# STŘEDOŠKOLSKÁ ODBORNÁ ČINNOST

**Obor č. 18: Informatika**

# Předvídání vlastností léčiv pomocí strojového učení

UNIVERZITA KARLOVA
Farmaceutická fakulta
v Hradci Králové

**Lukáš Majer**
**Zlínský kraj**

GYMNÁZIUM ZLÍN
LESNÍ ČTVRŤ

**Zlín 2024**

# STŘEDOŠKOLSKÁ ODBORNÁ ČINNOST

**Obor č. 18: Informatika**

# Předvídání vlastností léčiv pomocí strojového učení

# Predicting the properties of drugs using machine learning

**Autoři:** Lukáš Majer
**Škola:** Gymnázium Zlín – Lesní čtvrť, Lesní čtvrť II 1364, Zlín
**Kraj:** Zlínský kraj
**Konzultant:** Eugen Hruška, Ph.D.

Zlín 2024

# Prohlášení

Prohlašuji, že jsem svou práci SOČ vypracoval/a samostatně a použil/a jsem pouze prameny a literaturu uvedené v seznamu bibliografických záznamů.

Prohlašuji, že tištěná verze a elektronická verze soutěžní práce SOČ jsou shodné.

Nemám závažný důvod proti zpřístupňování této práce v souladu se zákonem č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších předpisů.

Ve Zlíně dne 29. 2. 2024 Lukáš Majer

# Poděkování

Děkuji především panu Eugenu Hruškovi, Ph.D., který mi pomohl seznámit se s chemoinformatikou, zprostředkoval mi přístup k potřebným informacím a velmi ochotně zodpovídal mé otázky k této problematice. Vděčím mu za možnost využívat nejsilnější superpočítače v Česku, které spravuje národní superpočítačové centrum IT4Innovations. Skrze alokace OPEN-27-38 a FTA-23-21 jsem využíval jejich superpočítače Barbora a Karolina. Pan Eugen Hruška, Ph.D. mi poradil, abych se ucházel o grant Univerzity Karlovy. Grant jsem obdržel, a proto děkuji Nadačnímu fondu Univerzity Karlovy za finanční podporu mé práce. Děkuji Gymnáziu Zlín – Lesní čtvrť, kde jsem v hodinách chemie objevil svůj zájem o svět molekul, za poskytnutí studijního volna pro návštěvu odborné konference a za zapůjčení odborné literatury. Za konzultaci mé SOČ bych rád poděkoval paní učitelce Mgr. Janě Hrabíkové.

NADAČNÍ FOND
Univerzita Karlova

VŠB TECHNICKÁ UNIVERZITA OSTRAVA | IT4INNOVATIONS NÁRODNÍ SUPERPOČÍTAČOVÉ CENTRUM

# Anotace

Ve své práci se budu zabývat aplikací strojového učení pro predikci farmakokinetických parametrů metabolismu léčiv. Konkrétně se budu zabývat dvěma z nejvýznamnějších parametrů metabolismu – eliminačním poločasem a clearance. Eliminační poločas udává čas, který je třeba k tomu, aby koncentrace látky v krevní plazmě klesla na polovinu počáteční dávky. Clearance je definováno jako objem krevní plazmy, která je za jednotku času očištěna o danou látku a má jednotky objem/čas. Jinak řečeno, clearance měří, jak rychle je látka vylučována z těla. Obě hodnoty je potřeba predikovat pro rychlejší, efektivnější a levnější vývoj léčiv. Zvolil jsem programovací jazyk Python. To hlavně kvůli množství dostupných nástrojů pro strojové učení, mezi které patří také žebříčky pro strojové učení, v medicíně dostupné na Therapeutics Data Commons. Z pěti modelů, které jsem zkoumal, měl nejlepší výsledky takřka vždy RandomForestRegressor. Přesnost jednotlivých modelů si uživatel může ověřit pomocí inference, kde uživatel zvolí předtrénovaný model a zadá látku, u které bude tento model predikovat hodnotu eliminačního poločasu, případně clearance.

# Klíčová slova

Strojové učení; cytochromy P450; metabolismus léčiv; optimalizace hyperparametrů

# Annotation

In my thesis, I will discuss the application of machine learning to predict pharmacokinetic parameters of drug metabolism. Specifically, I will be looking at two of the most important parameters of drug metabolism – elimination half-life and clearance. Elimination half-life represents the time needed to reduce the concentration of a compound in blood plasma to one half of the starting dose. Clearance is defined as the volume of blood plasma which is cleared of a drug over a period of time, and it has units of volume/time. In other words, clearance measures how fast, a substance is removed from the body. The prediction of both values is crucial for faster, more efficient, and cheaper development of pharmaceuticals. I have chosen the programming language Python. Mostly because there is an abundance of tools for machine learning, including the leaderboards for machine learning in medicine available through Therapeutics Data Commons. Out of the five models I used in my work, RandomForestRegressor almost always had the best results. The user can verify the accuracy of the models with interactive inference, where they choose a pre-trained model and a compound, the half-life or clearance of which they would like the model to predict.

# Keywords

Machine learning; cytochromes P450; drug metabolism; hyperparameter optimization

# Obsah

1 Úvod ..... 7
2 Teorie ..... 8
2.1 Princip strojového učení ..... 8
2.1.1 Parametry a hyperparametry ..... 8
2.1.2 Trénování ..... 8
2.1.3 Validace (ověřování) ..... 9
2.1.4 Testování a predikce ..... 9
2.2 SMILES ..... 10
2.3 Struktura enzymů ..... 10
2.3.1 Aminokyseliny ..... 10
2.3.2 Bílkoviny ..... 11
2.3.3 Význam enzymů ..... 11
2.4 Enzymatická kinetika ..... 14
2.5 Cytochromy P450 ..... 15
2.5.1 Struktura a funkce cytochromů P450 ..... 15
2.5.2 Oxygen-rebound mechanismus ..... 16
2.5.3 Hepatocyty a mikrozomy ..... 16
2.6 Metabolismus léčiv ..... 16
2.7 Farmakokinetika a farmakodynamika ..... 17
2.7.1 Eliminační poločas a clearance ..... 17
3 Metodika ..... 18
3.1 Struktura TDC-ADME benchmarků ..... 18
3.1.1 Data splity v TDC-ADME ..... 18
3.2 Použité modely strojového učení ..... 18
3.2.1 Lineární regrese s elastic net regularizací ..... 19
3.2.2 Ridge regrese s kernel trikem ..... 20
3.2.3 Posílená regrese rozhodovacího stromu ..... 20
3.2.4 Regrese náhodného lesa ..... 23
3.2.5 Vícevrstvý perceptron ..... 24
3.3 Optuna ..... 25
3.4 Směrodatná odchylka a směrodatná odchylka chyb ..... 25

