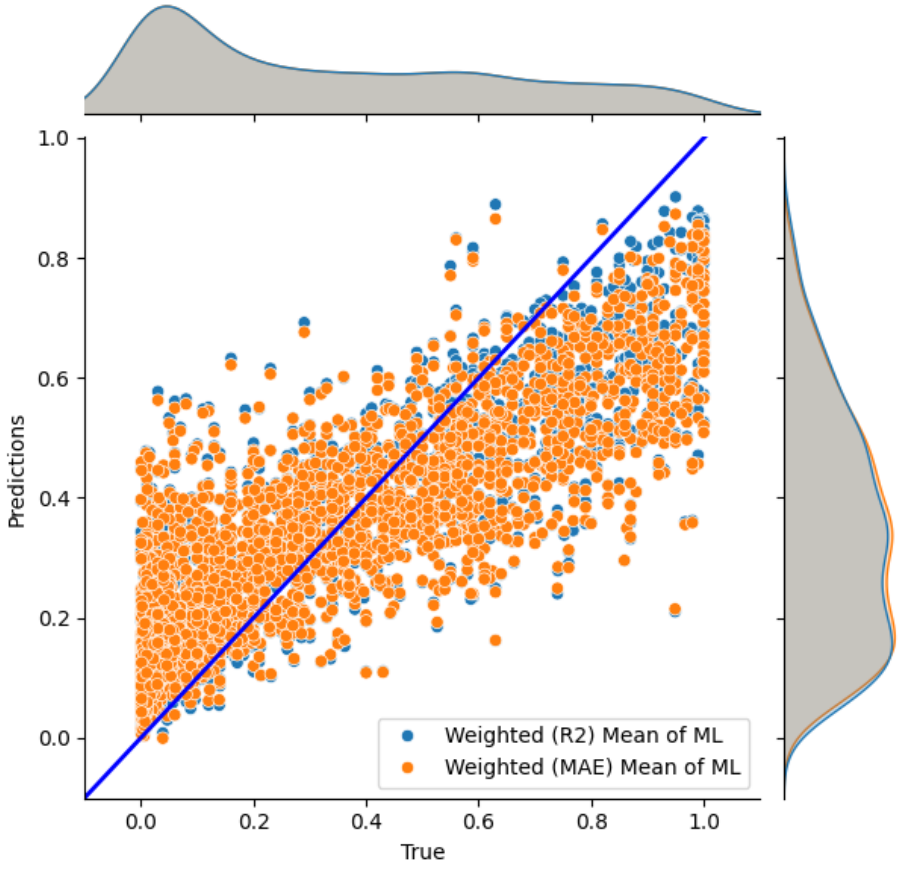

Figure 4-10: True and predicted (aggregated) with models, trained on original data, values of *Quantum yield*

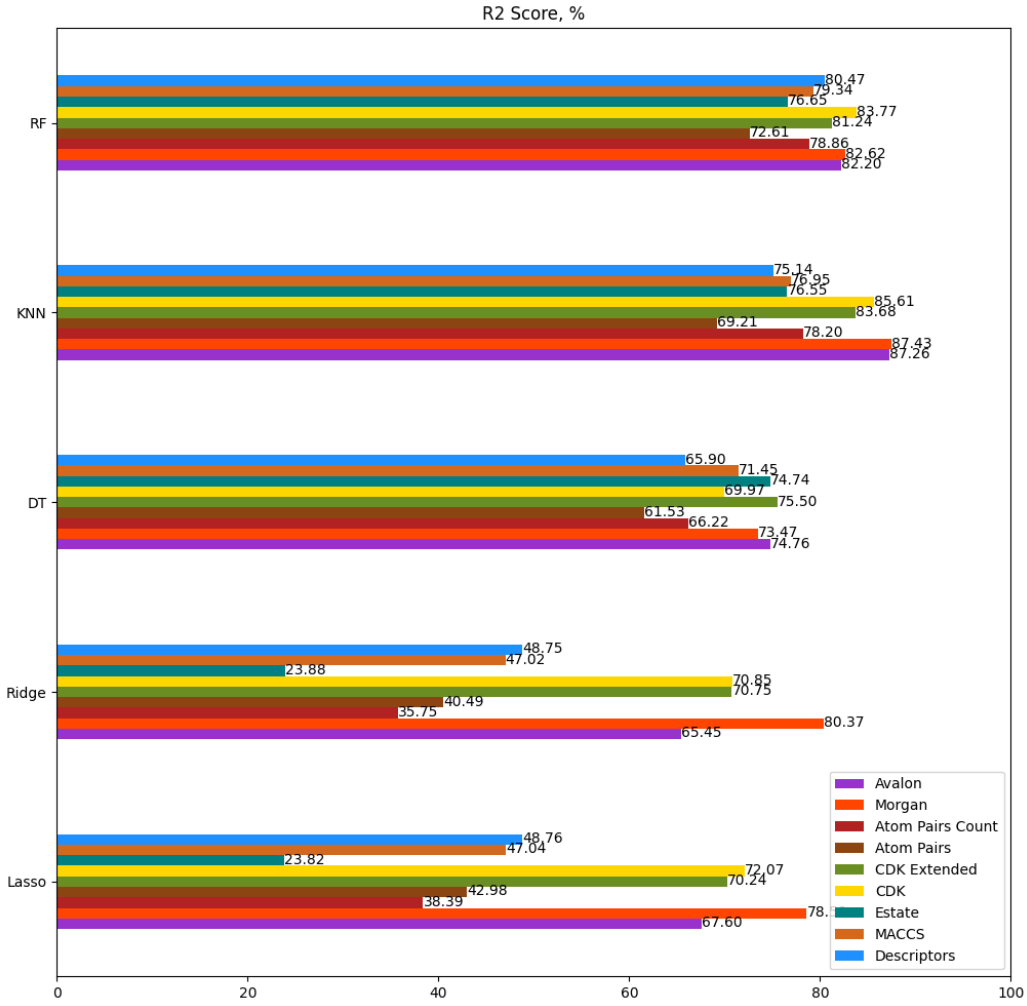

Figure 4-11: R^2 scores of models, trained on original data, for *Extinction coefficient*

Figures 4-11 and 4-12 show R^2 and MAE scores of selected models for *Extinction coefficient*.

All models then are aggregated with weights corresponding to the validation results, and Figure 4-13 displays plot of true against aggregated predicted values. Aggregated predictions with MAE scores from validation stage show $R^2 = 0.8382, MAE = 0.1505$, on the other hand, aggregated with R^2 show $R^2 = 0.8381, MAE = 0.1503$.

As can be seen, Morgan or Avalon fingerprints as feature space produced best results among any other descriptors. Overall, Morgan fingerprint as feature space and KNN model as predictor gave the highest results: $R^2 = 0.8743, MAE = 0.13$.

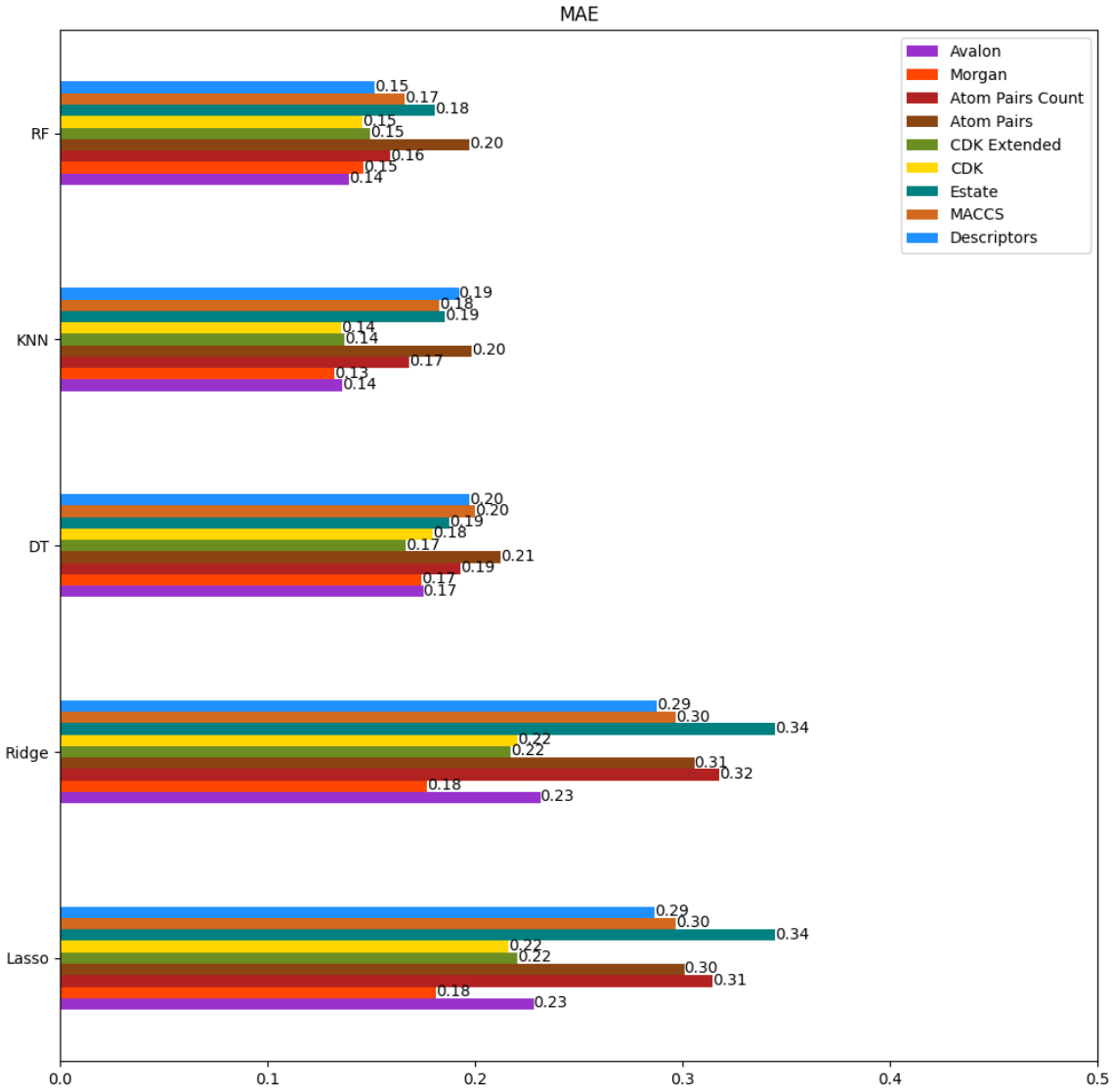

Figure 4-12: MAE scores of models, trained on original data, for *Extinction coefficient*

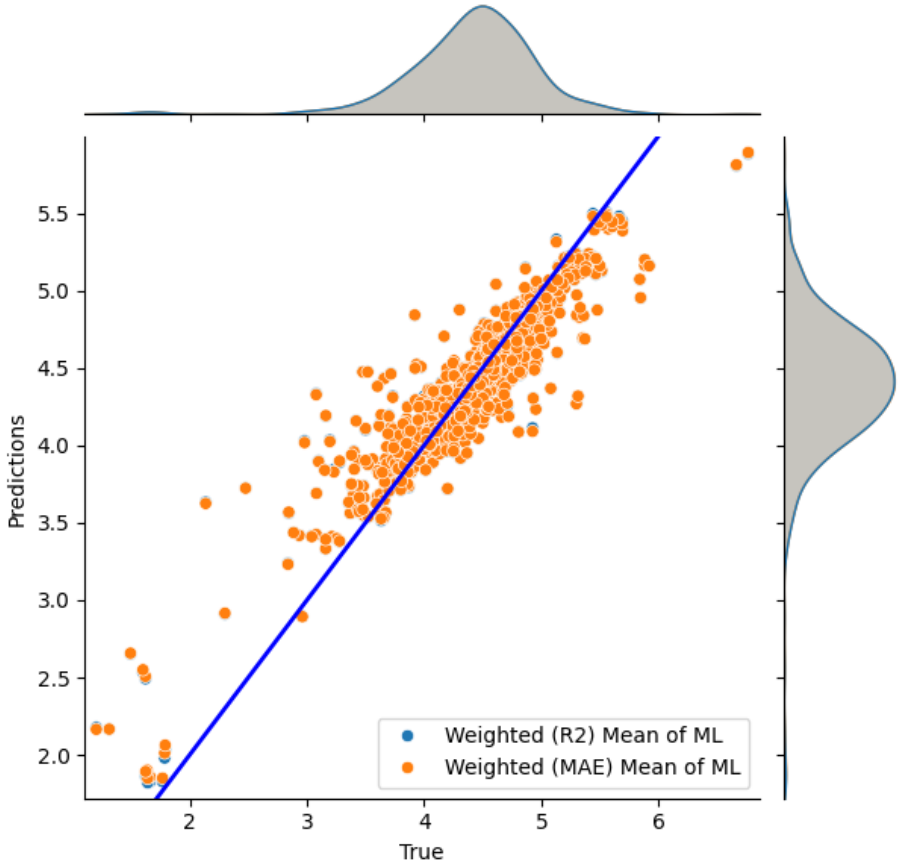

Figure 4-13: True and predicted (aggregated) with models, trained on original data, values of *Extinction coefficient*

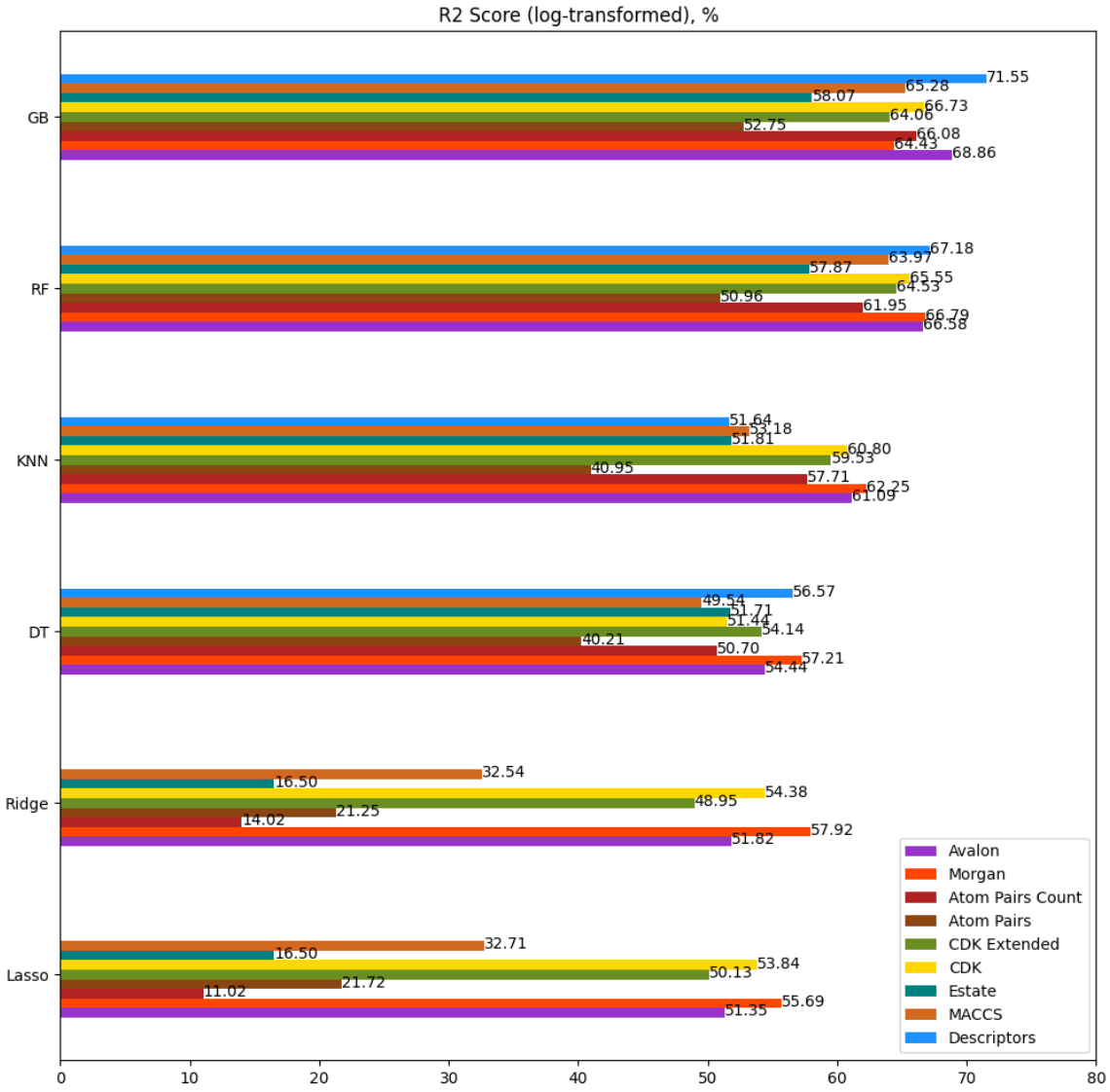

Figure 4-14: R^2 scores of models, trained on original data, for *Lifetime* (log-transformed)

Figures 4-14 and 4-15 show R^2 and MAE scores of selected models for *Lifetime*.

All models then are aggregated with weights corresponding to the validation results, and Figure 4-16 displays plot of true against aggregated predicted values. Aggregated predictions with MAE scores from validation stage show $R^2 = 0.6768, MAE = 0.5231$, on the other hand, aggregated with R^2 show $R^2 = 0.6831, MAE = 0.5147$.

Similar to *Quantum Yield*, predictions of *Lifetime* are average. Some linear models results are not acceptable at all, for example Ridge and Lasso models with descriptors as feature space produced negative R^2. That is why ensemble models are utilized.

The best obtained result correspond to GB model with Avalon fingerprints with $R^2 = 0.7155, MAE = 0.46$.

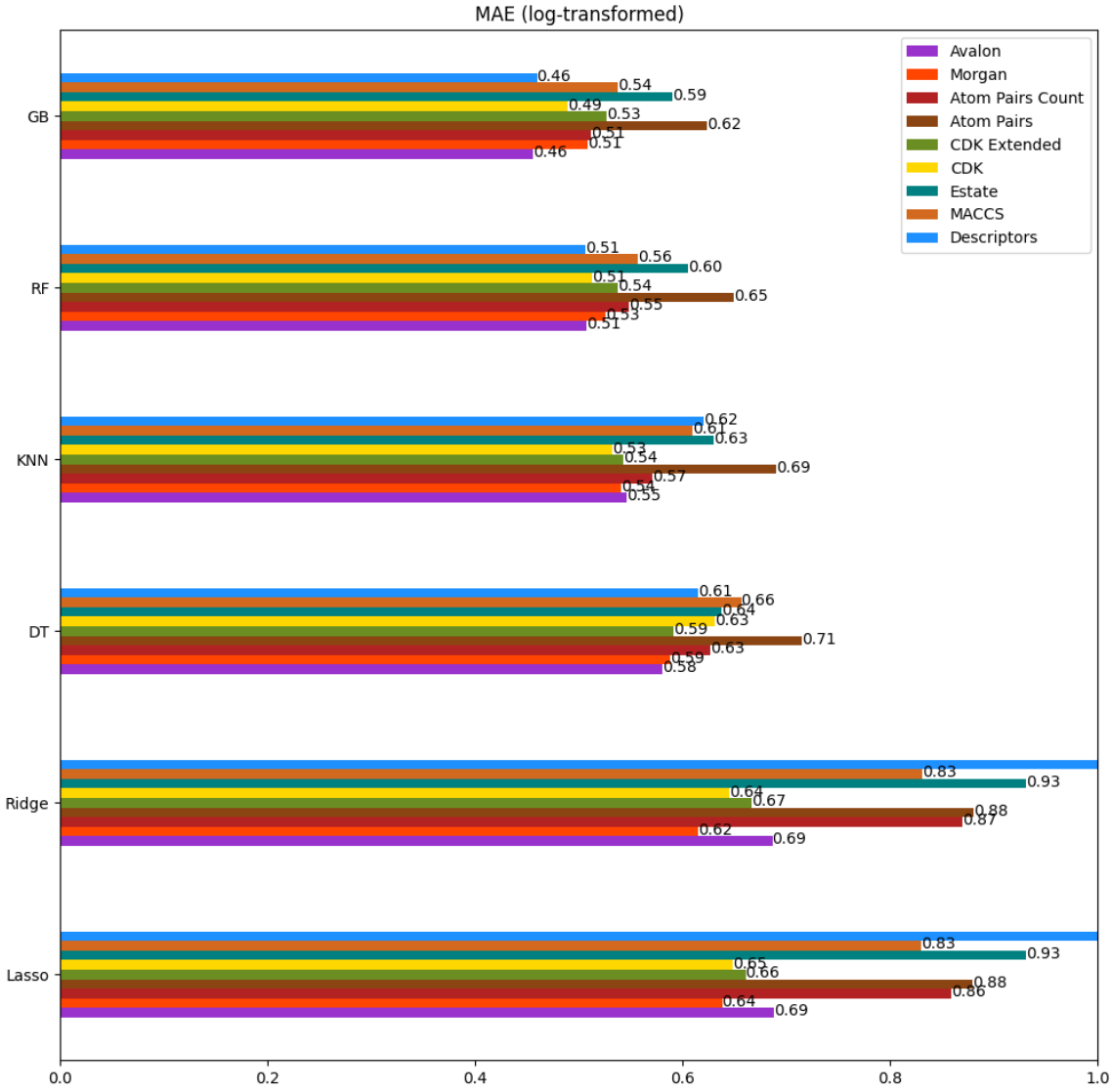

Figure 4-15: MAE scores of models, trained on original data, for *Lifetime* (log-transformed)

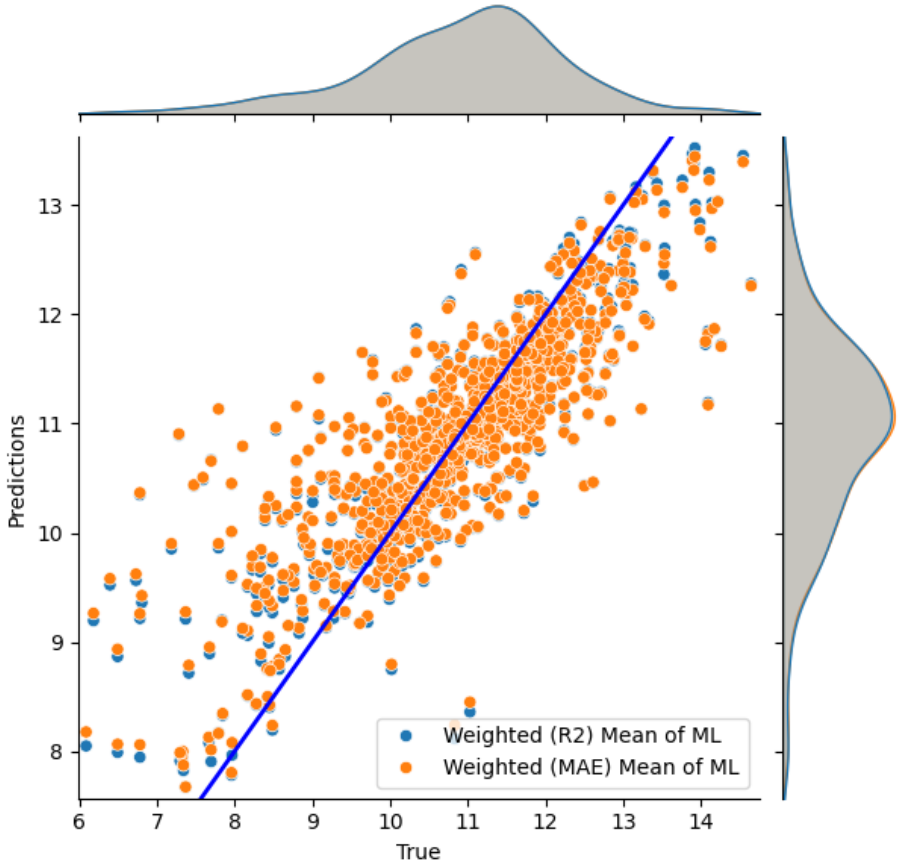

Figure 4-16: True and predicted (aggregated) with models, trained on original data, values of *Lifetime* (log-transformed)

4.4.2 Evaluating models trained on imputed data with feature imputation

Original data has missing values, and thus, it is imputed and predictive model is built with two options. The first option is feature imputation. This option means that there is no imputed value for target variable, however there are imputed features: other target values that are used in training, forecasting target property. Following figures show the results of evaluating models on each test set corresponding to 5 target values. To be fair at judgements the test and validation sets are kept the same as for the original data.