3.5 Spearmanův korelační koeficient a koeficient determinace ..... 26
3.6 Reprezentace molekul pro strojové učení ..... 26
3.6.1 Struktura molekul pomocí extended-connectivity fingerprintů ..... 26
3.6.2 Vlastnosti molekul pomocí molekulárních deskriptorů ..... 27
4 Praxe ..... 28
4.1 Příprava eliminačního poločasu a clearance látek pro strojové učení ..... 28
4.2 Příprava struktur molekul pro strojové učení ..... 28
4.2.1 Implementace ECFP ..... 28
4.2.2 Molekulární deskriptory ..... 29
4.3 Strojové učení a optimalizace hyperparametrů ..... 29
4.3.1 Strojové učení benchmarků TDC-ADME ..... 30
4.3.2 HyperparamTuner ..... 30
4.4 Interaktivní inference ..... 31
5 Výsledky ..... 31
5.1 Důležitosti hyperparametrů ..... 31
5.2 Distribuce hodnot eliminačních poločasů a clearance ..... 35
5.3 Analýza vztahu skutečných a predikovaných hodnot ..... 36
5.3.1 Poměr mezi skutečnými a predikovanými pozicemi ..... 36
5.4 Celková přesnost modelů ..... 37
5.5 Výsledky všech Optuna trials ..... 40
6 Diskuze ..... 41
7 Závěr ..... 43
8 Slovníček pojmů a seznam použité notace a zkratek ..... 43
8.1 Pojmy týkající se strojového učení ..... 43
8.2 Pojmy týkající se medicíny ..... 44
8.3 Seznam použité notace ..... 44
8.4 Seznam použitých zkratek ..... 45
9 Seznam obrázků a tabulek ..... 46
9.1 Seznam obrázků ..... 46
9.2 Seznam tabulek ..... 46
10 Reference ..... 47
11 Příloha 1: README z GitHub repozitáře ..... 51

# 1 ÚVOD

Neustále rostoucí poptávka po nových léčivech si vyžaduje vývoj nových metod pro zkoumání chování potenciálně léčivých látek. Umělá inteligence a konkrétně strojové učení se ukazuje jako jeden z nejvíce efektivních nástrojů pro tento účel. Výzkum ukazuje, že by využití umělé inteligence mohlo zrychlit a zlevnit vývoj léku. Tím by se zvýšila produkce a také dostupnost léčiv pro všechny. Proces potřebný pro to, aby se lék dostal na trh momentálně trvá 13-15 let a průměrně stojí v přepočtu zhruba 45 až 70 miliard korun. [1] Strojové učení se již ukázalo jako velice účinný přístup ve vývoji antibioticky působících látek [2] nebo třeba v určování struktury proteinů pomocí umělé inteligence AlphaFold2 [3].

AlphaFold2 je dle mého názoru momentálně největším úspěchem umělé inteligence aplikované v medicíně, a proto bych jej chtěl na úvod ve zkratce představit. AlphaFold2 vyvinula společnost DeepMind ve spolupráci s Evropským institutem pro bioinformatiku (EMBL-EBI). Tato umělá inteligence umí predikovat strukturu proteinů pouze z pořadí aminokyselin, ze kterých se protein skládá. V roce 2021 obsahovala databáze AlphaFold2 Protein Structure Database 300 tisíc predikovaných struktur proteinů. Nyní databáze zahrnuje ohromných 214 milionů předpovězených struktur. [4] Největším pokrokem oproti předešlým modelům je výrazné zvýšení přesnosti predikovaných struktur proteinů, pro které existuje mnoho sekvenčních dat aminokyselin (evoluční data pomocí Multiple Sequence Alignment nebo strukturní informace). Zlepšila se i přesnost u proteinů, pro které existuje méně dat. Přesnost těchto struktur však není srovnatelná s laboratorně popsanými proteiny. Pokud je pro protein dostatek experimentálních dat, jsou predikce dostatečně přesné a mohou se bezpečně využívat pro další experimenty.

V prvním odstavci jsem zmínil jak umělou inteligenci, tak strojové učení. Rozdíl mezi těmito dvěma pojmy je překvapivě sporný. Někteří odborníci tvrdí, že AI je podkategorie ML, někteří tvrdí, že je tomu obráceně, a ještě jiní argumentují, že jsou to naprosto nezávislé pojmy. Já osobně vnímám strojové učení jako podkategorii umělé inteligence. Nyní tyto pojmy definuji tak, jak je používám ve své práci.

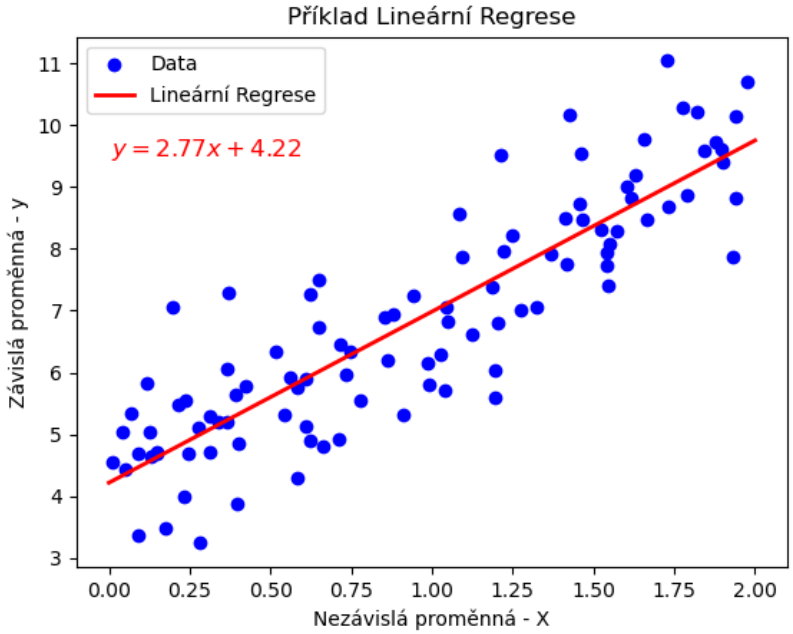
Umělá inteligence zahrnuje jakýkoliv případ, kdy se snažíme docílit toho, aby se počítač v některém ohledu choval jako člověk. Může to být například učení se z předchozích zkušeností, vlastní rozhodování nebo porozumění instrukcí člověka. Kdežto strojové učení je specifická aplikace umělé inteligence, kde se počítač „učí“ řešit konkrétní problém. V mém případě předvídání toho, jak se budou chovat léčiva v lidském těle. Učení probíhá iterativně, metodou pokus-omyl. Dalším kritickým rozdílem mezi strojovým učením a umělou inteligencí obecně je způsob rozhodování. Umělá inteligence se rozhoduje na základě předem daných pravidel (obdobně jako člověk). Strojové učení se však rozhoduje podle statistických modelů a pravděpodobností. Jinak řečeno, strojové učení slouží k rozpoznávání a využívání podobností a vzorů ve zvolených datech.

# 2 TEORIE

## 2.1 Princip strojového učení

### 2.1.1 Parametry a hyperparametry

Parametry modelu strojového učení jsou vnitřní vlastnosti modelu – nelze s nimi přímo interagovat. Před trénováním jsou jejich hodnoty náhodně určeny a během trénování se automaticky upravují, tak aby nejlépe odpovídaly vstupním datům. Příkladem parametru jsou koeficienty modelů lineární regrese (tj. způsob proložení bodů v grafu přímkou, viz obr. 1).




*Obr. 1: Ukázka, jak algoritmus lineární regrese predikuje data. Rovnice y = 2.77x + 4.22 popisuje vztah mezi x a y, který vznikl jako predikce modelu. 2.77 a 4.22 jsou konkrétní hodnoty dvou parametrů tohoto modelu.*

Hyperparametry se určují před samotným trénováním a ovlivňují jeho průběh. Zlepšováním jejich kvality pro daná data jsme schopni vytvářet mnohem přesnější modely pro predikování dat. Jednou z možností, jak zvolit vhodné hodnoty pro hyperparametry, je zasvěcený odhad, ten však mnohdy nestačí, a je proto třeba tento proces automatizovat. Princip, kterým jsou hyperparametry upravovány, je popsán dále v metodice a v praxi. [5]

### 2.1.2 Trénování

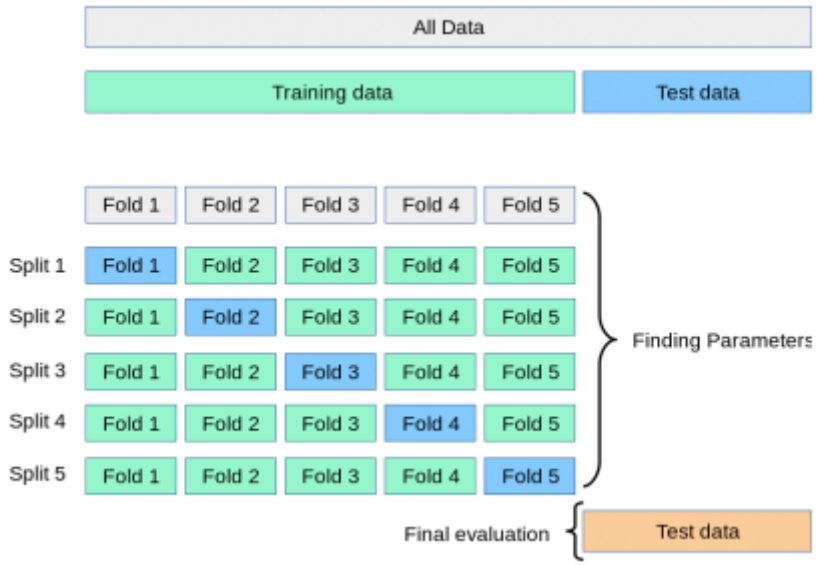
V této fázi modelu poskytneme trénovací data, která obsahují páry vstupů a cílových hodnot (tj. známé odpovědi). Cílem je, aby se model naučil vzory a pravidla, která spojují vstupy s odpovídajícími výstupy. Model se snaží minimalizovat rozdíl mezi jeho předpověďmi a skutečnými cílovými hodnotami pomocí určité optimalizační metody (konkrétním optimalizačním metodám se budu věnovat v praktické části textu).

V závislosti na zvoleném algoritmu se parametry modelu upravují tak, aby byl schopný co nejlépe aproximovat (odhadovat) závislost mezi vstupy a výstupy. Tuto závislost definuje takzvaná *loss function*, česky účelová funkce. Účelová funkce má několik vstupů, které se buď určí před strojovým učením, nebo během něj. Před učením uživatel určí data X a y, na kterých chce model trénovat. Dále také určí hyperparametry, což jsou konstanty, které mění průběh učení modelu. Jako poslední má model skryté parametry w, které se iterativně upravují na základě nějakého algoritmu gradientního sestupu[1].

### 2.1.3 Validace (ověřování)

Po dokončení trénování je důležité ověřit, jak dobře se model umí generalizovat na nová, neznámá data. V tomto kroku používáme k-fold křížovou validaci. Typicky se data rozdělí na několik (*k*) podmnožin a model je opakovaně trénován a testován na různých kombinacích těchto podmnožin. [6] Na obrázku 2 jsou trénovací data znázorněna zeleně, testovací data uvnitř jednotlivých podmnožin modře a testovací data pro konečné vyhodnocení oranžově.

Křížová validace pomáhá získat lepší odhad výkonnosti modelu na nových, neviděných datech a zabraňuje problémům s přetrénováním, kdy se model naučí „pamatovat si“ trénovací data, ale nedokáže se dobře generalizovat na nová data. Křížová validace tedy pomáhá zjistit, zda predikce modelu se zvolenými hyperparametry se budou dobře generalizovat na nová, neznámá data.