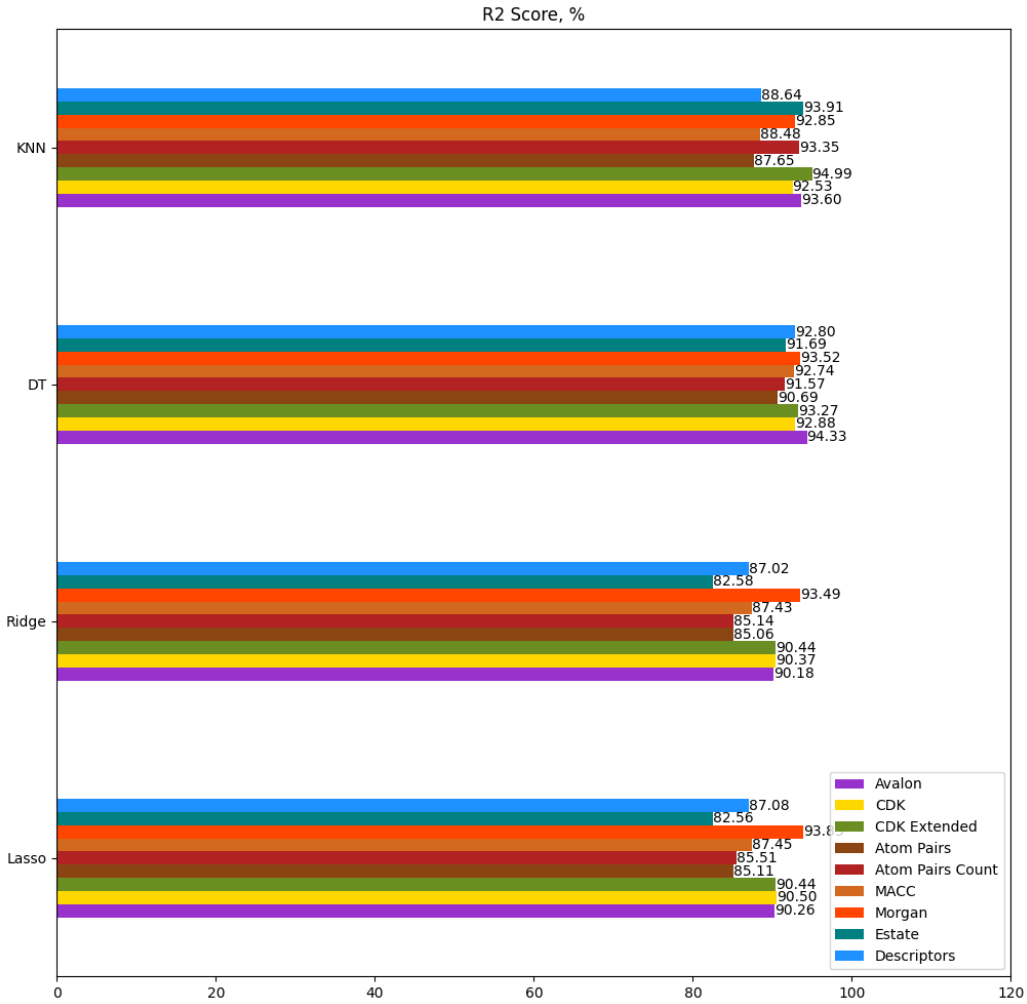

Figure 4-17: R^2 scores of models, trained on imputed data (feature imputation), for *Maximum absorption wavelength*

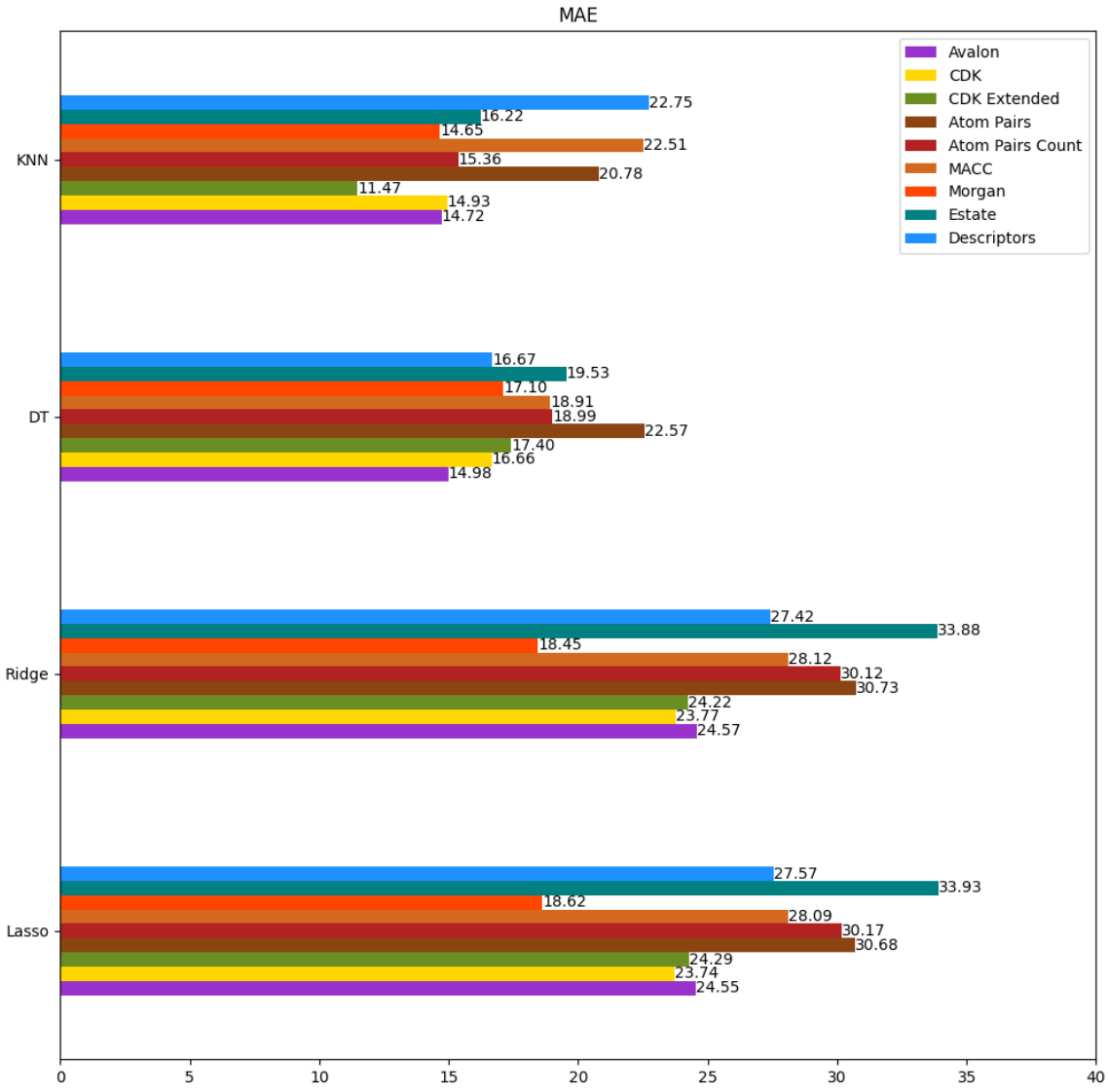

Figure 4-18: MAE scores of models, trained on imputed data (feature imputation), for *Maximum absorption wavelength*

Figures 4-17 and 4-18 show R^2 and MAE scores of selected models for *Maximum absorption wavelength*.

All models then are aggregated with weights corresponding to the validation results, and Figure 4-19 displays plot of true against aggregated predicted values. Aggregated predictions with MAE scores from validation stage show $R^2 = 0.9605, MAE = 14.57$, on the other hand, aggregated with R^2 show $R^2 = 0.9564, MAE = 15.498$.

As can be seen all selected models achieved acceptable results on test with $R^2 > 0.8$. Decision trees with all type of molecular representations achieved the highest results. The best result correspond to KNN model as predictor and CDK extended fingerprint with $MAE = 11.47$ in terms of MAE, and aggregated among all predictions with MAE scores with $R^2 = 0.9605$.

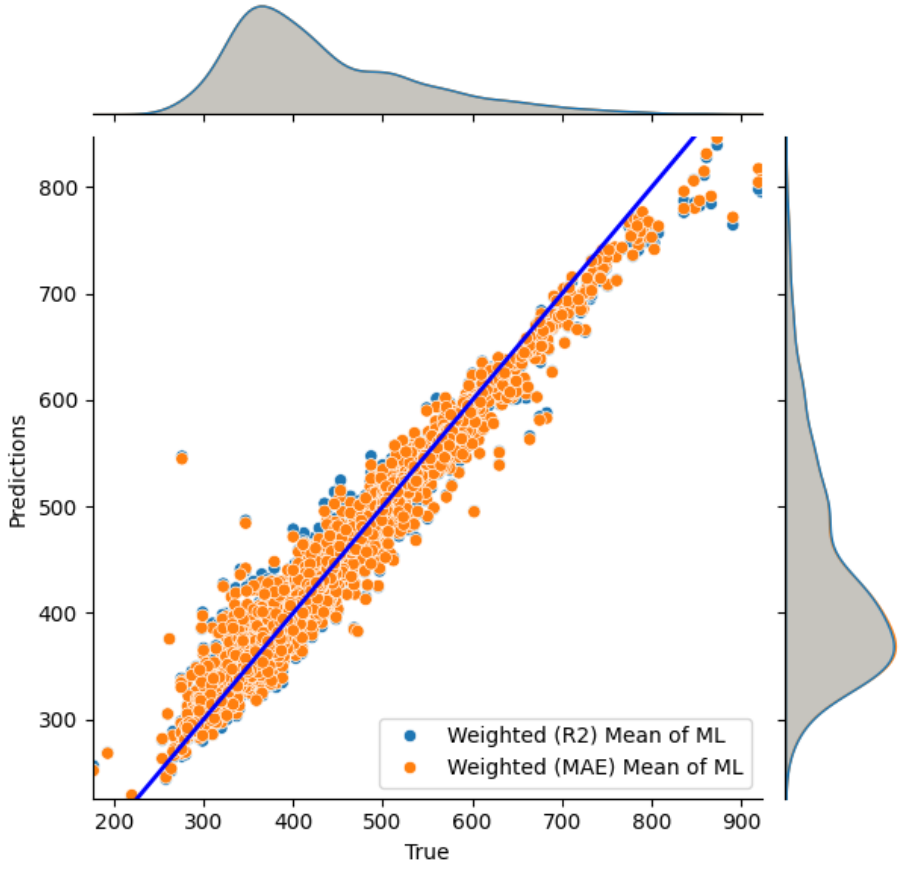

Figure 4-19: True and predicted (aggregated) with models, trained on imputed data (feature imputation), values of *Maximum absorption wavelength*

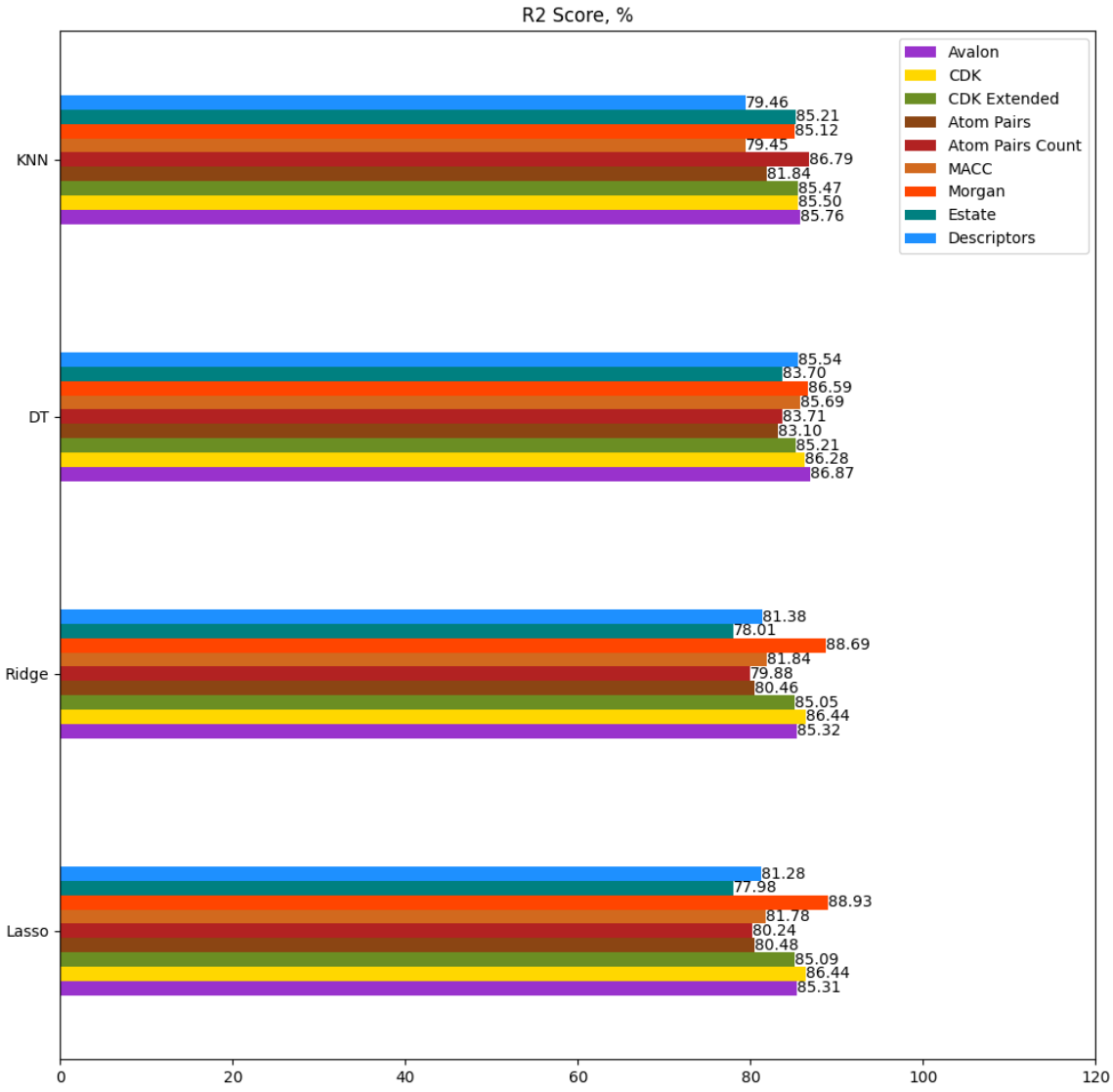

Figure 4-20: R^2 scores of models, trained on imputed data (feature imputation), for *Maximum emission wavelength*

Figures 4-20 and 4-21 show R^2 and MAE scores of selected models for *Maximum emission wavelength*.

All models then are aggregated with weights corresponding to the validation results, and Figure 4-22 displays plot of true against aggregated predicted values. Aggregated predictions with MAE scores from validation stage show $R^2 = 0.9173, MAE = 19.25$, on the other hand, aggregated with R^2 show $R^2 = 0.9156, MAE = 19.485$.

Similar *Maximum absorption wavelength*, adding properties positively affected results of predictions. All models achieved $R^2 \approx 0.8$. The best results correspond to

aggregated prediction with MAE scores with $R^2 = 0.9173, MAE = 19.25$.

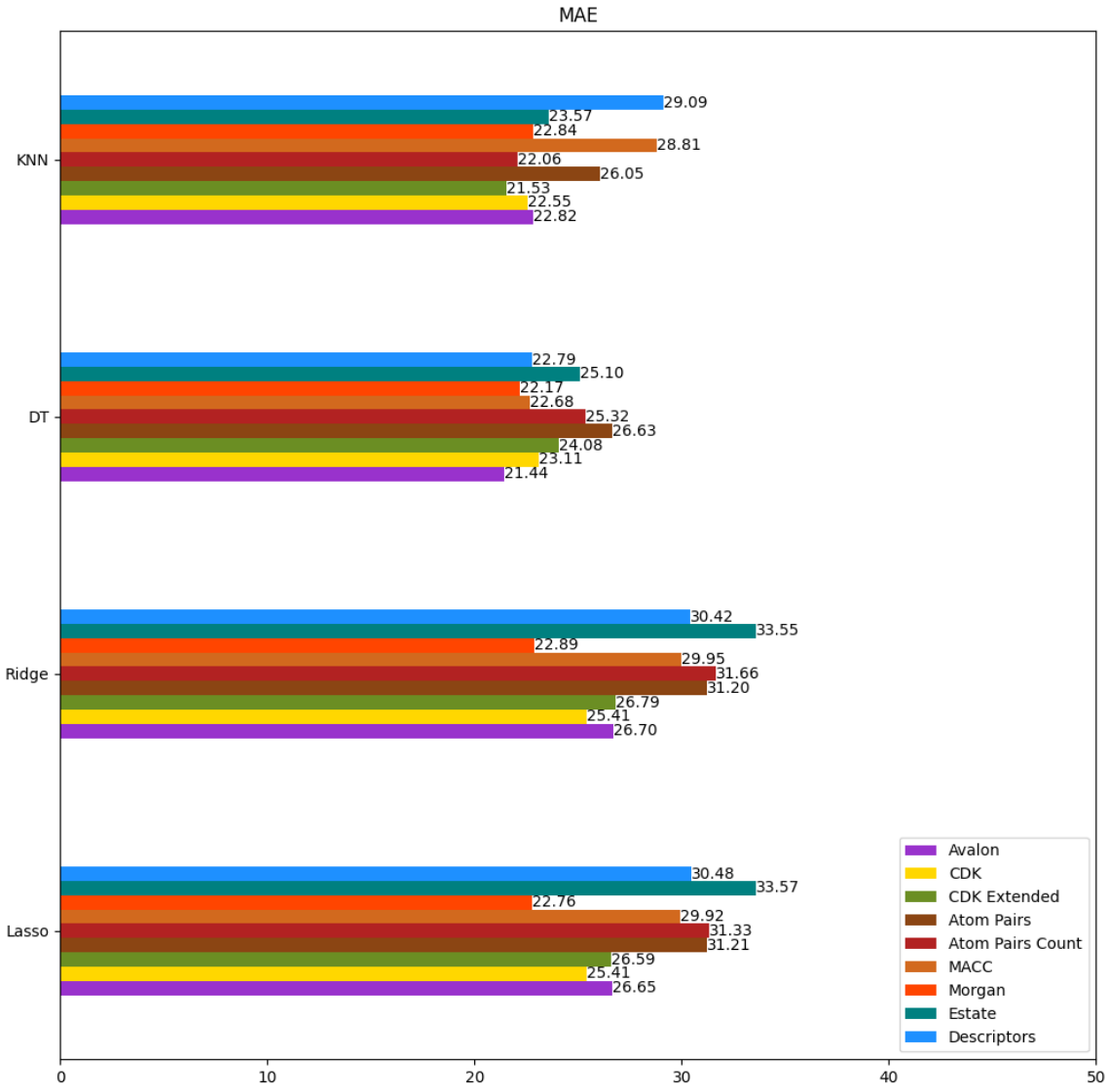

Figure 4-21: MAE scores of models, trained on imputed data (feature imputation), for *Maximum emission wavelength*