*Obr. 2: Grafické znázornění 5-fold křížové validace. [6]*

### 2.1.4 Testování a predikce

Poté, co je model trénován i validován, používáme ho k predikci hodnot pro nová data, která nebyla součástí trénovacích dat. To se nazývá testování. Testovací data jsou

[1] Gradientní sestup je iterativní algoritmus pro nalezení lokálního minima funkce.

klíčová pro posouzení skutečného výkonu modelu na neznámých datech. Pro zkoumání přesnosti predikcí modelu jsem zvolil mimo jiné Spearmanův korelační koeficient, který je využíván v žebříčcích Therapeutics Data Commons [7]. Význam této veličiny je blíže popsán v metodice. Pokud jsme pro predikce modelu spočítali dobré hodnoty metrik, tak to znamená, že model dobře vysvětluje vztah mezi zkoumanými hodnotami, a je tedy možné jej využít v praxi. Dá se tedy očekávat, že model s dobrými hodnotami metrik bude schopný přesně předpovědět hodnoty i v reálných aplikacích.

## 2.2 SMILES

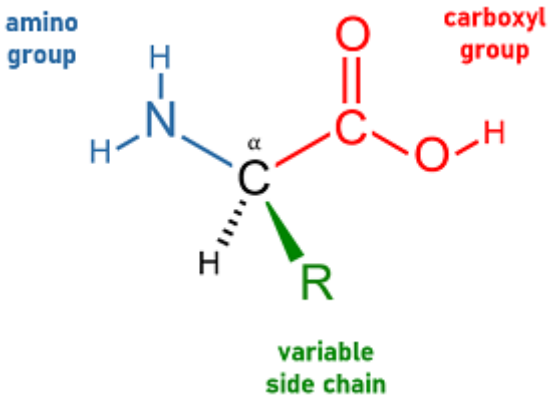
Simplified molecular-input line-entry system, česky zjednodušená molekulární specifikace pro vstupní řádky, je systém, pomocí kterého lze zapsat strukturu molekuly do řetězce. V řetězci SMILES jsou pomocí jejich chemické značky zapsány takřka všechny atomy, které molekula obsahuje. Většinou se nezapisují atomy vodíku. Většinou totiž nehrají důležitou roli a lze je dopočítat z ostatních atomů a vazeb mezi nimi. Samotné vazby jsou dány pořadím atomů v řetězci. Cykly a násobné vazby nelze vyvodit čistě z těchto informací, a proto musí být v řetězci explicitně stanoveny. To samé platí pro aromatické části molekuly a někdy i pro vodíky. 3D struktura molekul se v SMILES zapisuje dodatečnými symboly. [8]

## 2.3 Struktura enzymů

Enzymy jsou velké molekuly biologické povahy, které mají schopnost mnohonásobně urychlovat chemické reakce. Díky tomu jsou velmi významné pro medicínu i jiné obory.

### 2.3.1 Aminokyseliny

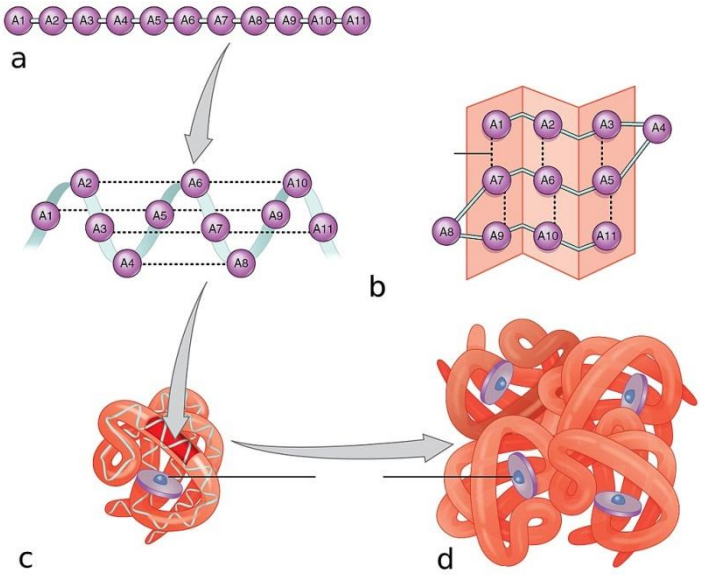
Základní stavební jednotkou proteinů, tudíž i enzymů, jsou aminokyseliny. Aminokyseliny jsou molekuly, které obsahují jak karboxylovou (-COOH), tak aminovou ($-NH_2$) skupinu, navázané na centrální atom uhlíku (tzv. alfa uhlík). Dále bývá na tento uhlík navázaný atom vodíku a další skupina, obecně značená R, kterou se aminokyseliny vzájemně liší. [9]




*Obr. 3: Obecná struktura aminokyseliny. [10]*

### 2.3.2 Bílkoviny

Spojením více aminokyselin vzniknou peptidy až bílkoviny neboli proteiny. Tato reakce aminokyselin se nazývá kondenzace. Bílkoviny tvoří největší podíl složení tkání vyšších organismů, nepočítáme-li vodu. Kondenzace aminokyselin probíhá propojením karboxylové skupiny jedné aminokyseliny s aminovou skupinou druhé. Tímto se obě aminokyseliny provážou velmi pevnou peptidickou vazbou -CO-NH-.




*Obr. 4: ilustrace jednotlivých úrovní uspořádání proteinu; a) primární; b) sekundární (vlevo α-helix, vpravo β-hřeben); c) terciární; d) kvartérní. [11]*

Vodíkové můstky jsou jedním ze základních typů vazeb, které se objevují při interakcích organických nebo bio molekul. Tyto vazby vznikají přitahováním mezi vodíkem jedné molekuly a jiným atomem sousední molekuly. Prvky, které jsou schopny tvořit vazby s vodíkem, musí mít vysokou elektronegativitu. Mezi takové prvky patří například kyslík nebo dusík. Samotný vodík musí také být navázaný na jeden z těchto prvků. V této interakci funguje atom, na který je vodík navázaný, jako donor. To znamená, že poskytuje vodík do vodíkového můstku. Molekula s elektronegativním atomem funguje jako akceptor, přijímající vodíkový atom. Díky tomuto sdílenému elektronu vzniká mezi donorem a akceptorem chemická vazba.

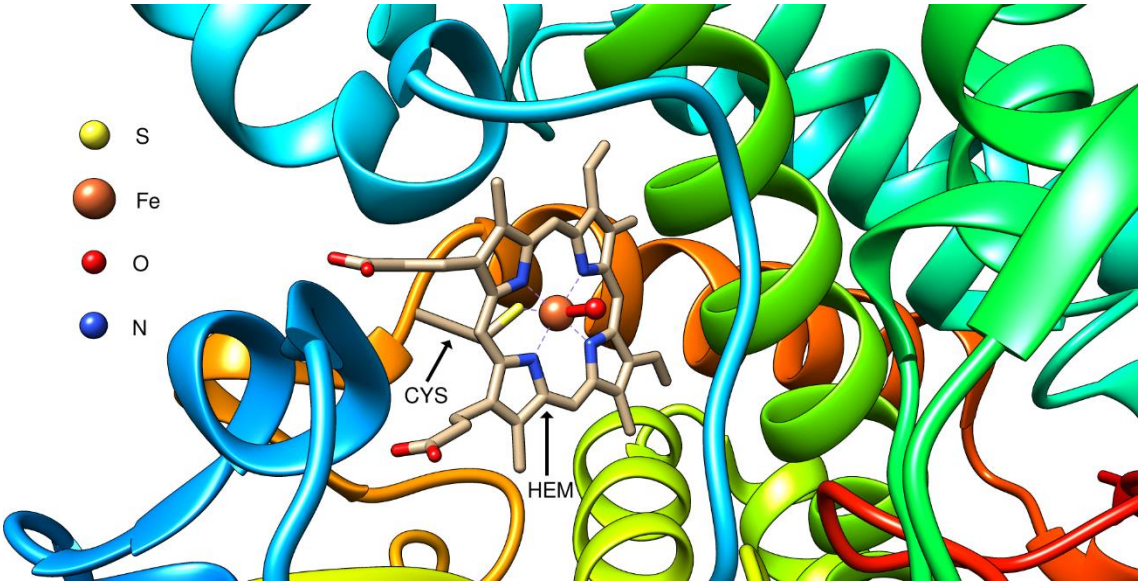
Struktura bílkovin je velmi komplikovaná. Dělí se na čtyři stupně: primární, sekundární, terciární a kvartérní. Primární struktura bílkovin je dána pořadím aminokyselin v řetězci, podmiňuje biologickou funkci bílkovin a předurčuje vyšší stupně organizace prostorového uspořádání molekuly. Sekundární struktura má dva typy: pravotočivá šroubovice (α-helix) a skládaný list (β-hřeben). Tyto struktury vznikají díky vodíkovým můstkům mezi kyslíkem z karboxylové skupiny jedné aminokyseliny a vodíkem z aminové skupiny jiné aminokyseliny. Terciární struktura je dána uspořádáním šroubovice či skládaného listu do konečného tvaru bílkoviny. A to buď do struktury fibrilární (vlákno) nebo globulární (klubko). Některé bílkoviny vykazují také strukturu kvartérní. Ta vypovídá o tom, jak jsou vzájemně uspořádány jednotlivé podjednotky bílkoviny (pokud je má). [12]

### 2.3.3 Význam enzymů

Při průběhu mnohých chemických reakcí je potřeba překonat nějakou energetickou bariéru. V případě reakcí, které probíhají v lidském těle, bývají bariéry nesmírně

vysoké. Proto je zapotřebí tyto reakce urychlit neboli katalyzovat. Většina biokatalyzátorů jsou enzymy. [13] Enzymy jsou bílkovinné povahy – jsou tvořeny bílkovinou a případně další částí, tzv. kofaktorem neboli koenzymem.

Na obrázku 5 je zobrazen detail struktury enzymu CYP121. Barevná „mašle“ je bílkovinná část enzymu.[2] Uprostřed obrázku můžeme vidět koenzym, konkrétně hemovou skupinu označenou HEM s atomem železa uprostřed[3]. Atom železa je navázán na šest dalších atomů. Prvním z nich je atom síry z aminokyseliny cystein (označeno CYS). Skrze interakci mezi atomem železa a atomem síry z této aminokyseliny je hemová skupina navázána na zbytek struktury enzymu. Součástí hemové skupiny je také tzv. porfyrin, ve kterém je samotné železo vázáno na čtyři atomy dusíku. Šestým a posledním atomem, který se na železo váže, je kyslík. Atomy vyznačené béžově jsou uhlíky. Na některé z vyobrazených atomů jsou také navázány vodíky, které však nejsou v daném kontextu důležité, a proto se nezaznačují.