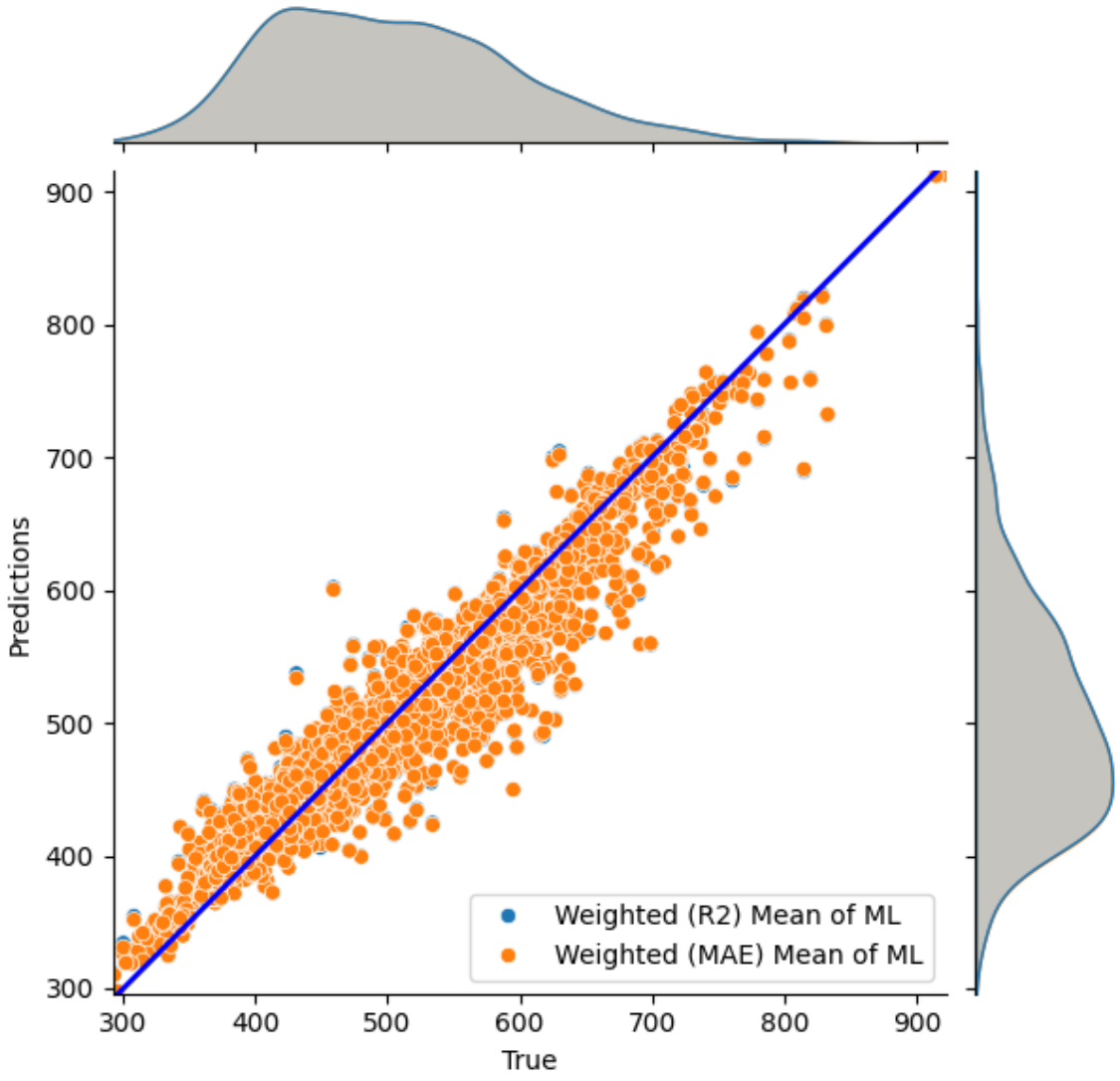

Figure 4-22: True and predicted (aggregated) with models, trained on imputed data (feature imputation), values of *Maximum emission wavelength*

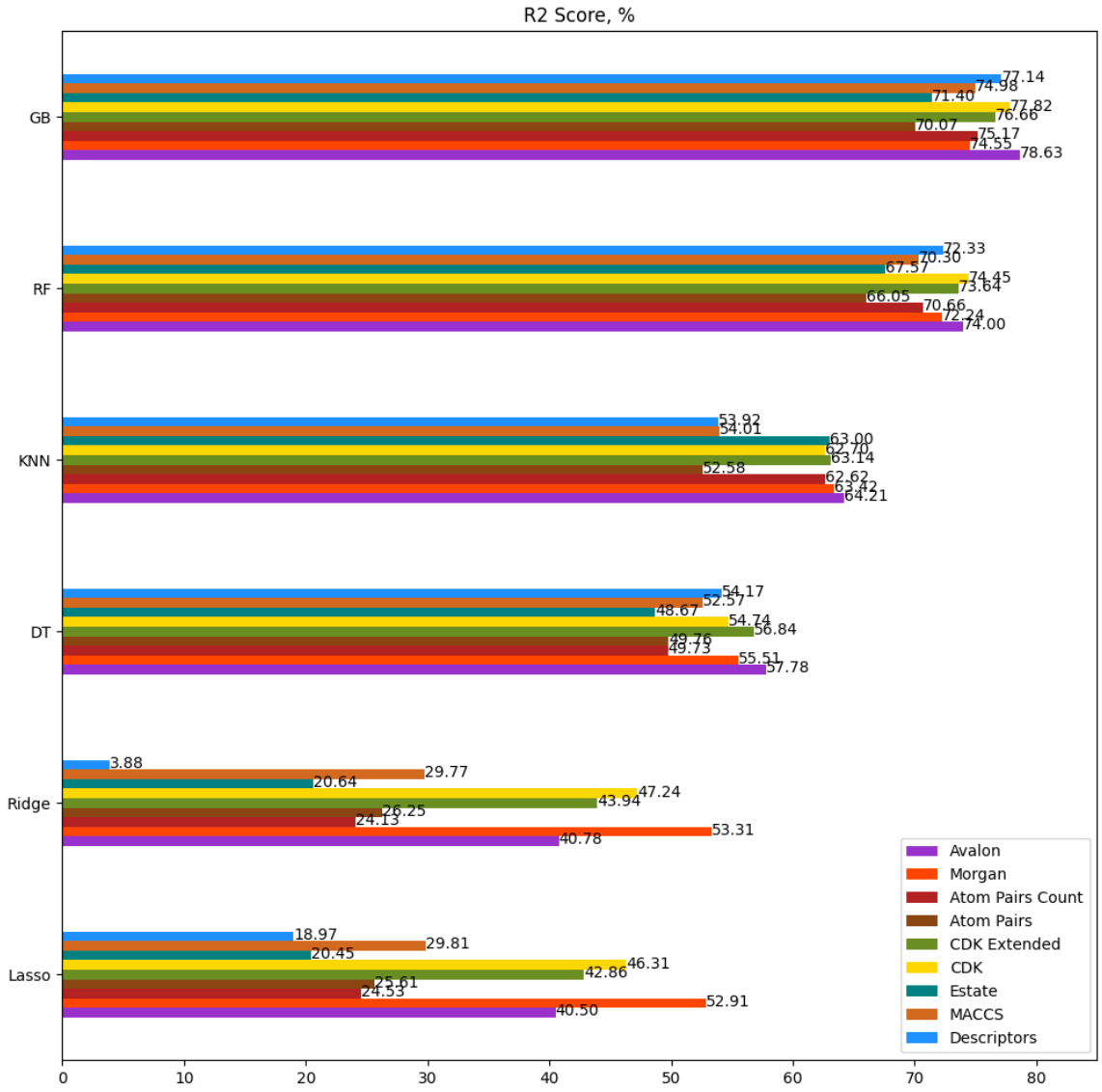

Figure 4-23: R^2 scores of models, trained on imputed data (feature imputation), for *Quantum yield*

Figures 4-23 and 4-24 show R^2 and MAE scores of selected models for *Quantum yield*.

All models then are aggregated with weights corresponding to the validation results, and Figure 4-25 displays plot of true against aggregated predicted values. Aggregated predictions with MAE scores from validation stage show $R^2 = 0.7341, MAE = 0.1225$, on the other hand, aggregated with R^2 show $R^2 = 0.7444, MAE = 0.119$.

Similar to above stated target properties, predictions of *Quantum Yield* are positively affected by adding other target properties to feature space. The best result

correspond to GB model with Avalon fingerprints with $R^2 = 0.7863, MAE = 0.1$.

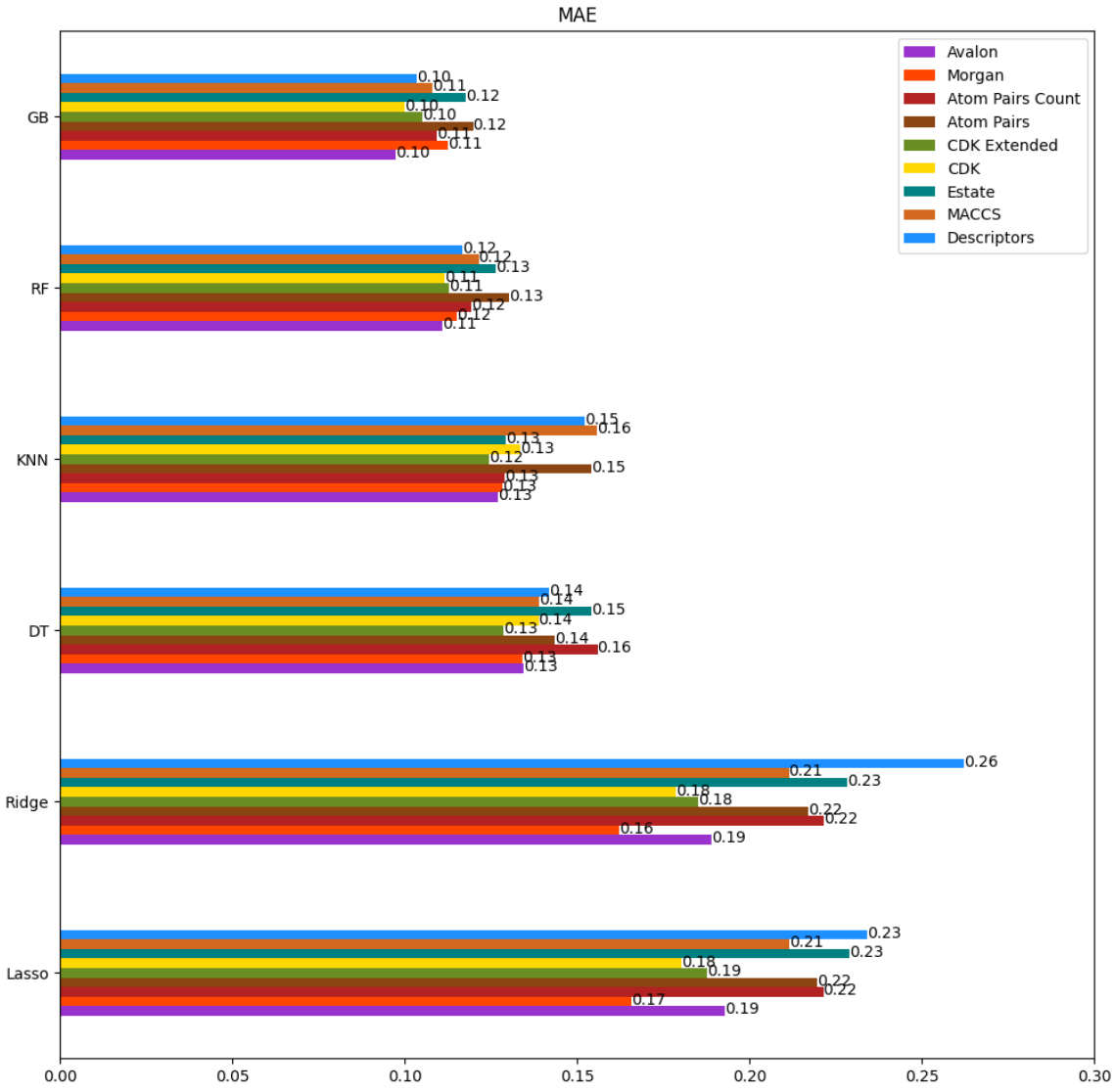

Figure 4-24: MAE scores of models, trained on imputed data (feature imputation), for *Quantum yield*

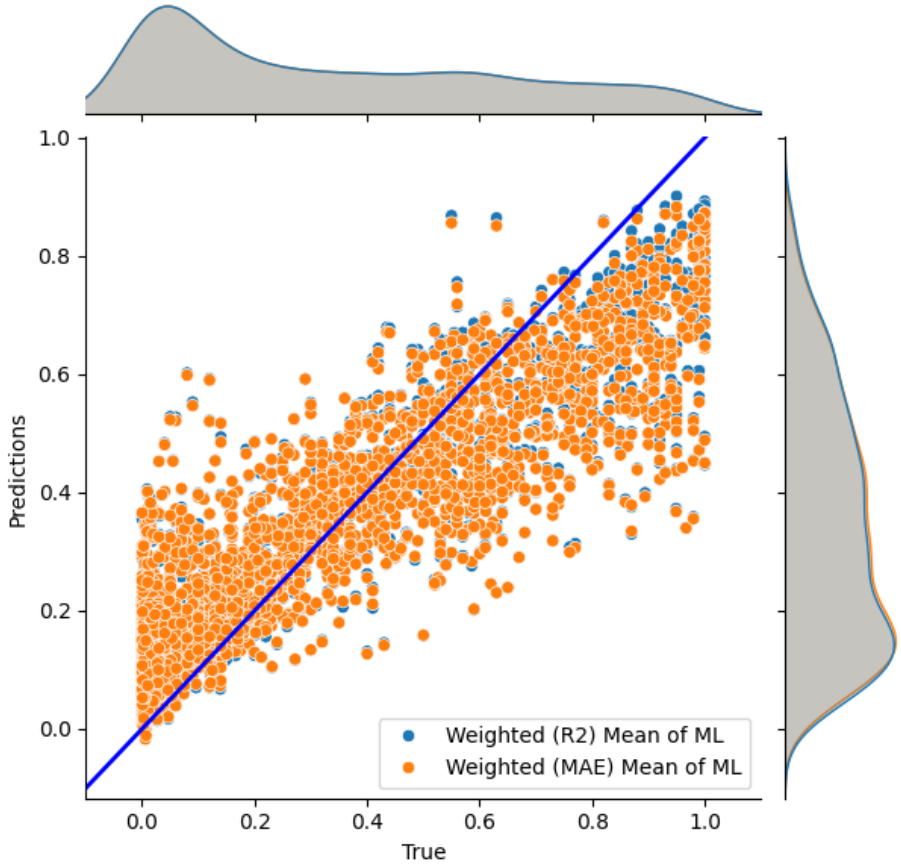

Figure 4-25: True and predicted (aggregated) with models, trained on imputed data (feature imputation), values of *Quantum yield*

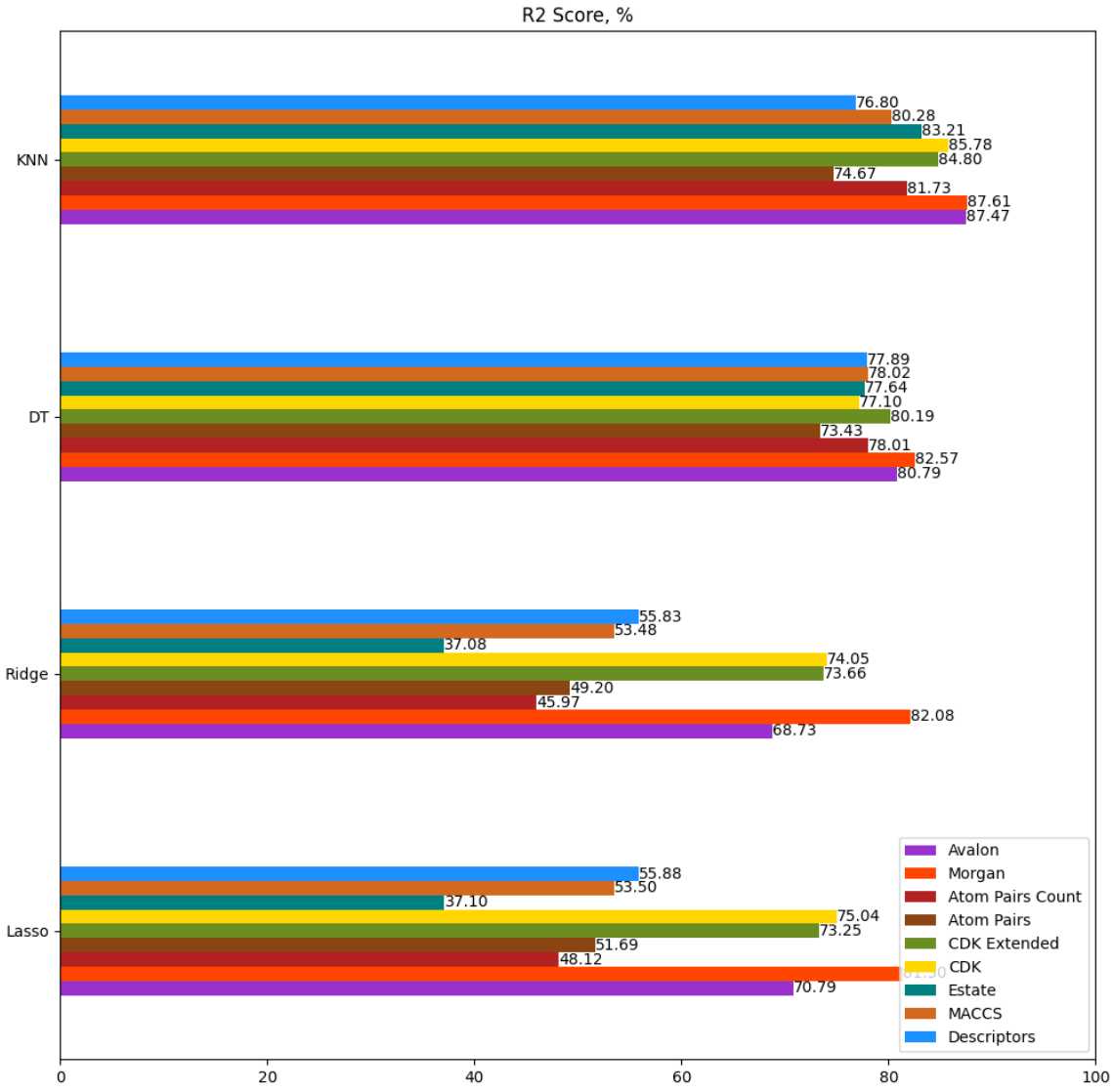

Figure 4-26: R^2 scores of models, trained on imputed data (feature imputation), for *Extinction coefficient*

Figures 4-26 and 4-27 show R^2 and MAE scores of selected models for *Extinction coefficient*.

All models then are aggregated with weights corresponding to the validation results, and Figure 4-28 displays plot of true against aggregated predicted values. Aggregated predictions with MAE scores from validation stage show $R^2 = 0.8636, MAE = 0.1417$, on the other hand, aggregated with R^2 show $R^2 = 0.8624, MAE = 0.1423$.

Although feature imputation positively affected the regression results, not all models has achieved acceptable results, mostly DT and KNN with any kind of molecular

representations is able to achieve $R^2 \approx 0.8$. The vest result correspond to KNN model and Morgan fingerprint with $R^2 = 0.8761, MAE = 0.13$.

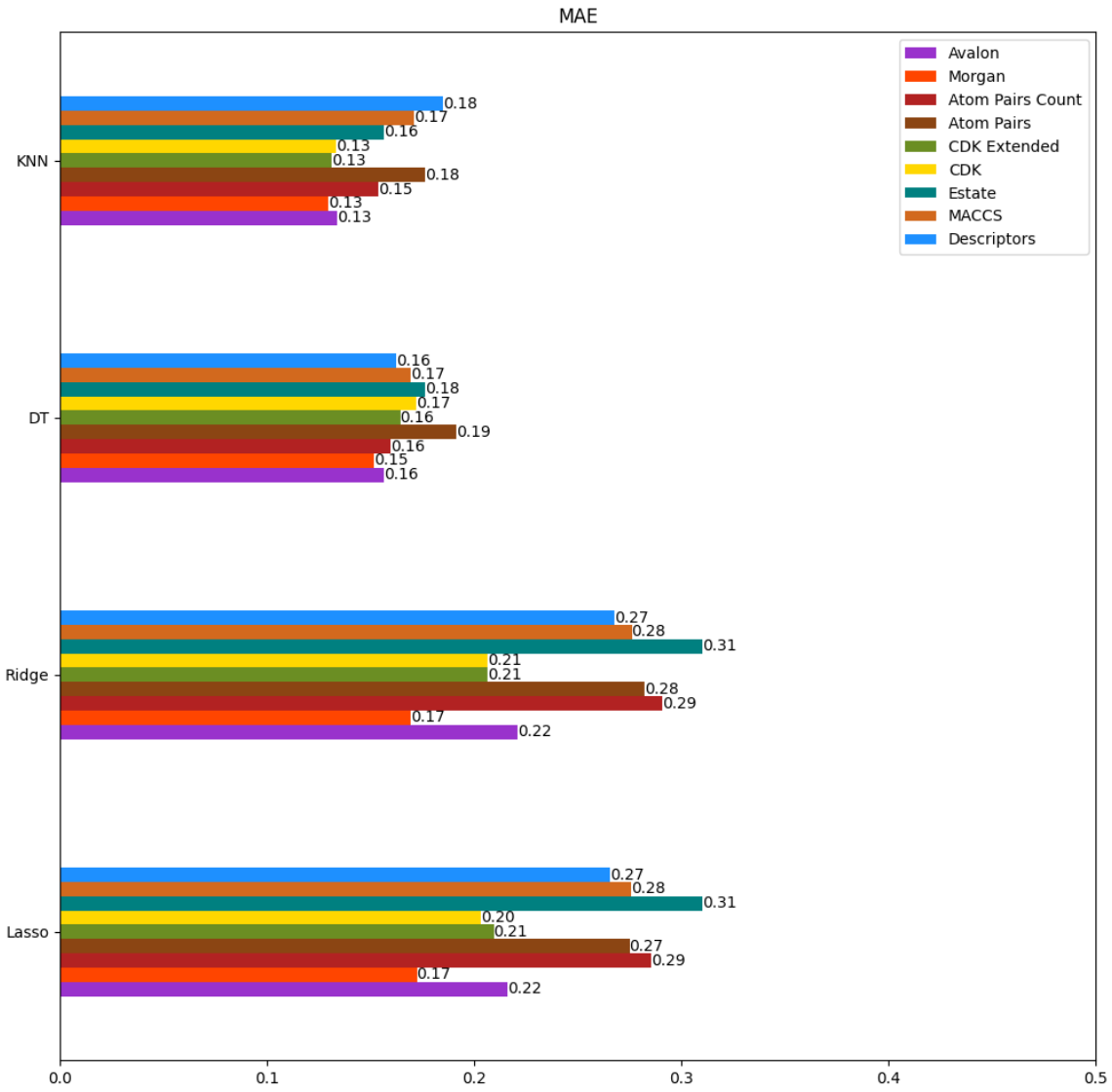

Figure 4-27: MAE scores of models, trained on imputed data (feature imputation), for *Extinction coefficient*

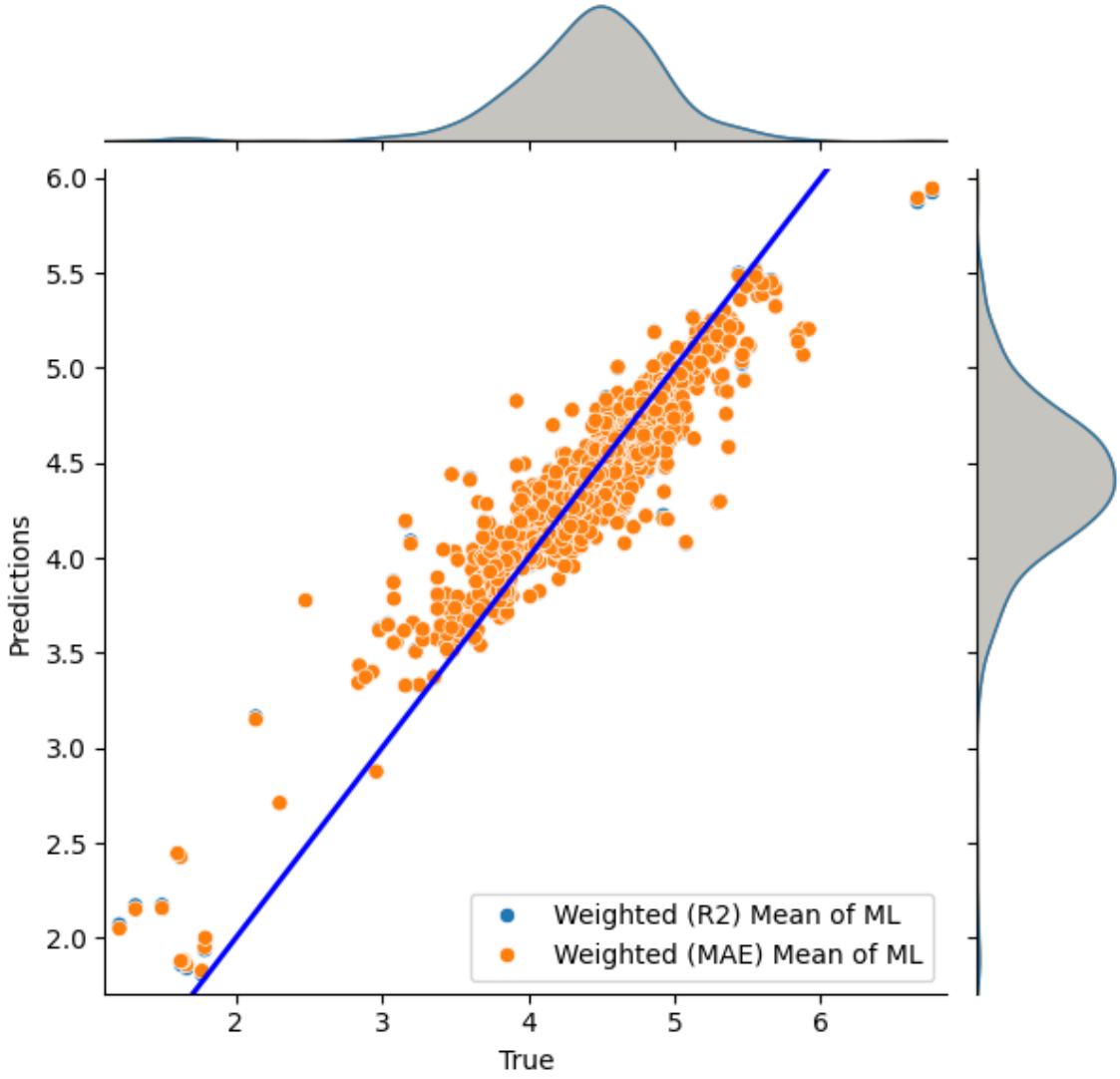

Figure 4-28: True and predicted (aggregated) with models, trained on imputed data (feature imputation), values of *Extinction coefficient*

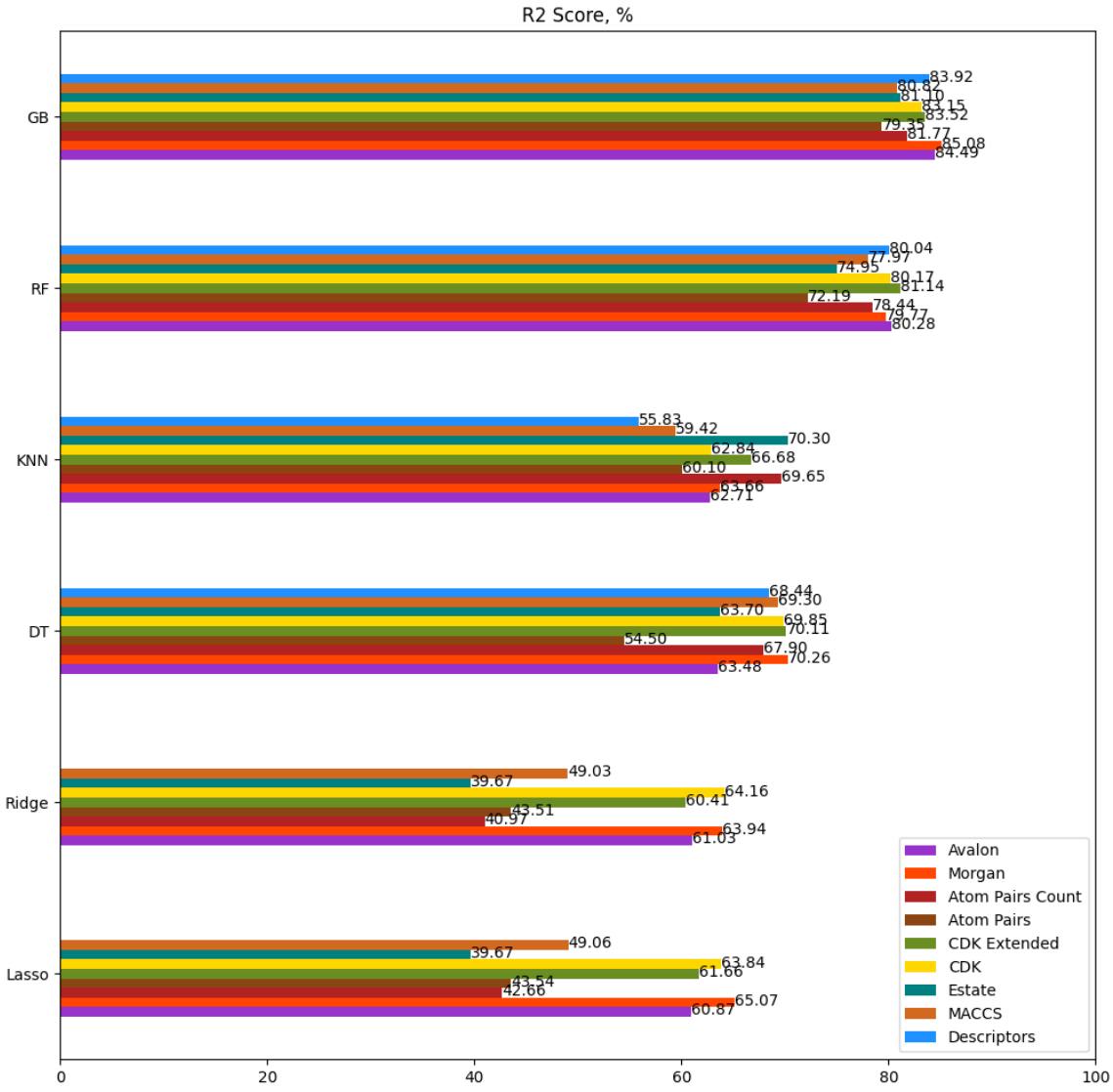

Figure 4-29: R^2 scores of models, trained on imputed data (feature imputation), for *Lifetime* (log-transformed)

Figures 4-29 and 4-30 show R^2 and MAE scores of selected models for *Lifetime*.

All models then are aggregated with weights corresponding to the validation results, and Figure 4-31 displays plot of true against aggregated predicted values. Aggregated predictions with MAE scores from validation stage show $R^2 = 0.815, MAE = 0.385$, on the other hand, aggregated with R^2 show $R^2 = 0.812, MAE = 0.389$.

Feature imputation positively affected predictions, however, some linear models produced non-satisfactory results. Nevertheless, now ensemble methods are able to achieve $R^2 \approx 0.8, MAE \approx 0.35$. The best result correspond to GB model with range

of Descriptors as feature space with $MAE = 0.33$, or Morgan fingerprints as feature space with $R^2 = 0.8508$.

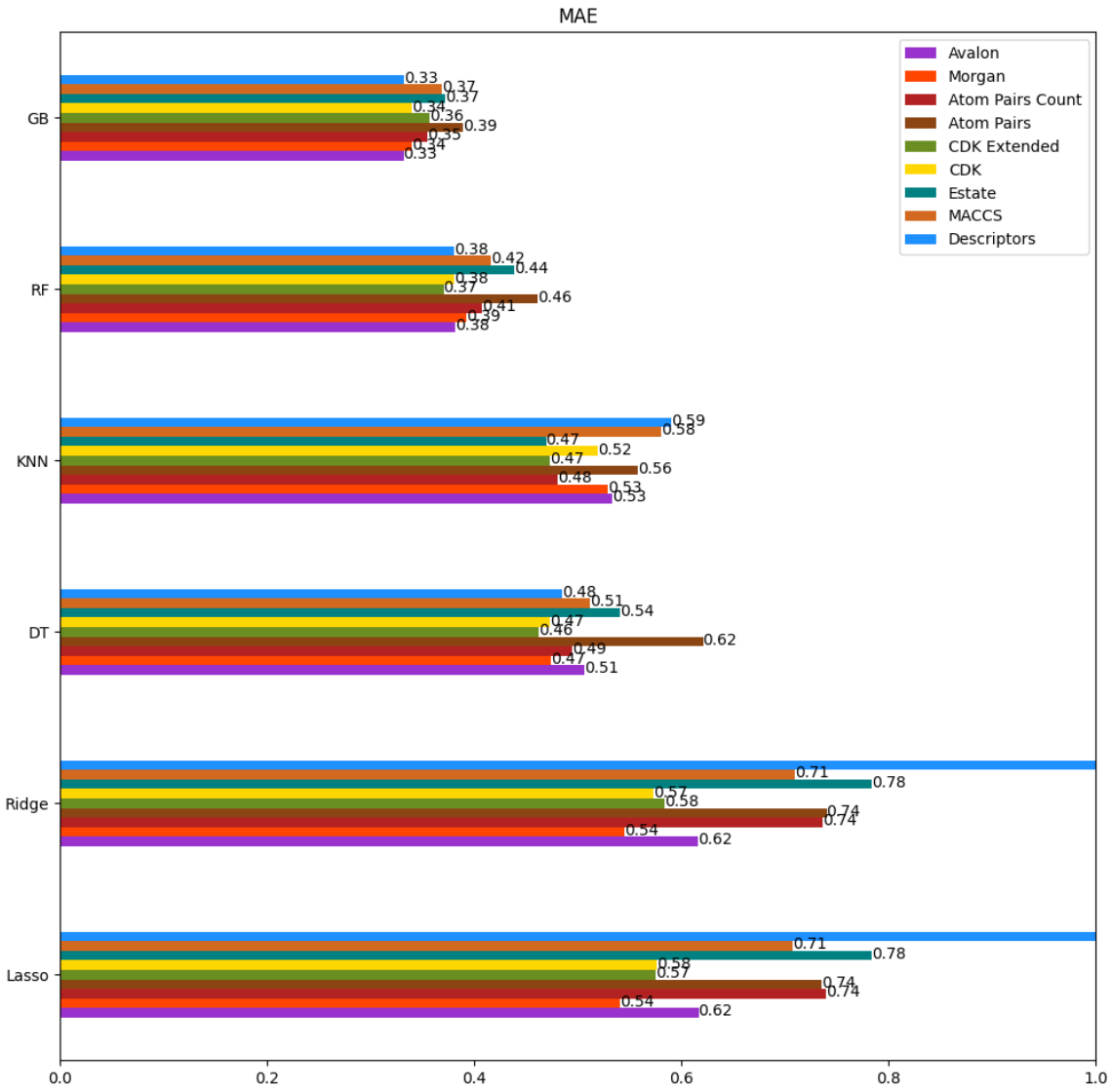

Figure 4-30: MAE scores of models, trained on imputed data (feature imputation), for *Lifetime* (log-transformed)

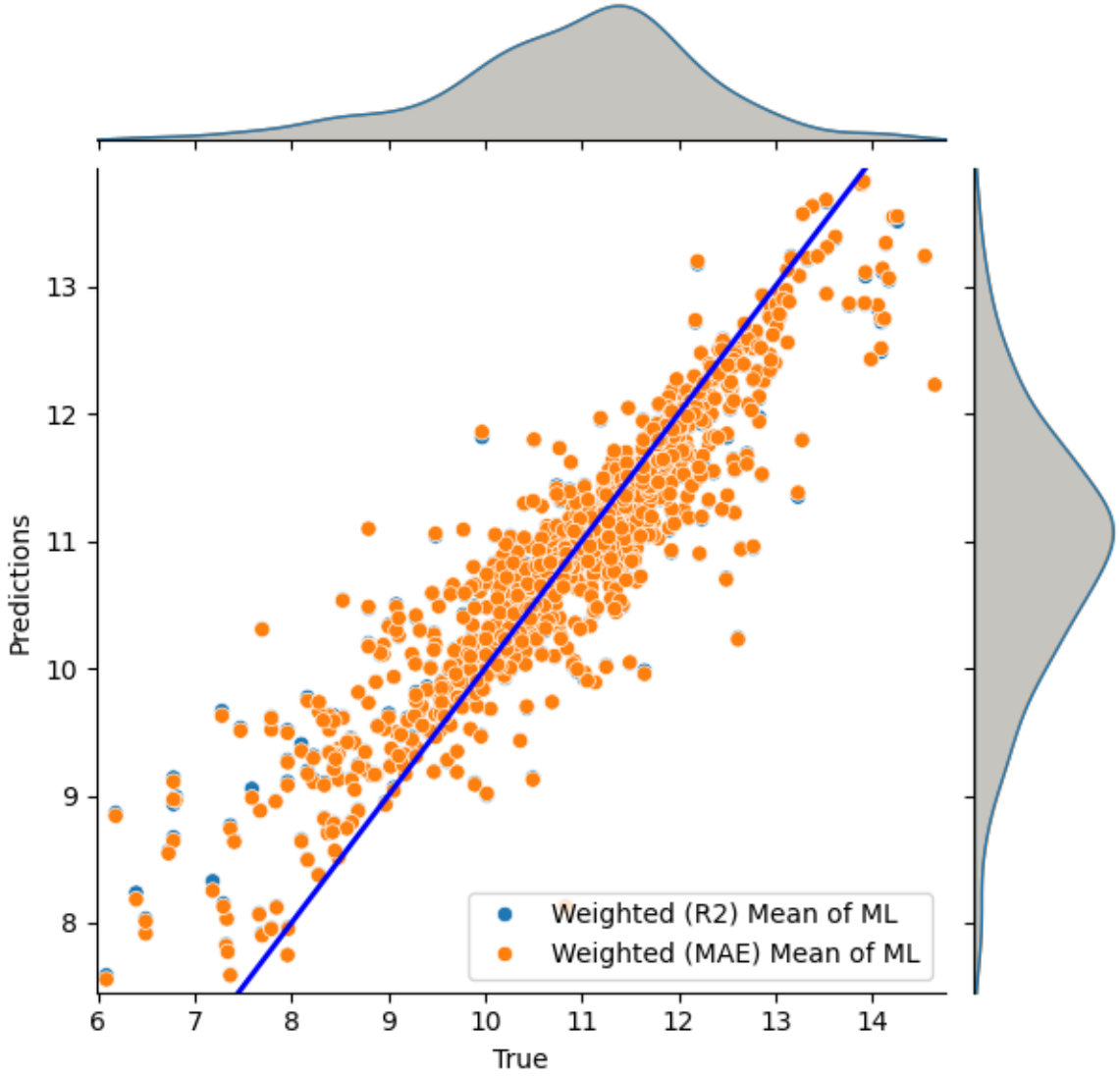

Figure 4-31: True and predicted (aggregated) with models, trained on imputed data (feature imputation), values of *Lifetime* (log-transformed)

4.4.3 Evaluating models trained on imputed data with target imputation

Another imputation option is target imputation. This means along with using other properties, one can also enlarge train data with imputed data points.

Figures 4-33 and 4-32 show R^2 and MAE scores of selected models for *Maximum absorption wavelength*.

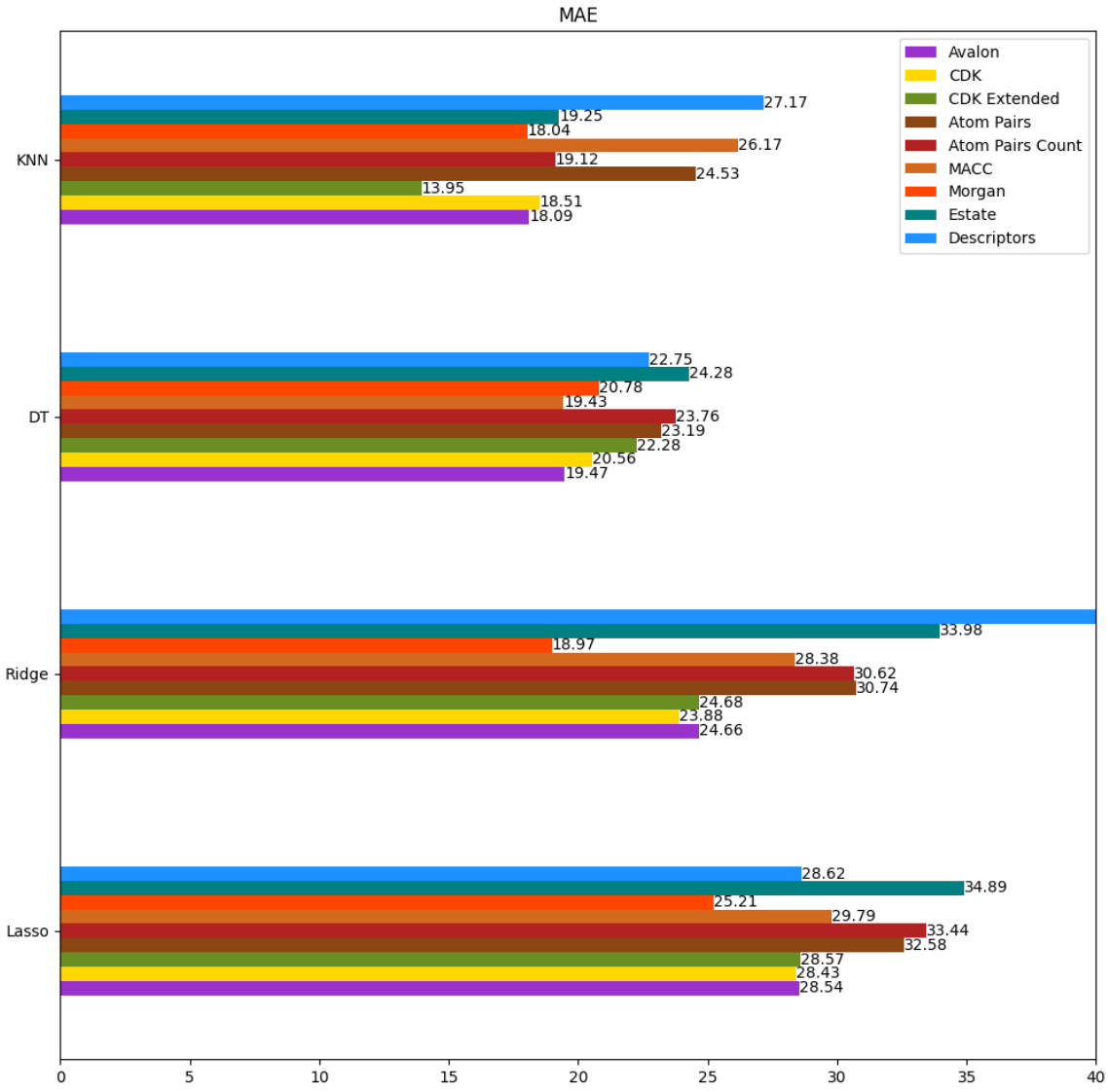

Figure 4-32: MAE scores of models, trained on imputed data (target imputation), for *Maximum absorption wavelength*

Target imputation slightly improved results, however, not as effective as feature

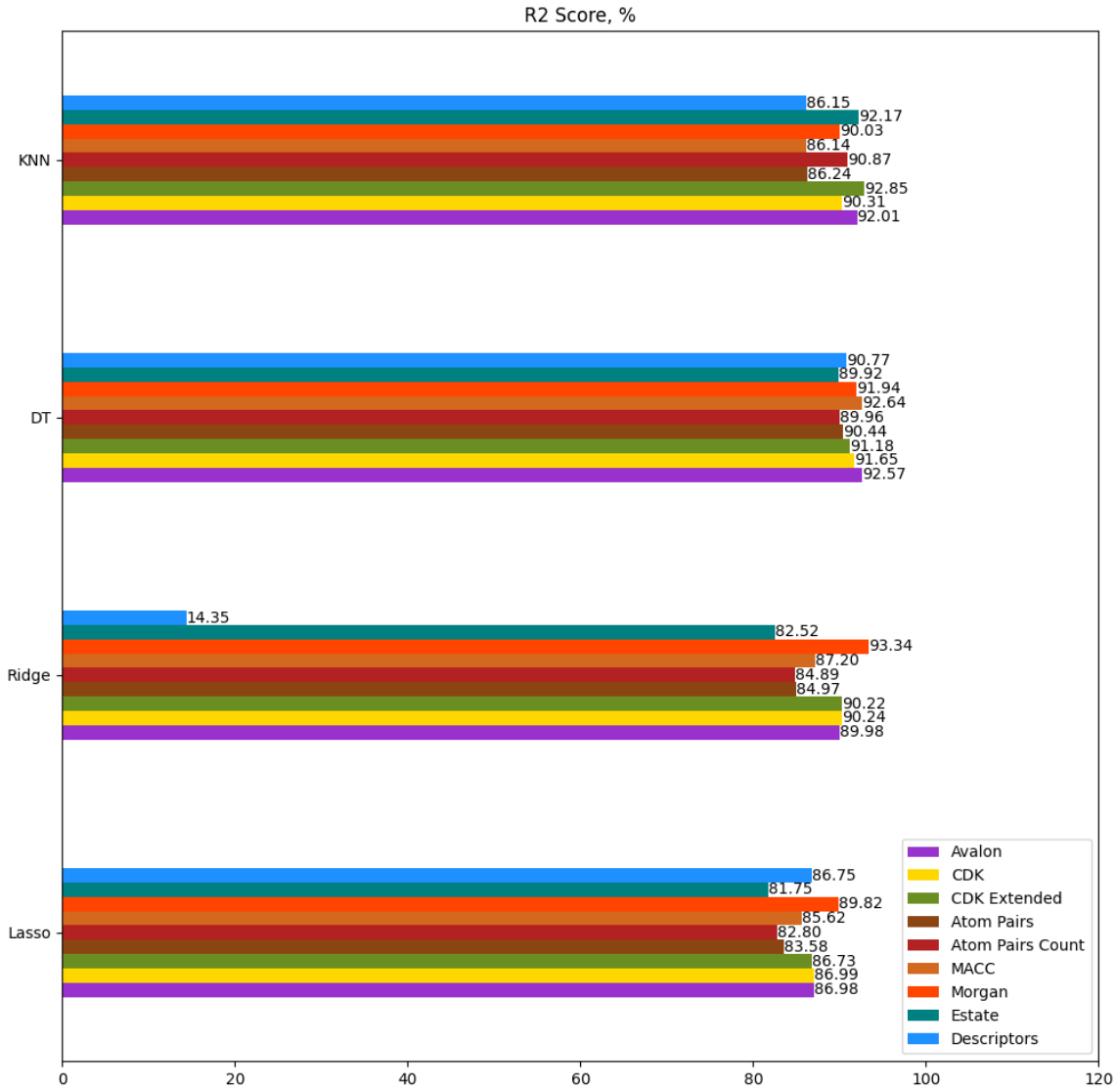

Figure 4-33: R^2 scores of models, trained on imputed data (target imputation), for *Maximum absorption wavelength*

imputation. All models, except for Ridge with range of Descriptors as feature space, achieved high R^2 and acceptable MAE scores.