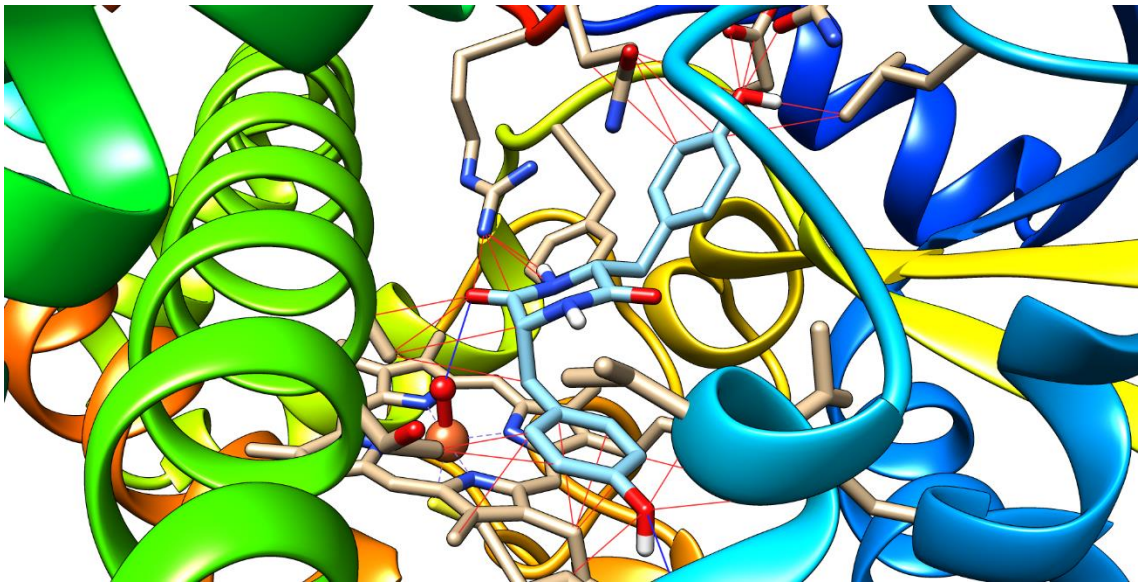

*Obr. 5: Detail struktury enzymu CYP121. [14], upraveno*

Látky vstupující do reakce s enzymem se nazývají substráty, nebo také ligandy. Aby reakce mohla proběhnout, tak se substrát musí navázat na správnou část enzymu, tzv. aktivní centrum. V aktivním centru je substrát vázán na protein pomocí slabých mezimolekulárních vazeb, jako jsou vodíkové můstky. Tyto vazby jej udržují v blízkosti koenzymu, který katalyzuje samotnou reakci.

V literatuře se občas uvádí klasická ilustrace specificity enzymu, kde je aktivní centrum enzymu přirovnáno k zámku a substrát ke klíči: aby se „zámek odemkl“ – reakce proběhla – musí „klíč zapadnout do zámku“. Čili každý enzym se podílí na katalýze pouze některých specifických reakcí a reaguje jen s některými substráty. [15]

[2] Bílkovina je barvená podle barevného spektra od tzv. N-terminální části proteinu po C-terminální část. To jsou „konce“ bílkovinné struktury.
[3] Stejná skupina s železem je obsažena i v červeném krevním barvivu hemoglobin.

Obrázek 6 vyobrazuje detail interakce ligandu (ligand je jakákoliv látka, která se váže na protein) s enzymem v aktivním centru enzymu CYP121. Atomy uhlíku ligandu jsou vyznačeny tyrkysovou, aby nesplývaly s hemovou skupinou. Atomu vodíku, které jsou navázány na kyslík nebo dusík, jsou jako jediné znázorněny, a to bílou barvou. Modrými čárami jsou znázorněny vodíkové můstky, červenými zase ostatní mezimolekulární síly. Oba typy interakcí přispívají k navázání látky k enzymu.



*Obr. 6: Interakce ligandu [19] s enzymem u aktivního jádra s hemovou skupinou. [14], upraveno*

Obrázky 5 i 6 jsem vytvořil pomocí počítačového programu UCSF Chimera [16] a AutoDock Vina [17, 18]. Strukturu enzymu jsem ve formátu pdb stáhl z Protein Data Bank, konkrétně jsem stáhl model 2IJ7. Pomocí UCSF Chimera jsem model upravil tak, aby obsahoval pouze jeden enzym a nezahrnoval původní ligandy. Dále jsem trochu pozměnil model, aby více odpovídal skutečnosti. V modelu je na atomu železa hemové skupiny navázána molekula vody. V takovémto stavu však enzym nekatalyzuje reakce, to se děje, když je na železo navázán samotný atom kyslíku dvojnou vazbou (viz kapitola 2.5.2). Pomocí online nástroje MolView, který je dostupný na webových stránkách molview.org, jsem překreslil strukturu molekuly 1 z obrázku 1 z [19] a exportoval jsem soubor ligand.mol[4]. Strukturu enzymu ve formátu pdb a strukturu ligandu ve formátu mol jsem pomocí programu AutoDock Vina upravil do jejich formátu pdbqt a provedl jsem docking. Docking je výpočetní postup, který určí, do jakého místa se ligand naváže do enzymu. Výsledkem dockingu bylo 9 pozic a orientací v prostoru ligandu okolo aktivního jádra enzymu. Nejlepší z těchto pozic ligandu jsem dále použil pro vytvoření vizualizace pomocí UCSF Chimera.

[4] Anglický systematický název této molekuly je 3,6-bis[(4-hydroxyphenyl)methyl]piperazine-2,5-dione

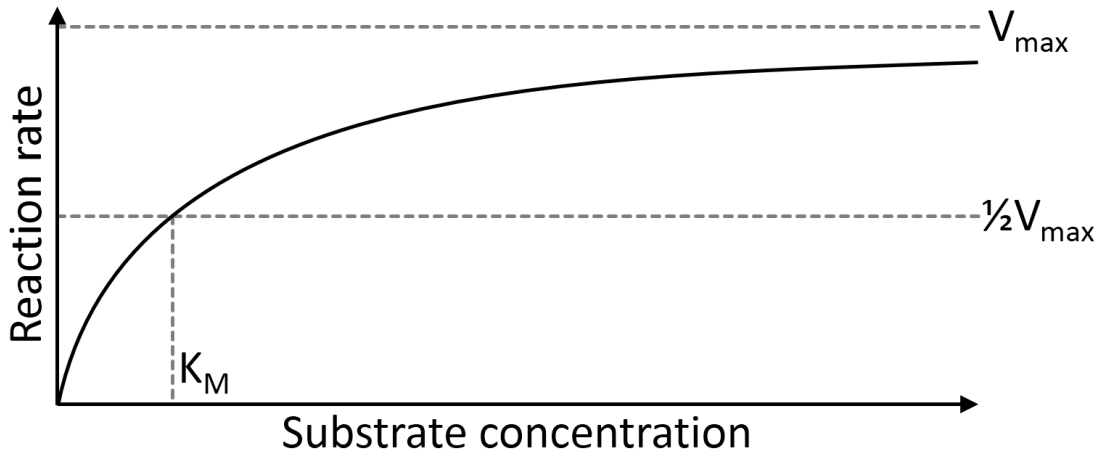
## 2.4 Enzymatická kinetika

Průběh, a především rychlost chemických reakcí katalyzovaných enzymy popisuje Michalis-Mentenova kinetika a rovnice stejného jména. Pomocí této rovnice lze velice přesně aproximovat průběh reakce katalyzované enzymem.