All models then are aggregated with weights corresponding to the validation results, and Figure 4-34 displays plot of true against aggregated predicted values. Aggregated predictions with MAE scores from validation stage show $R^2 = 0.9352, MAE = 19.334$, on the other hand, aggregated with R^2 show $R^2 = 0.9331, MAE = 19.986$.

The best results correspond to KNN model with CDK extended fingerprints with $MAE = 13.95$, and aggregated with MAE weights of validation stage predictions of all models with $R^2 = 0.9377$.

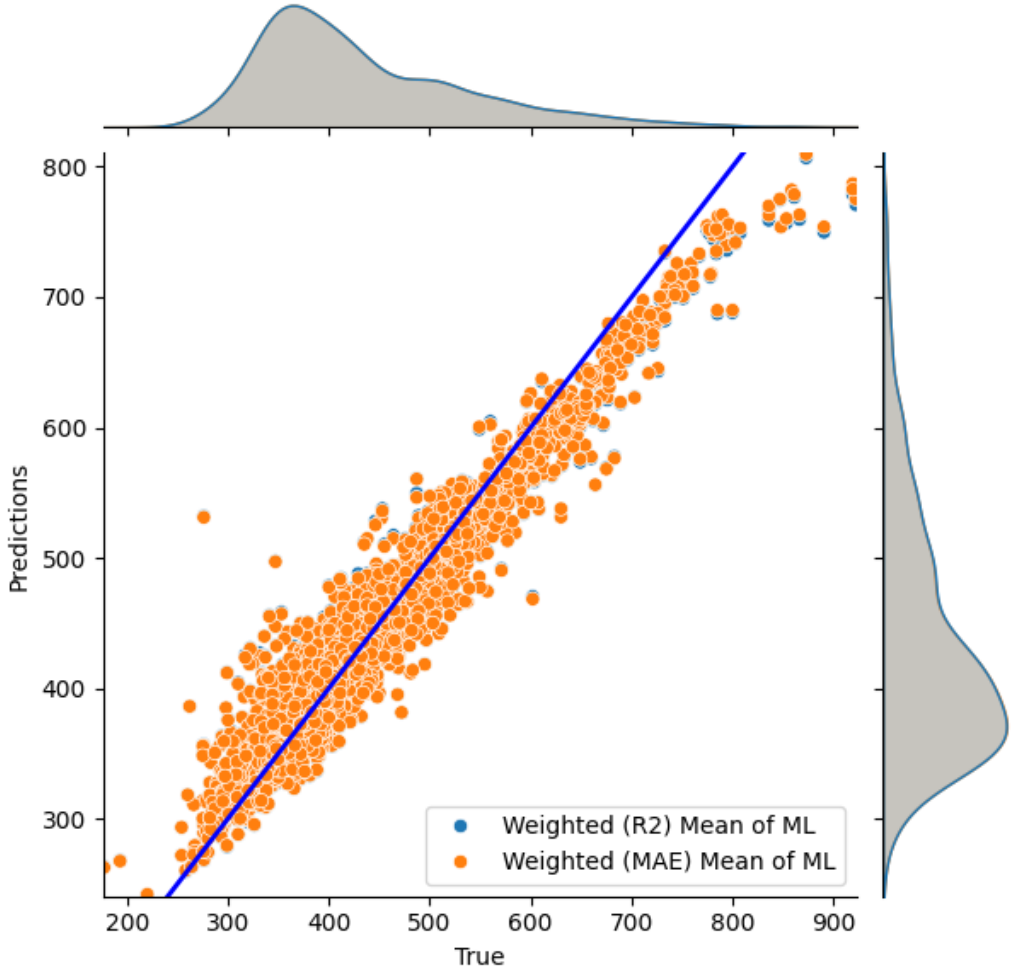

Figure 4-34: True and predicted (aggregated) with models, trained on imputed data (target imputation), values of *Maximum absorption wavelength*

Figures 4-36 and 4-35 show R^2 and MAE scores of selected models for *Maximum emission wavelength*.

Similar to *Maximum absorption wavelength*, predictions of *Maximum emission wavelength* with target imputation is somewhat improved, but not as effective as with feature imputation.

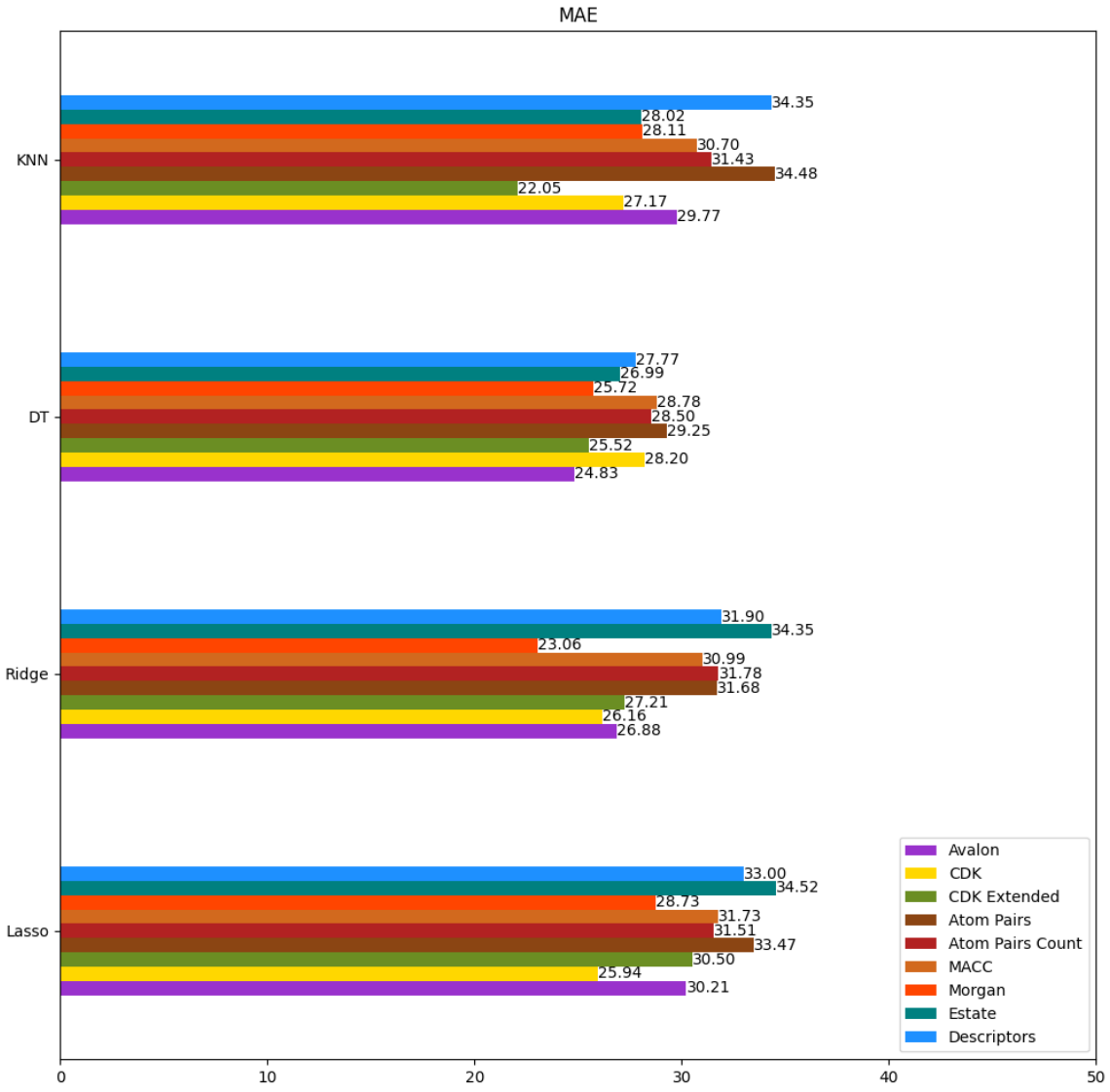

Figure 4-35: MAE scores of models, trained on imputed data (target imputation), for *Maximum emission wavelength*

All models then are aggregated with weights corresponding to the validation results, and Figure 4-37 displays plot of true against aggregated predicted values. Aggregated predictions with MAE scores from validation stage show $R^2 = 0.8786, MAE =$

23.995, on the other hand, aggregated with R^2 show $R^2 = 0.8763, MAE = 24.232$.

The best results correspond to KNN model with Morgan fingerprints with $MAE = 22.05$, and, surprisingly, Ridge with Morgan fingerprints with $R^2 = 0.8872$.

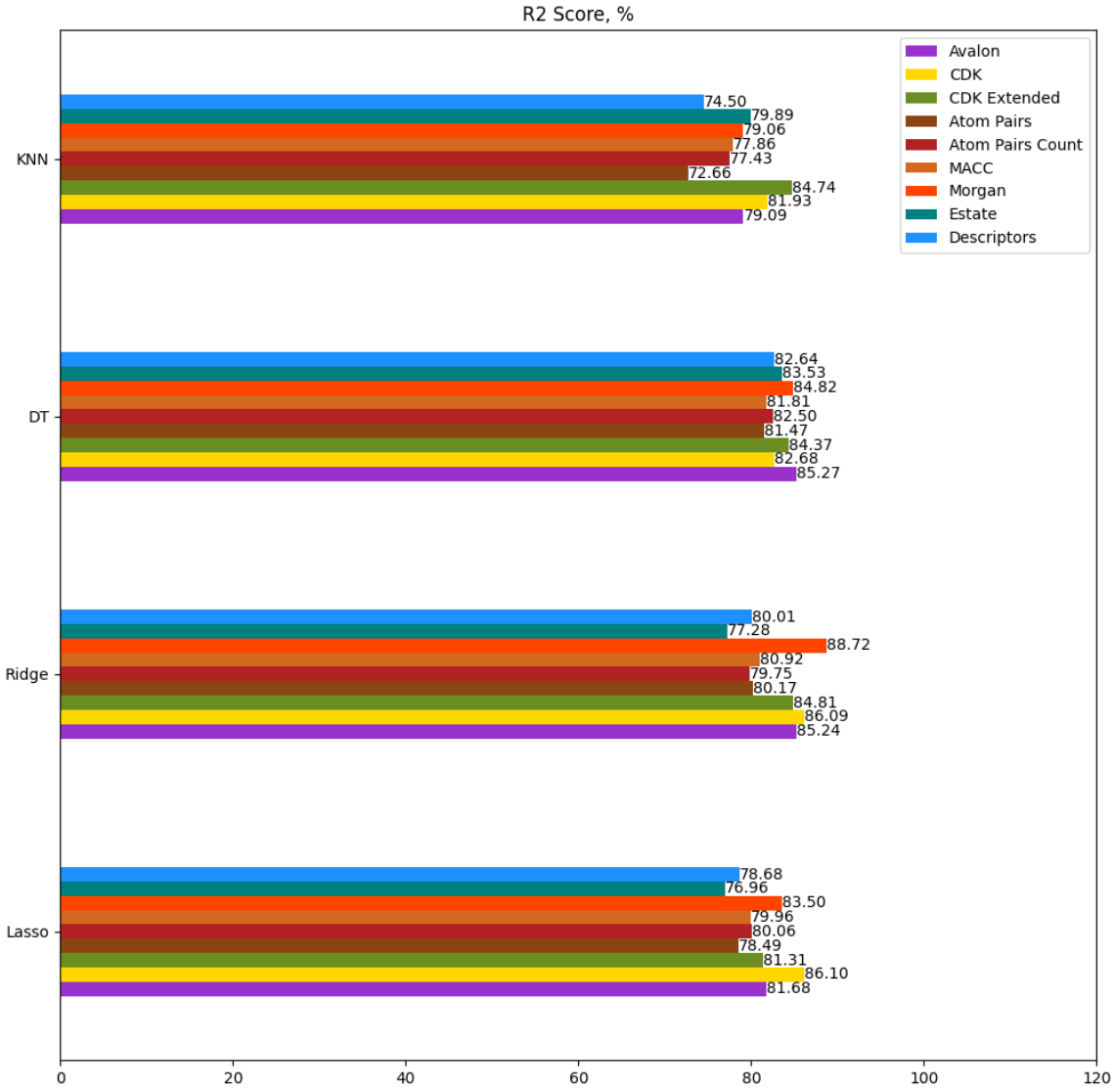

Figure 4-36: R^2 scores of models, trained on imputed data (target imputation), for *Maximum emission wavelength*

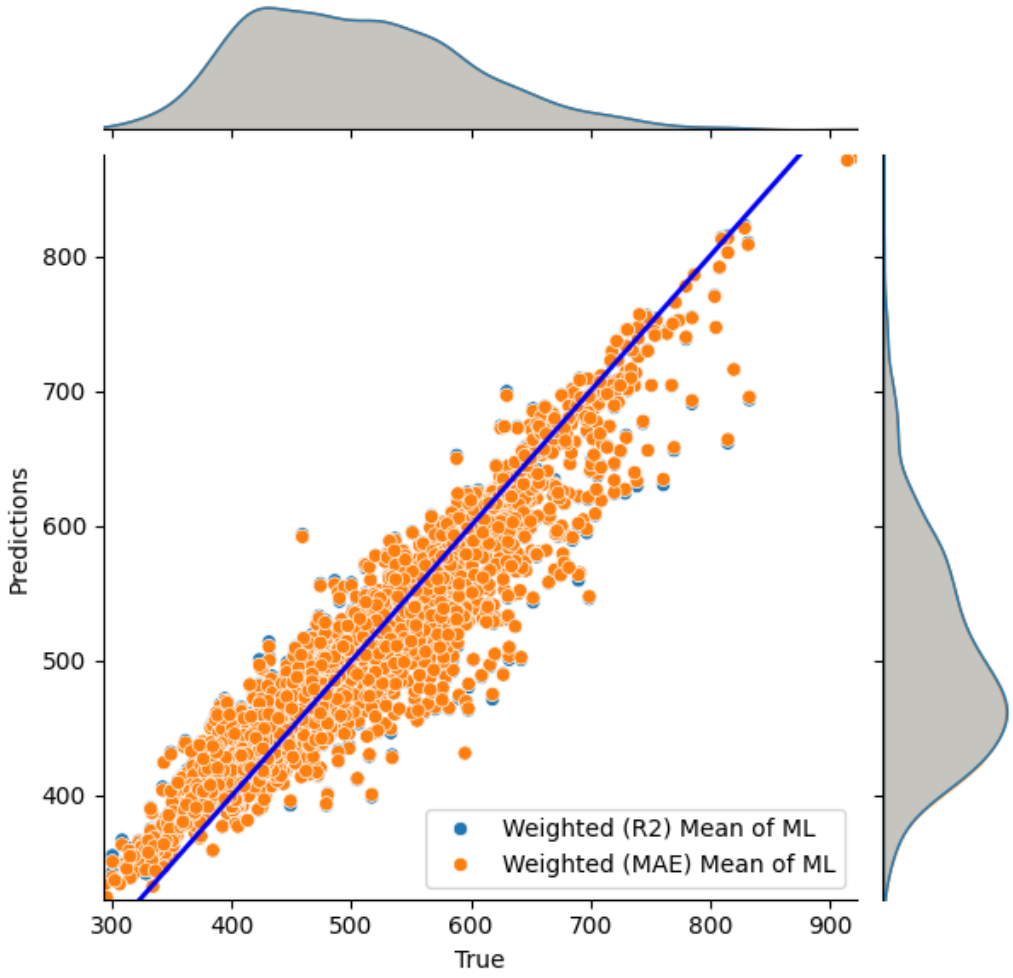

Figure 4-37: True and predicted (aggregated) with models, trained on imputed data (target imputation), values of *Maximum emission wavelength*

Figures 4-38 and 4-39 show R^2 and MAE scores of selected models for *Quantum yield*.

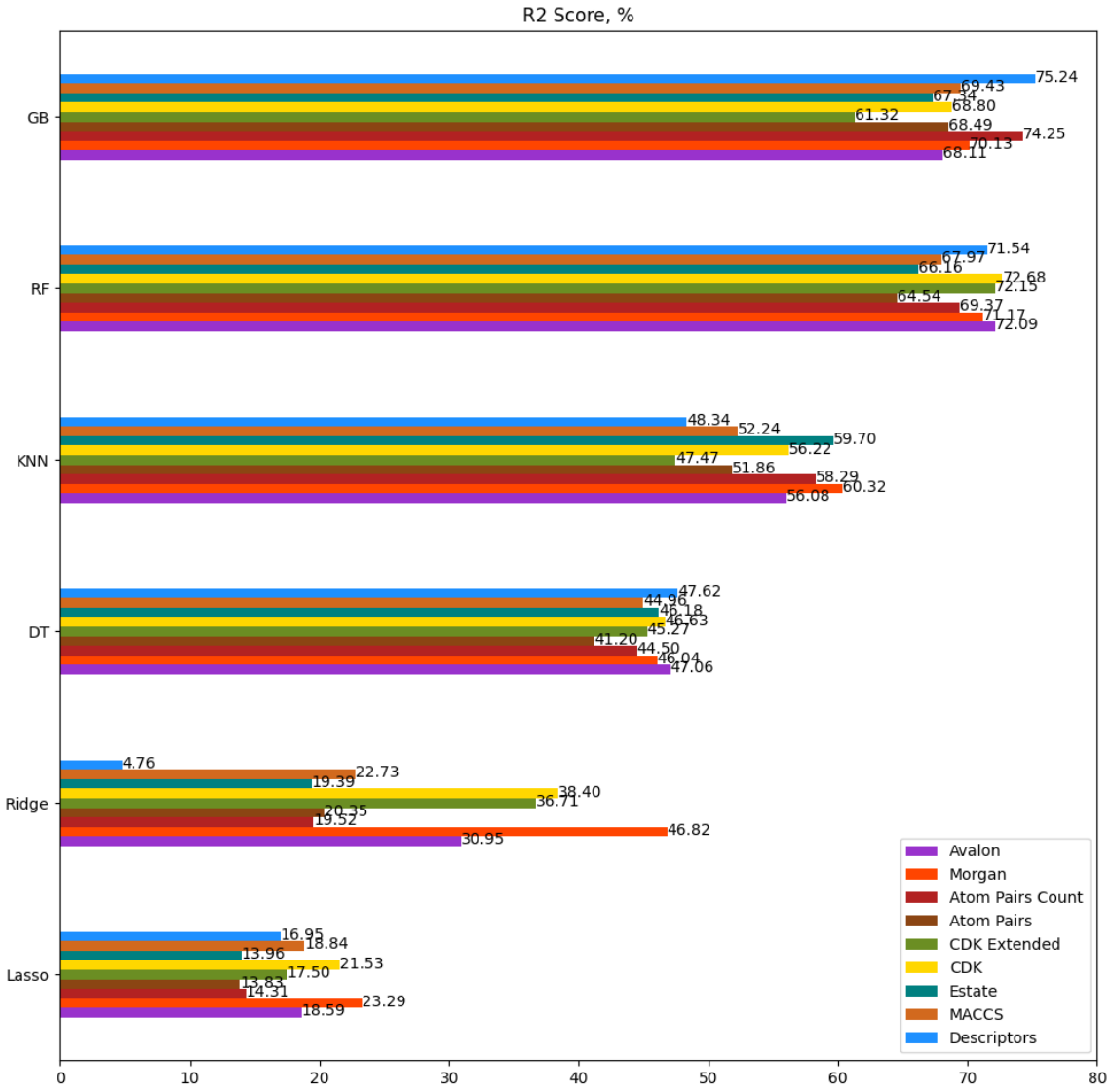

Figure 4-38: R^2 scores of models, trained on imputed data (target imputation), for *Quantum yield*

All models then are aggregated with weights corresponding to the validation results, and Figure 4-40 displays plot of true against aggregated predicted values. Aggregated predictions with MAE scores from validation stage show $R^2 = 0.6302, MAE = 0.1512$, on the other hand, aggregated with R^2 show $R^2 = 0.6723, MAE = 0.1387$.

Similar to other variants of train data, GB are best at predictions of *Quantum Yield*. The best result correspond to GB model with a range of Descriptors as feature

space with $R^2 = 0.7524, MAE = 0.11$.