Michalis-Mentenova rovnice uvádí průběh obecné enzymatické reakce a lze ji zapsat následovně:

$$E + A \rightleftharpoons EA \rightarrow E + P$$

První dvě části rovnice znázorňují navázání substrátu A na enzym E za vzniku substrát-enzymového komplexu EA. Obousměrná šipka mezi těmito stavy znázorňuje, že je interakce vratná – tedy substrát se může odštěpit z enzymu, aniž by se změnila jeho struktura. Ve třetí části rovnice je opět enzym E a produkt P, který vznikl ze substrátu A, a následně se odštěpil od enzymu.




*Obr. 7: Grafické znázornění kinetiky enzymů. Na ose x je koncentrace substrátu. Na ose y je reakční rychlost. [20]*

Na obr. 7 je vyobrazena závislost rychlosti reakce katalyzované enzymem na koncentraci substrátu. $V_{max}$ je limitující rychlost, ke které se přibližuje soustava při zvyšující se koncentraci substrátu. Koncentrace, při které je dosažená rychlost rovna $\frac{1}{2}V_{max}$, se značí $K_M$. Každý enzym interaguje s každým substrátem jinak, tudíž se hodnota $V_{max}$ a $K_m$ vždy určuje pro konkrétní pár enzym-substrát. [21]

## 2.5 Cytochromy P450

Níže budu pojednávat o cytochromech P450, jaterních buňkách hepatocytech a mikrozomech. Pochopení těchto pojmů je pro mou práci důležité. Většina molekul, u kterých budu ve své práci předvídat farmakokinetické parametry, je metabolizována v játrech.

### 2.5.1 Struktura a funkce cytochromů P450

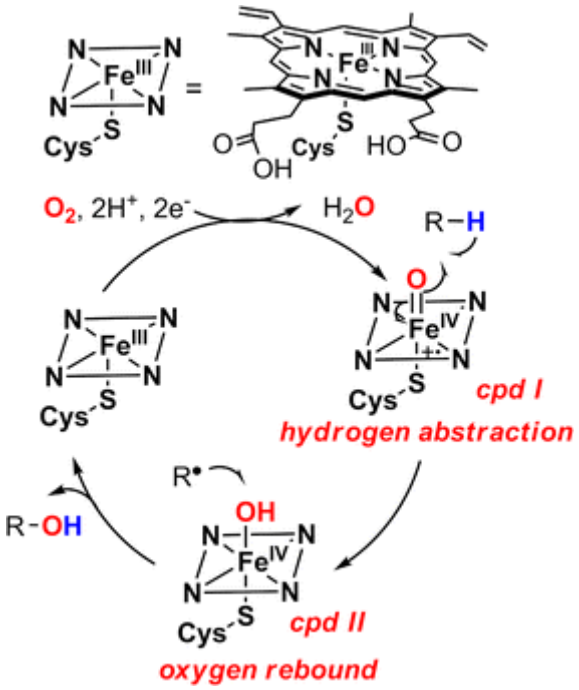
Enzymy patřící do skupiny cytochromů P450 jsou pro metabolismus léčiv jedny z nejvýznamnějších enzymů. Cytochromy P450 katalyzují (urychlují) metabolismus takřka 75 % léčiv v lidském těle. [22] Struktura cytochromů P450 obsahuje hem (viz obr. 5), ve kterém je vázané železo. Hemová skupina je ve struktuře těchto enzymů unikátně navázána přes atom síry aminokyseliny cystein z proteinové části enzymu. Tato struktura umožňuje cytochromům P450 efektivně zprostředkovávat přenos elektronů pomocí železa. Elektrony se dále využívají při katalýze reakcí. Jednou z nejvýznamnějších reakcí, které cytochromy P450 katalyzují, je monooxygenace (tj. reakce, při které dochází k navázání OH skupiny na látku). Cytochromy P450 tedy katalyzují reakci:

$$RH + O_2 + 2H^+ + 2e^- \rightleftharpoons ROH + H_2O$$

Reakce však neprobíhá celá najednou, nýbrž je to sled několika dílčích reakcí. Zápis pouze znázorňuje sumární průběh reakcí. Přísun elektronů pro monooxygenaci zprostředkovává enzym P450 reduktáza, která jako zdroj elektronů využívá molekulu NADPH. [23]

### 2.5.2 Oxygen-rebound mechanismus

Poslední krok přeměny substrátu R-H na metabolit R-OH za pomocí cytochromu P450 popisuje oxygen-rebound mechanismus. Tento mechanismus začíná až ke konci celkového katalytického cyklu P450. Před začátkem oxygen-rebound mechanismu je na centrálním atomu železa navázaný kyslík dvojnou vazbou[5]. Tato fáze je na obrázku označena jako cpd 1 (compound 1, neboli sloučenina 1). V prvním kroku se na tento kyslík naváže atom vodíku ze substrátu a zbude radikál R· s jedním volným elektronem, který se podílel na vazbě s vodíkem. Takový radikál je vysoce reaktivní, a tak se také naváže na kyslík. Nakonec se ze železa odštěpí metabolit (produkt metabolismu) R-OH. [24]




*Obr. 8: Schéma oxygen-rebound mechanismu, který probíhá na cytochromech P450. Ve vrchní části obrázku je znázorněná zjednodušená reprezentace struktury hemové skupiny cytochromů P450. [24]*

### 2.5.3 Hepatocyty a mikrozomy

Jaterní buňky hepatocyty obsahují velké množství enzymů. K nejdůležitějším z nich patří skupina zvaná mikrozomální enzymy. Mikrozomy jsou frakce buňky vzniklé při dělení buněčného materiálu centrifugací (rozdělením vzorků pomocí odstředivé síly); mikrozomy se tedy nevyskytují v živých buňkách. Pomocí zkoumání vlastností mikrozomů lze experimentálně stanovit farmakokinetické parametry (viz kap. 2.7). Mikrozomální enzymy jsou jaterní enzymy účastnící se metabolismu, zejména toxinů a léčiv. K nejdůležitějším mikrozomálním enzymům patří cytochrom P450 mikrozomy. [25]

## 2.6 Metabolismus léčiv

Metabolismus léčiv v lidském těle je rozdělen podle typu reakce na fázi 1 a fázi 2 (reakce z fáze 1 však nemusí vždy nastat před reakcí z fáze 2). Produkty těchto reakcí se lépe vylučují z těla nežli výchozí látky.