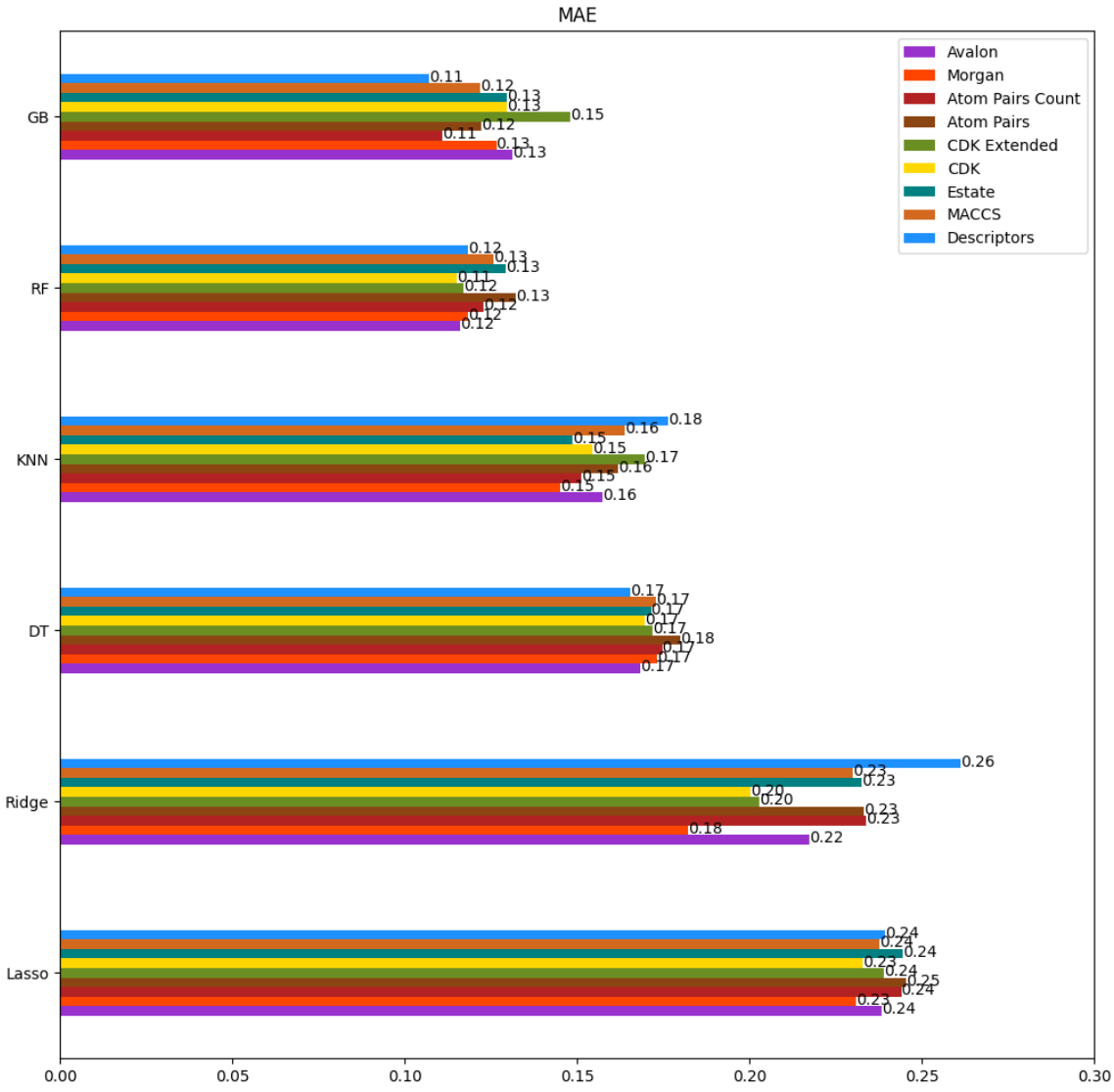

Figure 4-39: MAE scores of models, trained on imputed data (target imputation), for *Quantum yield*

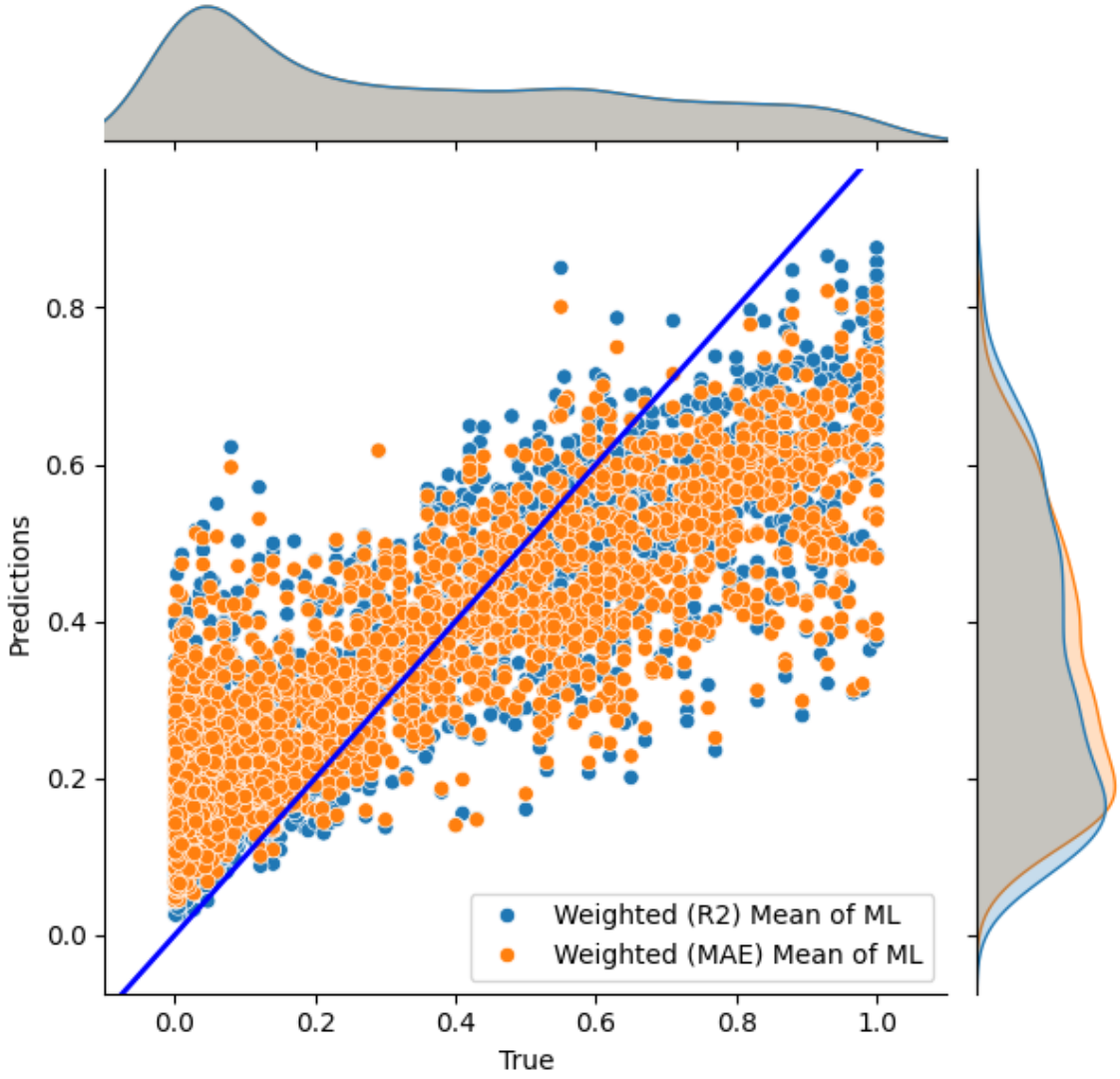

Figure 4-40: True and predicted (aggregated) with models, trained on imputed data (target imputation), values of *Quantum yield*

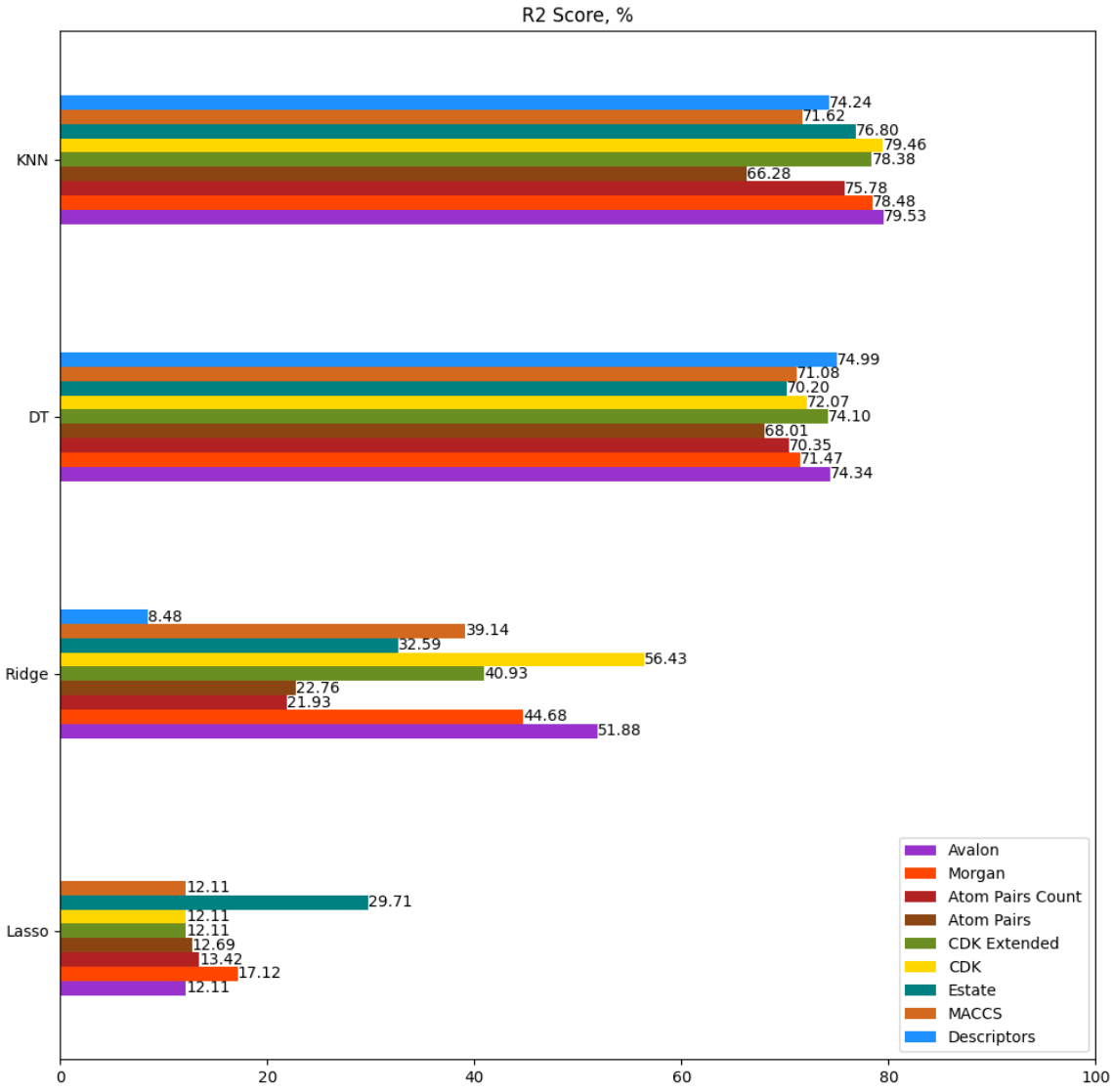

Figure 4-41: R^2 scores of models, trained on imputed data (target imputation), for *Extinction coefficient*

Figures 4-41 and 4-42 show R^2 and MAE scores of selected models for *Extinction coefficient*.

All models then are aggregated with weights corresponding to the validation results, and Figure 4-43 displays plot of true against aggregated predicted values. Aggregated predictions with MAE scores from validation stage show $R^2 = 0.6394, MAE = 0.2393$, on the other hand, aggregated with R^2 show $R^2 = 0.6668, MAE = 0.2312$.

Unlike above stated properties, target imputation worsened predicting abilities of models. This is explained with growth of artificially prepared data, The best result

correspond to KNN models with CDK extended fingerprints with $MAE = 0.15$ and Avalon fingerprints with $R^2 = 0.7953$.

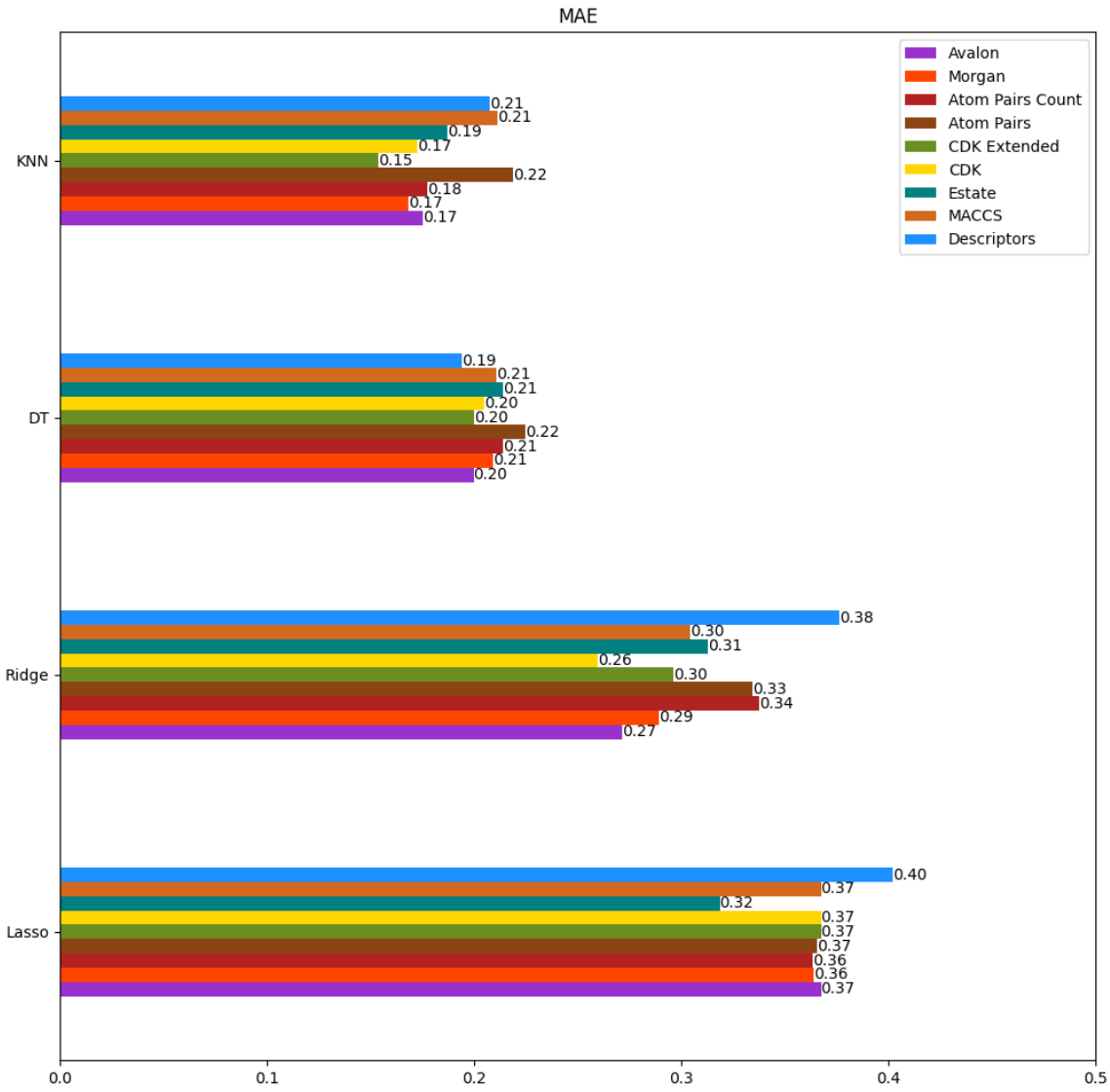

Figure 4-42: MAE scores of models, trained on imputed data (target imputation), for *Extinction coefficient*

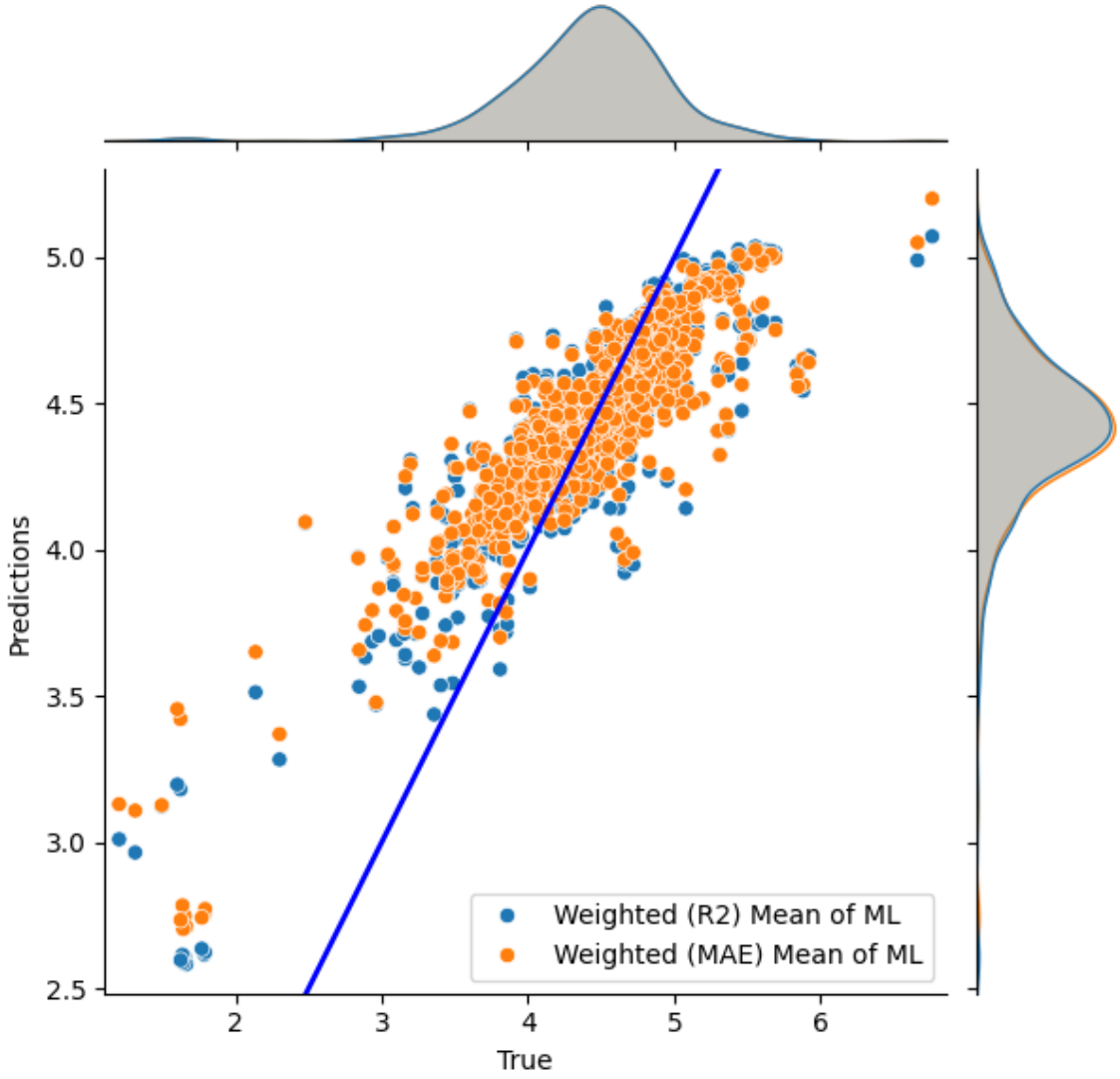

Figure 4-43: True and predicted (aggregated) with models, trained on imputed data (target imputation), values of *Extinction coefficient*

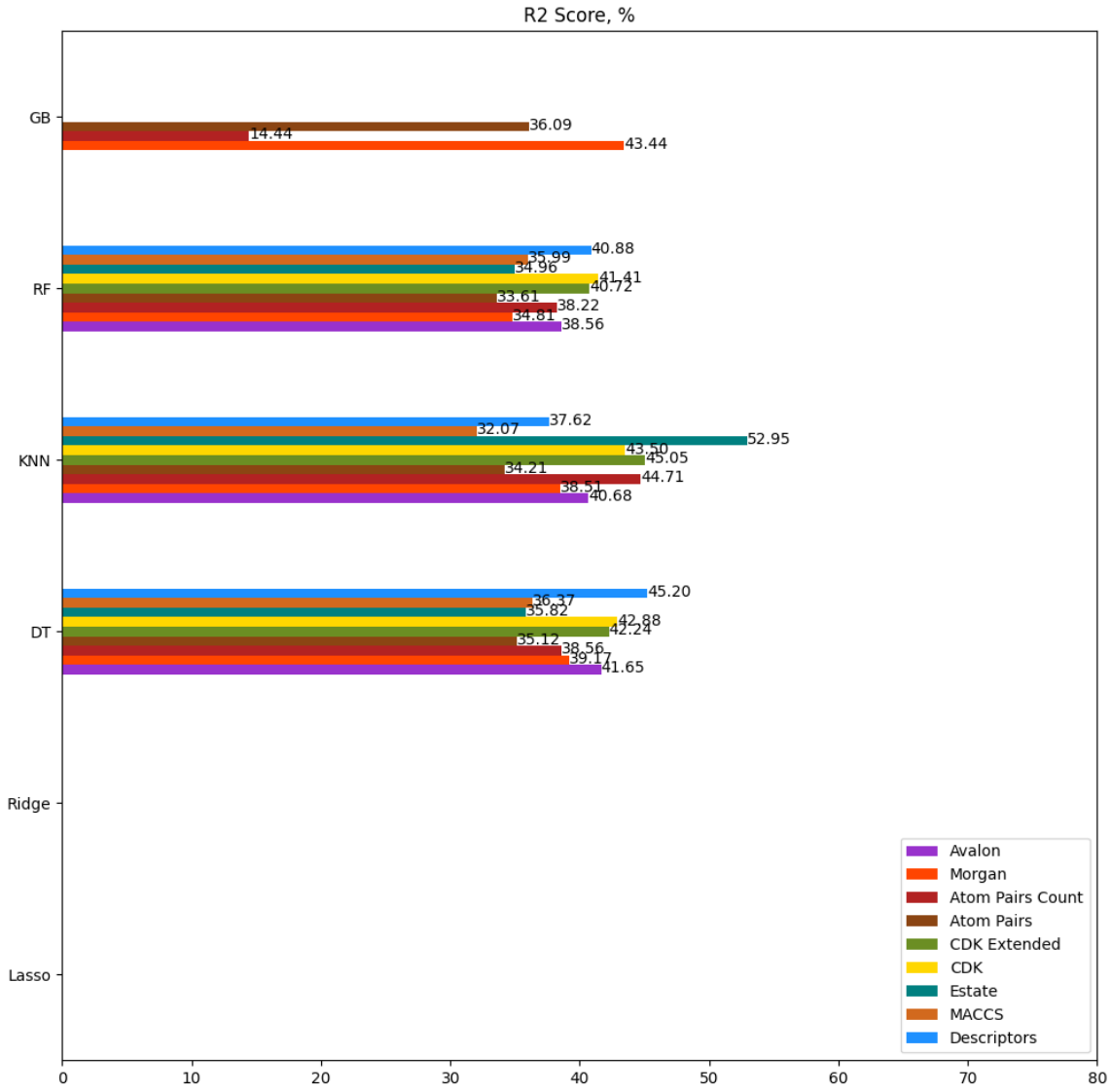

Figure 4-44: R^2 scores of models, trained on imputed data (target imputation), for *Lifetime* (log-transformed)

Figures 4-44 and 4-45 show R^2 and MAE scores of selected models for *Lifetime*.

All models then are aggregated with weights corresponding to the validation results, and Figure 4-46 displays plot of true against aggregated predicted values. Models that showed negative coefficient of determination are not considered. Aggregated predictions with MAE scores from validation stage show $R^2 = 0.4625, MAE = 0.666$, on the other hand, aggregated with R^2 show poor results.

Similar to *Extinction coefficient*, predictions of *Lifetime* worsened with target imputation due to great amount of artificial data. Now the best results correspond

to KNN models with EState fingerprints with $R^2 = 0.5295, MAE = 0.6$. EState fingerprints have the lowest size among all other molecular representations used in this work. Training scores are not provided, however there is an issue with overfitting.

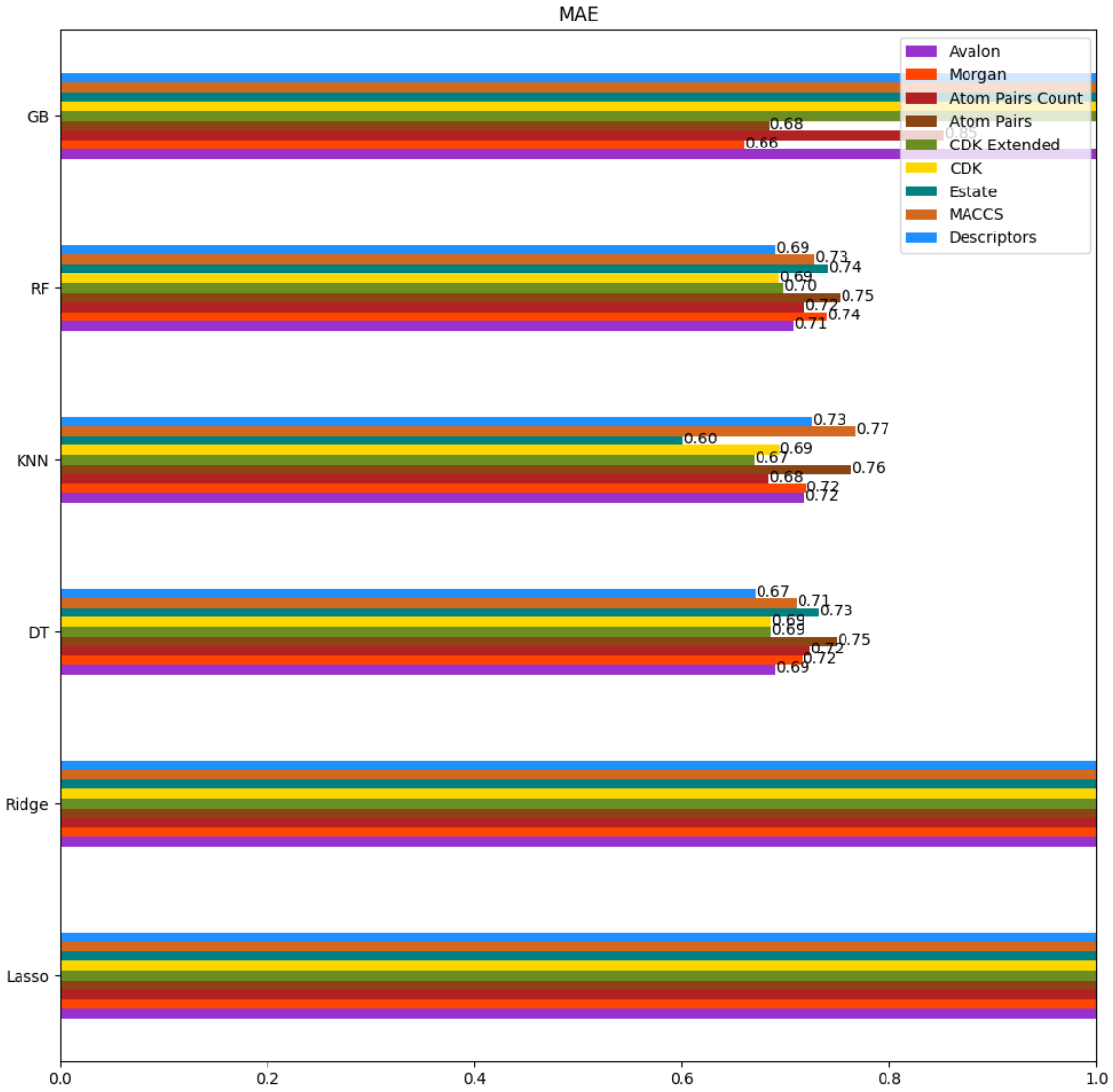

Figure 4-45: MAE scores of models, trained on imputed data (target imputation), for *Lifetime* (log-transformed)

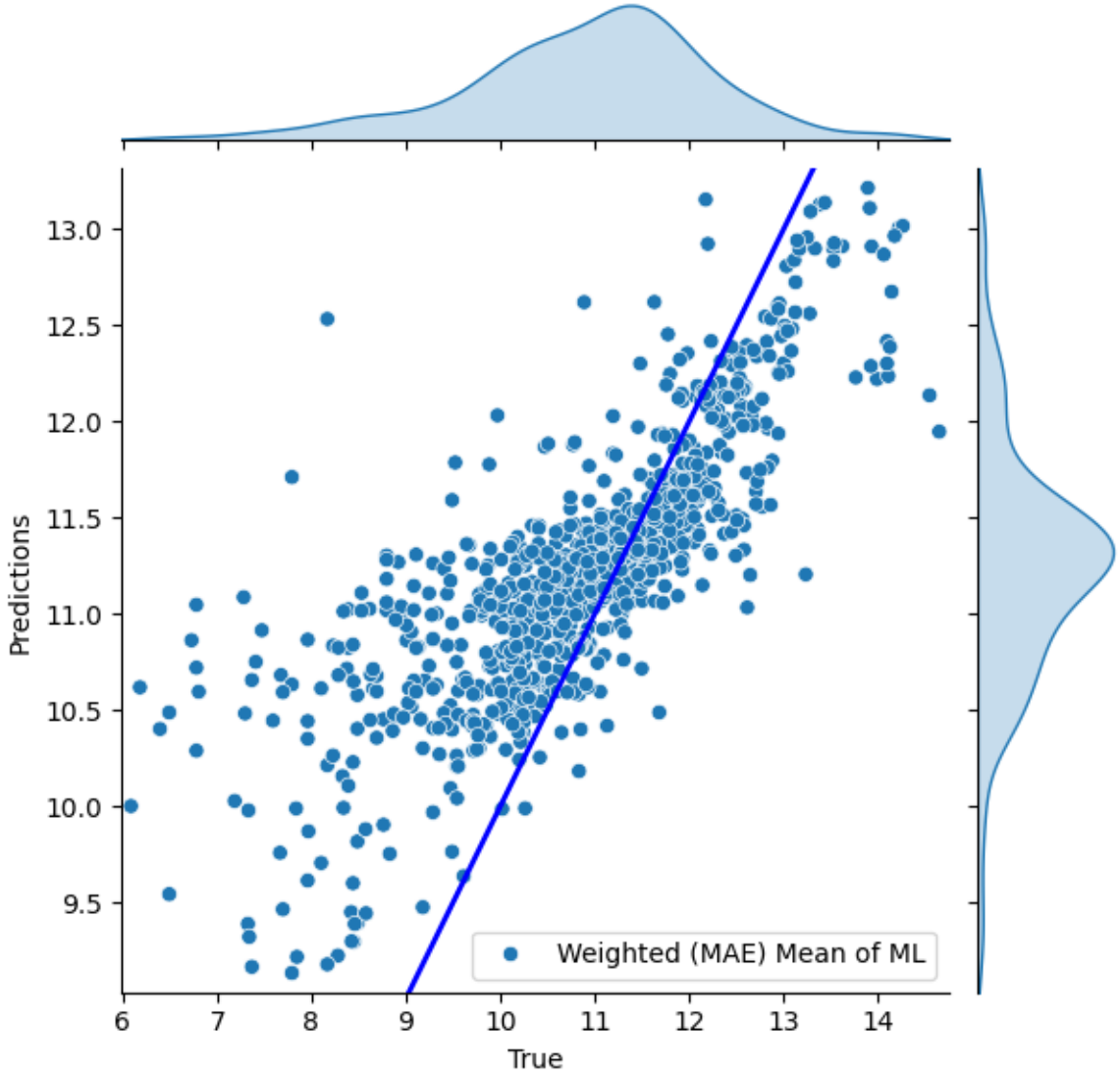

Figure 4-46: True and predicted (aggregated) with models, trained on imputed data (target imputation), values of *Lifetime* (log-transformed)

Chapter 5

Discussion

This section discusses the main points of obtained results in this work, covered in section 4, along with limitations and suggestions.

Among selected models to predict *Maximum absorption wavelength*, some showed the highest scores of R^2, MAE for each data (original and imputed). Figure 5-1 shows their results by indicating data option (ORIGINAL or any kind of imputation), selected model, and feature space. It can be observed that CDK-Extended fingerprints are best among other features to predict this particular property. Moreover, KNN exhibited the best performance. This could be explained by the fact that KNN is a non-parametric and instance-based learning algorithm that relies on similarity measures between data points. In the context of predicting fluorescent properties, the inherent similarity between fluorescent molecules or compounds likely played a significant role, or in other words, molecules with similar structures tend to have similar properties. That is why KNN might be able to identify similar instances in the dataset.

Another observation is that there is a fair difference between the results of models trained on different amounts of data. Feature imputation means that in addition to extracted features like descriptors and fingerprints, also other target properties are used, and if target properties are not provided, they are imputed. Figure 5-1 shows that in comparison with the first and the second models that are trained on original data with the KNN model with CDK-Extended and Avalon fingerprints,

the third and the fourth models trained on feature imputation data achieved better results (R^2 = 0.9019/0.933 against R^2 = 0.9499/0.9605). Figure 5-2 also supports this observation, predictions of best-selected models trained on feature imputation are closely aligned to the ideal line of equivalence. This alignment signifies a profound level of concordance between predicted outcomes and ground truth values. On the other hand, there is barely a noticeable difference between models trained on original and target imputation data. Target imputation implies that train data is enlarged with imputed data points as well as other target properties.

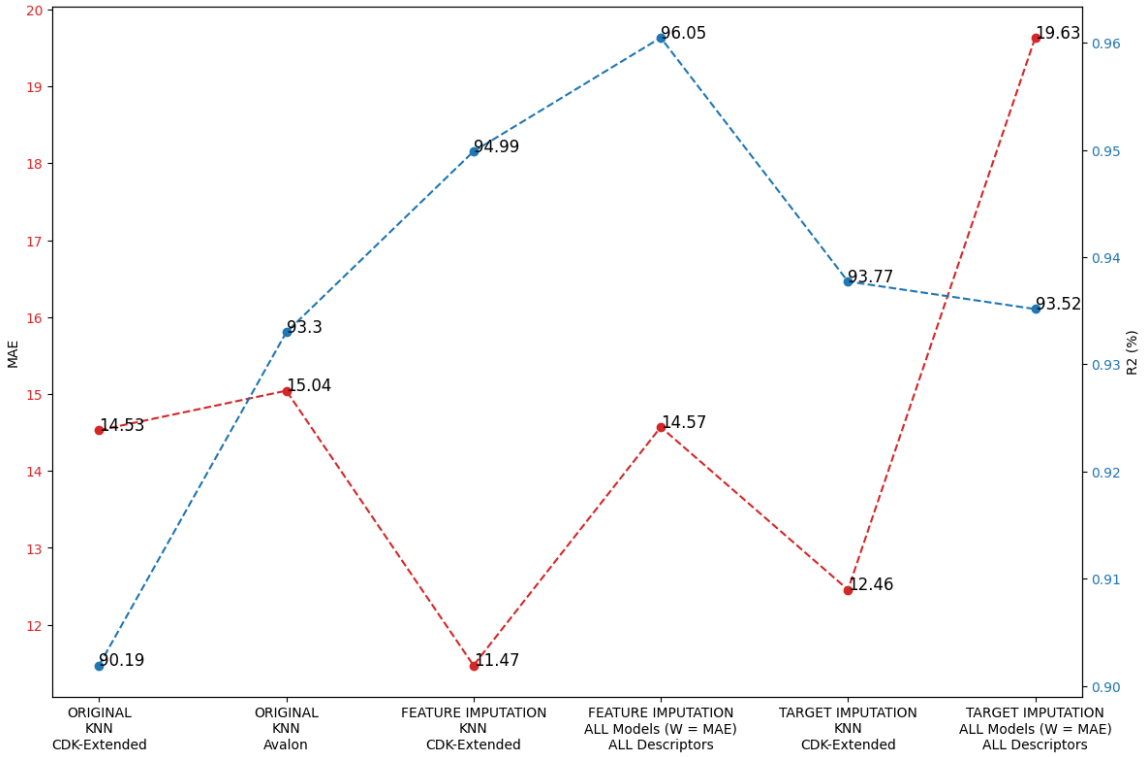

Figure 5-1: Evaluation scores of the best selected models to predict *Maximum absorption wavelength*

Similar to above-mentioned property, predictions of *Maximum emission wavelength* are improved by adding other target properties to feature space. According to Figure 5-3 with feature imputation MAE scores dropped to 19.25 and R^2 increased to 0.9173. Figure 5-4 also shows better concordance between predicted and ground truth values for the case of feature imputation. Target imputation also improved the results of models in comparison with the original data, however, it is worse than

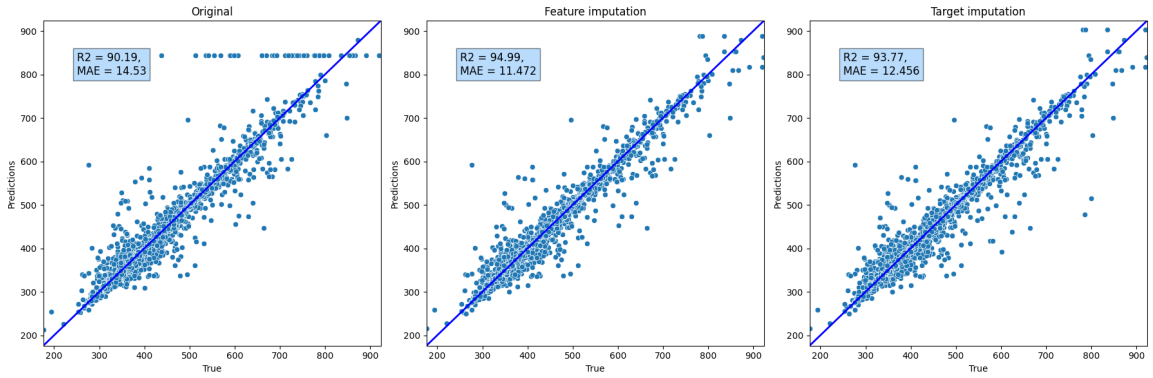

Figure 5-2: True against predicted values of *Maximum absorption wavelength* of 3 selected model trained on original and augmented data

feature imputation.

Generally speaking, improvement in predictions of *Quantum yield* is significant to the scope of this project, since unlike *Maximum absorption wavelength* or *Maximum emission wavelength*, one can detect whether a molecule is fluorescent based on *Quantum yield*. According to its definition, a *Quantum yield* of 0 means that the substance did not emit any light, thus, is not considered as fluorescent. Classification of molecules, whether they are fluorescent or not, is not one of the research objectives, however, fluorescence probe design is one of many interests in chemoinformatics.

However, results for *Quantum yield* are lower than for wavelengths. Figure 5-5 shows that the results of predictions of *Quantum yield* are mediocre, since there is no strong alignment between ground truth and predicted values. Nevertheless, there is still room for improvement. Figure 5-6 demonstrates that GB model with Avalon fingerprints and other target properties (imputed, if not provided) has achieved $R^2 \approx 0.79, MAE \approx 0.1$, which is the best obtained result in the literature.

This achievement shows that model could be built in such a way that it, firstly, predicts other target properties and those predicted values could be used to predict *Quantum yield*. This way does not contradict with ML approach because by their definition all properties of molecules from experimentation, in this case fluorescent properties, are experimental descriptors, or molecular representations. Thus, a multivariate prediction model could be suggested. However, if the model makes errors in predicting one target variable, it can propagate these errors to the predictions of

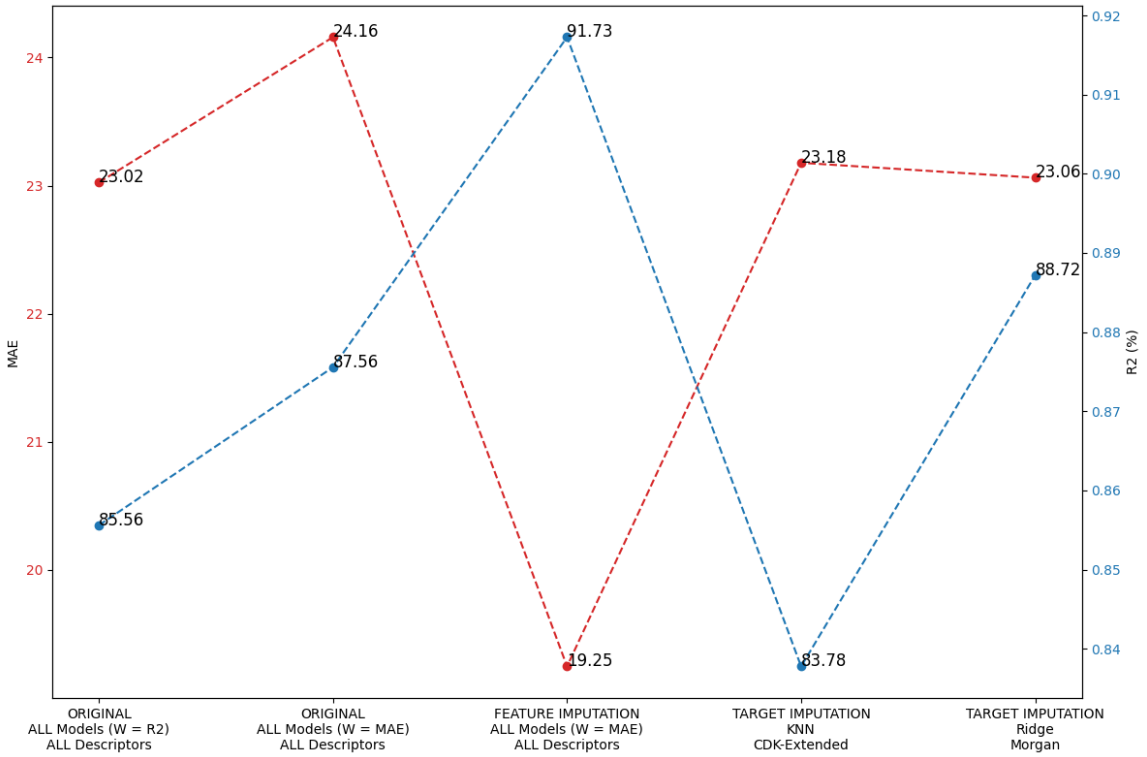

Figure 5-3: Evaluation scores of the best selected models to predict *Maximum emission wavelength*

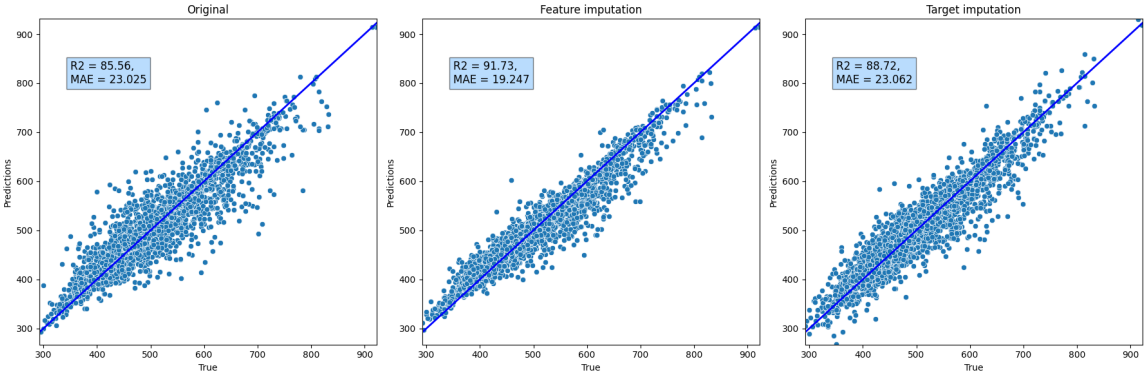

Figure 5-4: True against predicted values of *Maximum emission wavelength* by 3 selected models trained on original and augmented data

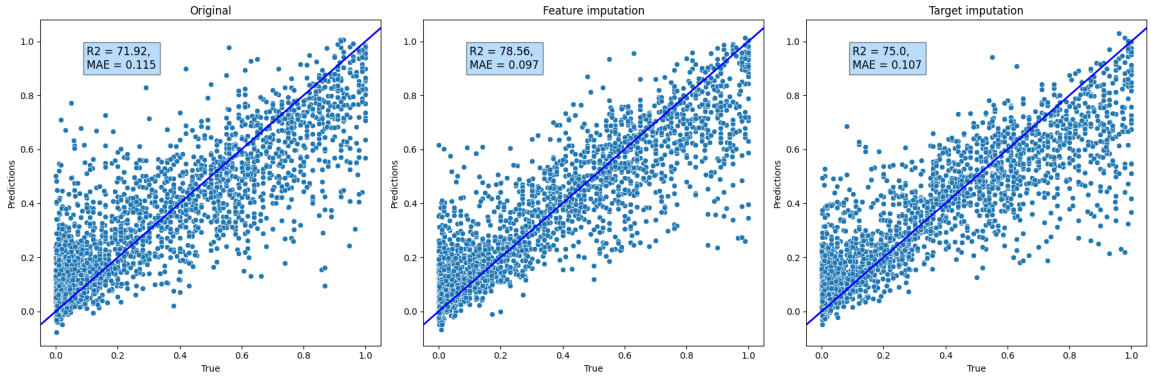

Figure 5-5: True against predicted values of *Quantum yield* by 3 selected models trained on original and augmented data

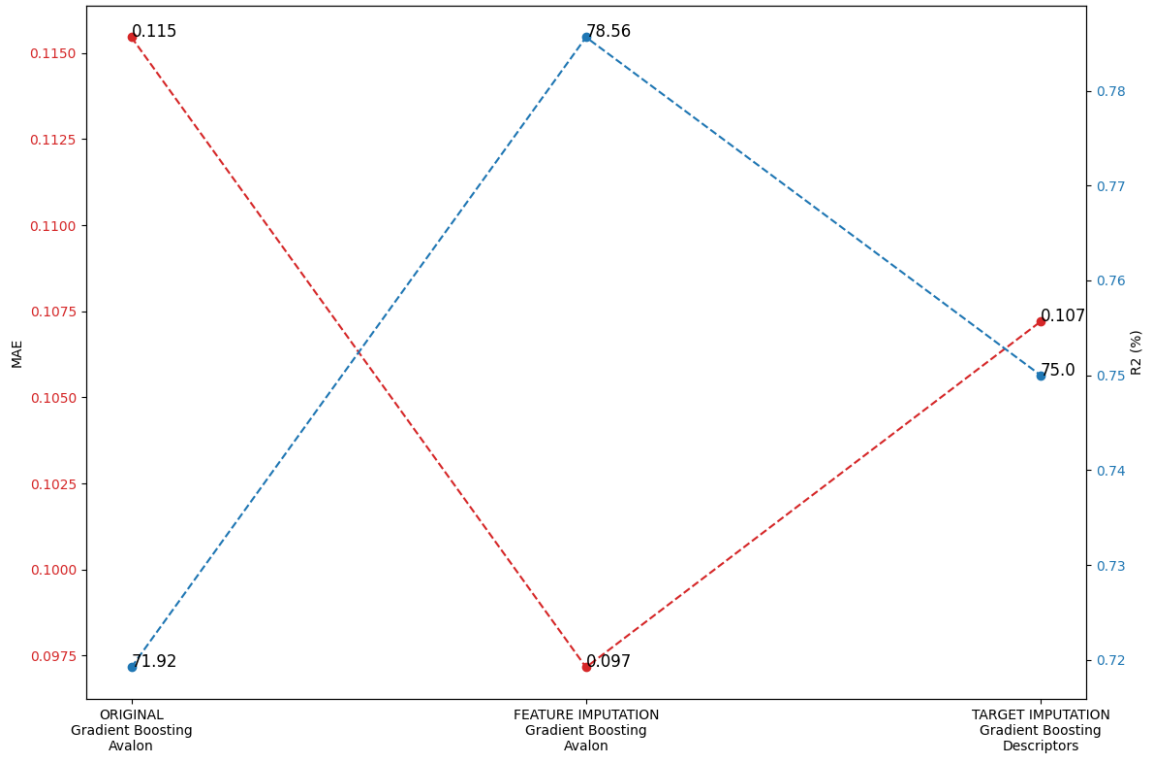

Figure 5-6: Evaluation scores of the best selected models to predict *Quantum yield*

other target variables. This phenomenon is known as error propagation. Errors in predicting one target variable can affect the model's parameter estimates, which in turn can influence the predictions of other target variables. Therefore, it's crucial to carefully consider the interdependencies between the targets when designing the model architecture.

For above mentioned 3 properties, both options of augmented data improved results of models even slightly. The number of non-missing values for *Maximum absorption wavelength*, *Maximum emission wavelength* and *Quantum yield* is 20471, 20924, and 15836, respectively. However, for the remaining properties, *Extinction coefficient* and *Lifetime*, there are only 6703, 7919 provided data points out of 22907 available combinations of chromophores and solvents. This means that the greatest amount of data used for training models with target imputation is artificial or imputed data. This circumstance has significantly worsened the results of the model to predict these particular properties.

Along with *Quantum yield*, *Lifetime* could be also an indicator of whether the substance has fluorescence or not. According to its definition, a *Lifetime* of 0 means that the substance did not spend any time in the excited state and, thus, is not considered a fluorophore.

As reported by Figure 5-7 with original models achieved at most $R^2 \approx 0.7$, while with feature imputation it is improved to $R^2 > 0.8$. In addition, Figure 5-8 displays that predicted values are better aligned with true values. However, similar to the *Extinction coefficient*, target imputation significantly dropped evaluation scores. Similarly, this is due to the fact that most of the target imputed data included data points with imputed targets.

Figure 5-9 shows that predictions of *Extinction coefficient* there is a minor difference between feature imputation and original ($R^2 = 0.8761$ against $R^2 = 0.8743$), however, target imputation results are significantly lower than others. Also, Figure 5-10 displays that there are more outliers in predictions with target imputation.

This observation means that there is an opportunity to improve imputation results. The imputation process, especially the iterative one, is computationally costly,

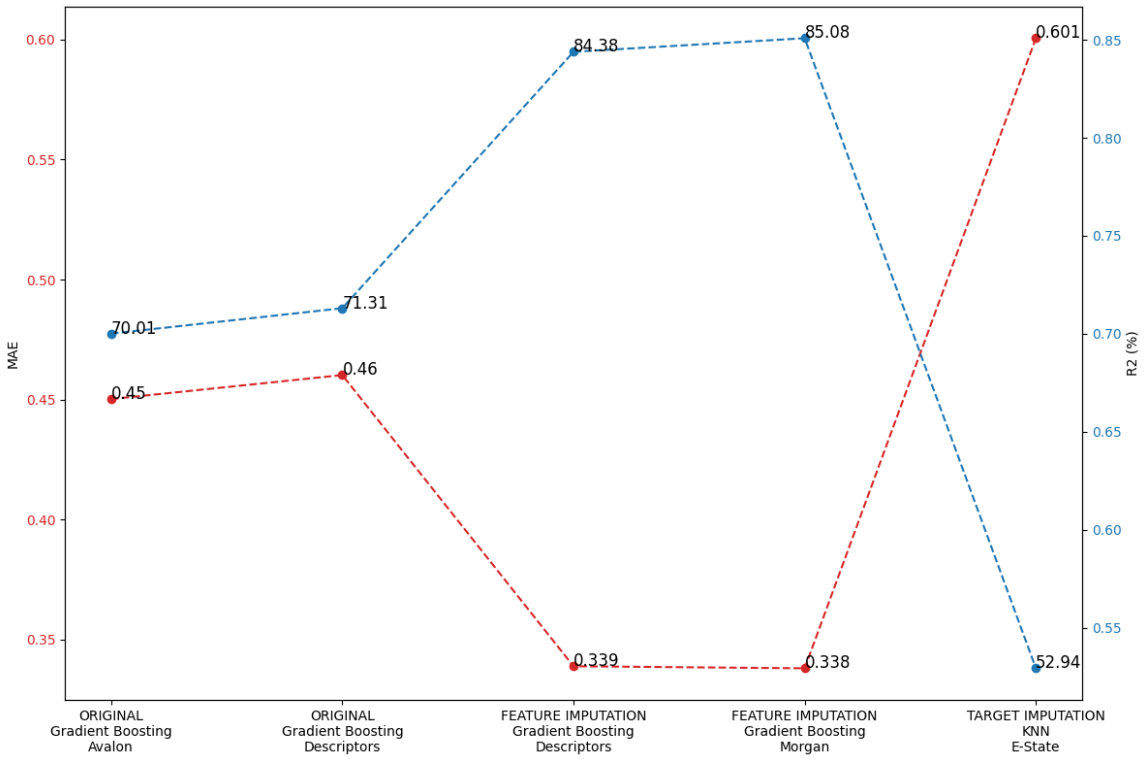

Figure 5-7: Evaluation scores of the best selected models to predict *Lifetime*

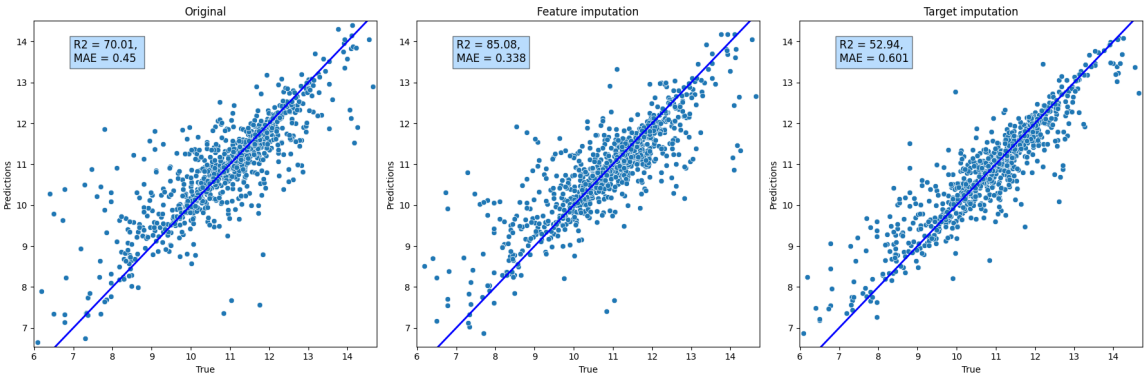

Figure 5-8: True against predicted values of *Lifetime* by 3 selected models trained on original and augmented data

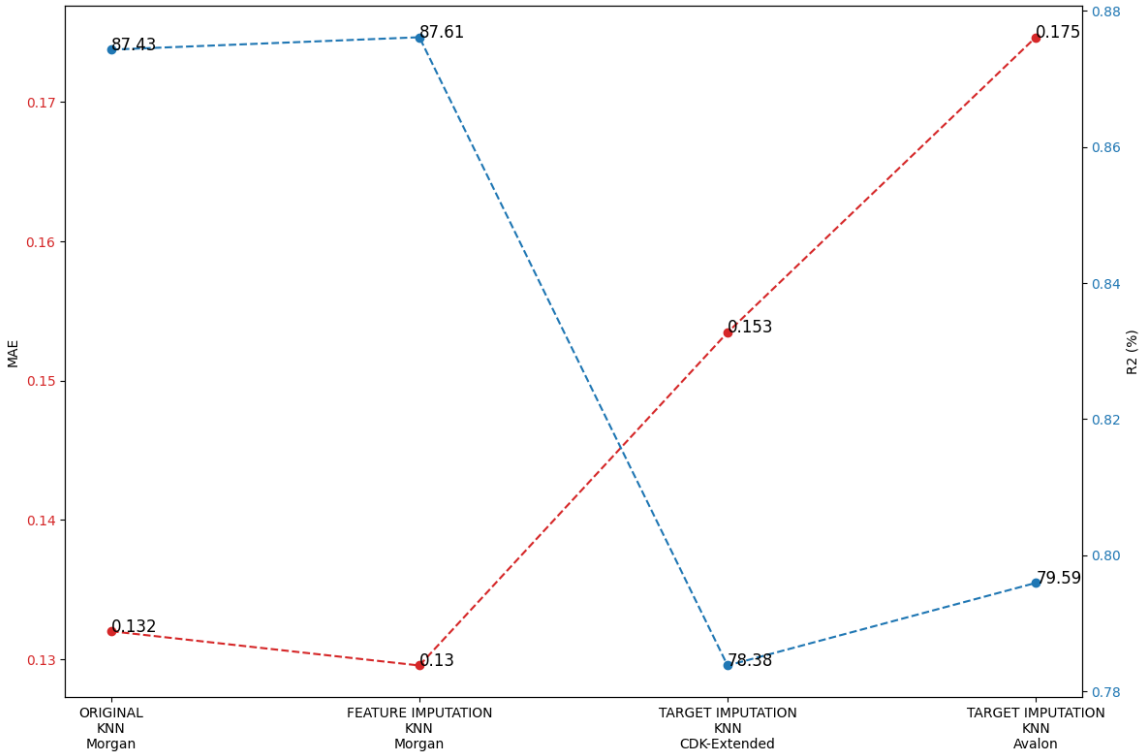

Figure 5-9: Evaluation scores of the best selected models to predict *Extinction coefficient*

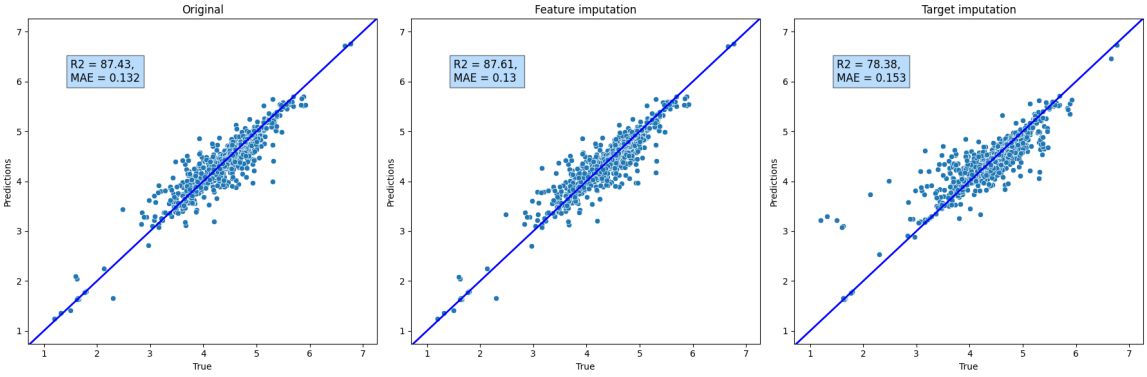

Figure 5-10: True against predicted values of *Extinction coefficient* by 3 selected models trained on original and augmented data

and only some descriptors are used along with target properties, as described before. Therefore, it is essential to properly define feature space. Along with target values, it is crucial to extract important features. This work just selected some of the descriptors, however, a more competent selection is needed. Moreover, for validation purposes multi-ranking is used. There are many approaches to accomplish rank aggregation, like Spearman's Footrule [36], and Borda Count Method [37], it is suggested to dive into this field.

Overall, this thesis has successfully addressed the challenges associated with the prediction of photophysical characteristics in organic fluorescent materials through the application of ML techniques. This work leveraged a comprehensive database and explored various imputation methods while constructing predictive models. The obtained results show the effectiveness of the proposed methodology and pave the way for further research in this area. The next step after machine-learning-assisted prediction is machine-learning enhanced design, i.e. generative models. Thus, the thesis offers a data-driven approach to address the challenges in fluorescence probe design.

Chapter 6

Conclusion

The development of fluorescent materials is crucial for various applications, such as virtual screening and imaging. However, the traditional methods of predicting fluorescent characteristics rely heavily on quantum mechanical computations and experimentations, which are often time-consuming and resource-intensive. This thesis has addressed this challenge by exploring the intersection of machine learning and fluorescence probe prediction. Through the utilization of two comprehensive databases of optical properties of organic compounds collected from various scientific papers, this research has investigated methods to overcome the limitations posed by missing data. Along with predictive modeling, imputation techniques have been explored to handle the inconsistencies in the dataset, thereby enhancing the performance of the leveraged models.

This thesis has focused on predicting five target fluorescent properties, such as Maximum absorption wavelength, Maximum emission wavelength, Quantum yield, Extinction coefficient, and Lifetime. Machine learning models included some common linear estimators, like Linear Regression, etc., and non-linear ones like, K-Nearest Neighbours, and tree-based models, along with various imputation techniques. Regression models have achieved $R^2 > 0.9$ for the first two properties, and $R^2 \approx 0.85$ for the last two properties. As for Quantum yield, the GB model with Avalon fingerprints and other target properties (imputed, if not provided, with sufficient approach) has shown $R^2 \approx 0.79, MAE \approx 0.1$, which is considered an outstanding result in the

literature. The obtained results not only underscore the potential of machine learning in fluorescence probe design but also lay a solid foundation for future advancements in the field, promising more efficient and accurate methodologies for predicting photophysical properties.

Overall, this research contributes to the advancement of the field by offering a data-driven methodology to predict the optical properties of fluorescent materials. The results obtained from the constructed predictive models are expected to demonstrate the effectiveness of machine learning techniques in predicting photophysical properties, thus paving the way for future developments in the field of fluorescence probe design.

Bibliography

[1] David E. Wolf. Chapter 4 - fundamentals of fluorescence and fluorescence microscopy. In Greenfield Sluder and David E. Wolf, editors, *Digital Microscopy*, volume 114 of *Methods in Cell Biology*, pages 69–97. Academic Press, 2013.

[2] Elizabeth A. Specht, Esther Braselmann, and Amy E. Palmer. A critical and comparative review of fluorescent tools for live-cell imaging. *Annual Review of Physiology*, 79(1):93–117, 2017. PMID: 27860833.

[3] Cheuk Hei Chan, Mingzi Sun, and Bolong Huang. Application of machine learning for advanced material prediction and design. *EcoMat*, 4(4):e12194, 2022.

[4] Yike Yang, Yumei Ji, Xu Han, Yunxin Long, Callum Stewart, Yiqiang Wen, Hok Yeung Lee, Tian Cao, Jinsong Han, Sijie Chen, and Linxian Li. Implement the materials genome initiative: Machine learning assisted fluorescent probe design for cellular substructure staining. *Advanced Materials Technologies*, 8(17):2300427, 2023.

[5] Zong-Rong Ye, I.-Shou Huang, Yu-Te Chan, Zhong-Ji Li, Chen-Cheng Liao, Hao-Rong Tsai, Meng-Chi Hsieh, Chun-Chih Chang, and Ming-Kang Tsai. Predicting the emission wavelength of organic molecules using a combinatorial qsar and machine learning approach. *RSC Advances*, 10:23834–23841, 2020.

[6] Cheng-Wei Ju, Hanzhi Bai, Bo Li, and Rizhang Liu. Machine learning enables highly accurate predictions of photophysical properties of organic fluorescent materials: Emission wavelengths and quantum yields. *Journal of Chemical Information and Modeling*, 61(3):1053–1065, 2021. PMID: 33620207.

[7] Shuai Wang, ChiYung Yam, Shuguang Chen, LiHong Hu, Liping Li, Faan-Fung Hung, Jiaqi Fan, Chi-Ming Che, and GuanHua Chen. Predictions of photophysical properties of phosphorescent platinum(ii) complexes based on ensemble machine learning approach. *Journal of Computational Chemistry*, 45(6):321–330, 2024.

[8] Jiao Chen, Mengqian Zhang, Zijun Xu, Ruoxin Ma, and Qingdong Shi. Machine-learning analysis to predict the fluorescence quantum yield of carbon quantum dots in biochar. *Science of The Total Environment*, 896:165136, 2023.

[9] Daniel S. Wigh, Jonathan M. Goodman, and Alexei A. Lapkin. A review of molecular representation in the age of machine learning. *WIREs Computational Molecular Science*, 12(5):e1603, 2022.

[10] Laurianne David, Amol Thakkar, Rocío Mercado, and Ola Engkvist. Molecular representations in ai-driven drug discovery: a review and practical guide. *Journal of Cheminformatics*, 12, 09 2020.

[11] Joonyoung Joung, Minhi Han, Minseok Jeong, and Sungnam Park. Experimental database of optical properties of organic compounds. *Scientific data*, 7:295, 09 2020.

[12] Stef van Buuren and Karin Groothuis-Oudshoorn. mice: Multivariate imputation by chained equations in r. *Journal of Statistical Software*, 45(3):1–67, 2011.

[13] Olga Troyanskaya, Michael Cantor, Gavin Sherlock, Pat Brown, Trevor Hastie, Robert Tibshirani, David Botstein, and Russ B. Altman. Missing value estimation methods for DNA microarrays . *Bioinformatics*, 17(6):520–525, 06 2001.

[14] 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, pages 785–794, New York, NY, USA, 2016. ACM.

[15] Ronald. R. Yager. On ordered weighted averaging aggregation operators in multicriteria decisionmaking. *IEEE Transactions on Systems, Man, and Cybernetics*, 18(1):183–190, 1988.

[16] RDKit. `https://www.rdkit.org/new_docs/cppapi/namespaceRDKit_1_1Descriptors.html`. [Online; accessed 11-April-2024].

[17] Lowell H. Hall and Lemont B. Kier. *The Molecular Connectivity Chi Indexes and Kappa Shape Indexes in Structure-Property Modeling*, pages 367–422. John Wiley & Sons, Ltd, 1991.

[18] Alexandru T. Balaban. Highly discriminating distance-based topological index. *Chemical Physics Letters*, 89(5):399–404, 1982.

[19] David Bonchev and Nenad Trinajstić. Information theory, distance matrix, and molecular branching. *The Journal of Chemical Physics*, 67(10):4517–4533, 11 1977.

[20] Scott A. Wildman and Gordon M. Crippen. Prediction of physicochemical parameters by atomic contributions. *Journal of Chemical Information and Computer Sciences*, 39(5):868–873, 1999.

[21] H. Lewis Morgan. The generation of a unique machine description for chemical structures-a technique developed at chemical abstracts service. *Journal of Chemical Documentation*, 5(2):107–113, 1965.

[22] Labute P. A widely applicable set of descriptors. *Journal of molecular graphics & modelling*, 18(4-5):464–477, 2000.

[23] Peter Ertl, Bernhard Rohde, and Paul Selzer. Fast calculation of molecular polar surface area as a sum of fragment-based contributions and its application to the prediction of drug transport properties. *Journal of Medicinal Chemistry*, 43(20):3714–3717, 2000. PMID: 11020286.

[24] Peter Gedeck, Bernhard Rohde, and Christian Bartels. Qsar–how good is it in practice? comparison of descriptor sets on an unbiased cross section of corporate data sets. *Journal of chemical information and modeling*, 46(5):1924–1936, 2006.

[25] Joseph L. Durant, Burton A. Leland, Douglas R. Henry, and James G. Nourse. Reoptimization of mdl keys for use in drug discovery. *Journal of Chemical Information and Computer Sciences*, 42(6):1273–1280, 2002. PMID: 12444722.

[26] Lemont B. Kier and Lowell H. Hall. An electrotopological-state index for atoms in molecules. *Pharmaceutical research*, 7(8):801–807, 1990.

[27] Raymond E. Carhart, Dennis H. Smith, and R. Venkataraghavan. Atom pairs as molecular features in structure-activity studies: definition and applications. *Journal of Chemical Information and Computer Sciences*, 25(2):64–73, 1985.

[28] David Freedman. *Statistical Models : Theory and Practice*. Cambridge University Press, August 2005.

[29] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

[30] Marvin Gruber. *Improving Efficiency by Shrinkage: The James-Stein and Ridge Regression Estimators*. Routledge, 11 2017.

[31] Lior Rokach and Oded Maimon. *Decision Trees*, pages 165–192. Springer US, Boston, MA, 2005.

[32] Tin Kam Ho. Random decision forests. In *Proceedings of 3rd International Conference on Document Analysis and Recognition*, volume 1, pages 278–282 vol.1, 1995.

[33] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *Boosting and Additive Trees*, pages 337–387. Springer New York, New York, NY, 2009.

[34] T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27, 1967.

[35] Jay L. Devore. *Probability and statistics for engineering and the sciences*. Biometrics, 1982.

[36] Kenneth J. Berry and Jr. Paul W. Mielke. Spearman's footrule as a measure of agreement. *Psychological Reports*, 80(3):839–846, 1997.

[37] Peter Emerson. *From Majority Rule to Inclusive Politics*. Springer International Publishing, 2016.