Během fáze 1 probíhají oxidační, redukční a hydrolytické reakce, které obecně zvyšují polaritu metabolitu, v porovnání se vstupní molekulou. Hlavní skupina enzymů, která zprostředkovává fázi 1 metabolismu, jsou cytochromy P450. Cytochromy P450 jsou zodpovědné za formaci zhruba 60 % primárních metabolitů (tj. pro život nezbytné látky,

[5] V enzymech, které využívám pro tvorbu svých ilustrací, byla zjištěna neobvykle dlouhá vazba mezi sírou z cysteinu a železem. Díky tomu se spíše uvádí, že se v tomto kroku na železo váže již skupina OH. [26]

jako jsou aminokyseliny nebo sacharidy) a 40 % sekundárních (tj. látky, které nejsou pro život nezbytné a nevyskytují se ve všech organismech). [27] Fází 2 metabolismu se má práce nezabývá, a proto ji zde nebudu blíže popisovat.

## 2.7 Farmakokinetika a farmakodynamika

Farmakokinetika i farmakodynamika jsou vědní obory, které se zabývají interakcí chemických látek v lidském těle. Je však mezi nimi zásadní rozdíl: farmakokinetika se zabývá efektem těla na látky, farmakodynamika se však zabývá účinky chemických látek na lidské tělo. Tato práce spadá pod obor farmakokinetiky. Chování látek v lidském těle lze studovat pomocí takzvaných farmakokinetických parametrů. Mezi farmakokinetické parametry patří mimo jiné dvě hodnoty, které popisují, jak rychle je daná látka odbourávána z těla. Těmito veličinami jsou eliminační poločas a clearance. Význam těchto hodnot je popsán v následující kapitole. [28]

### 2.7.1 Eliminační poločas a clearance

Jedny z nejdůležitějších farmakokinetických parametrů metabolismu jsou eliminační poločas (značeno $t_{1/2}$) a clearance. Obě veličiny popisují, jak bude probíhat odbourávání dané látky z těla. Vyhodnocování těchto parametrů slouží mimo jiné ke stanovení dávkování léčiv. Příliš krátký eliminační poločas, respektive vysoká hodnota clearance, by mohli vést k tomu, že by lék byl metabolizován moc brzy. Kvůli tomu by nestihl řádně účinkovat v těle pacienta. Naopak, kdyby léčivo mělo příliš vysoké $t_{1/2}$ nebo nízkou clearance, mohly by se vyskytnout nepředvídané nežádoucí vedlejší účinky.

Eliminační poločas určuje čas, za který se sníží koncentrace látky v těle na polovinu začáteční dávky. Různé látky mohou mít velice odlišné eliminační poločasy – např. noradrenalin má eliminační poločas 2 minuty [29], kdežto bedaquilin má $t_{1/2}$ 165 dní [30]. Clearance určuje množství krve, které je za jednotku času očištěno od látky. V mé práci se konkrétně jedná o jaterní clearance[6]. Jinak řečeno se jedná o míru toho, jak efektivně dokážou játra odfiltrovat danou látku z krve. Pokud se látka nedostane do moči (tudíž není vyloučena z těla, ale plně vstřebána) má nulovou clearance. [25]. Pro noradrenalin byla zjištěna clearance mezi 1,4 a 2,5 litrů za minutu [31] a pro bedaquilin 2,78 litrů za hodinu, což je asi 0,046 litrů za minutu. [32]

[6] V mé práci pracuji s hepatocytární a mikrozomální clearance. Tyto hodnoty se využívají jako aproximace jaterní clearance, jelikož jsou snáze měřitelné než samotná jaterní clearance.

# 3 METODIKA

## 3.1 Struktura TDC-ADME benchmarků

Databáze Therapeutics Data Commons neboli TDC obsahuje několik různorodých benchmarků pro strojové učení v medicíně. Benchmarky se soustředí buď na identifikaci léčiv, modelování chování léčiv v lidském těle, účinnost a bezpečnost léčiv, anebo na výrobu léků. Já používám tzv. *ADME* benchmarky. Benchmarky v této sekci obsahují malé molekuly a soustředí se na predikci jejich účinnosti a bezpečnosti. Zkratka ADME znamená Absorbtion Distribution Metabolism Excretion, česky Absorbce Distribuce Metabolismus Exkrece. Pro každou z těchto čtyř kategorií existuje v TDC-ADME několik benchmarků od různých vědeckých skupin. Ve své práci používám benchmarky z poslední kategorie – Exkrece. Tyto benchmarky jsou od vědeckých skupin Obach et al. [33] a AstraZeneca [34]. Obach et al. publikovali jeden dataset na predikci eliminačního poločasu látky na základě její sturktury. Dále ve své práci budu používat označení *obach* pro tento dataset. Společnost AstraZeneca publikovala dva datasety na clearance: *hepatocyte* a *microsome*, které budu i takto nazývat.

### 3.1.1 Data splity v TDC-ADME

Datasety v TDC jsou rozděleny do trénovacích, validačních a testovacích podmnožin pomocí různých způsobů. Pro mé datasety jsou implementovány dva z nich: random a scaffold. Random split je náhodné rozdělení dat na trénovací, validační a testovací podmnožiny. Scaffold split rozděluje data podle struktury molekul. Konkrétně je rozděluje tak, aby se molekuly v jednotlivých množinách co nejvíce lišily.

## 3.2 Použité modely strojového učení

Pro svou práci jsem zvolil pět modelů z knihovny Scikit-learn [35], které zastupují jedny z nejpoužívanějších algoritmů v problémech regrese. Python třídy, ve kterých jsou tyto modely implementovány, jsou: `ElasticNet`, `KernelRidge`, `GradientBoostingRegressor`, `RandomForestRegressor` a `MLPRegressor`. U jednotlivých modelů jsem vynechal některé hyperparametry. Konkrétně ty, které ani neoptimalizuji, ani neovlivňují hyperparametry, které optimalizuji.

### 3.2.1 Lineární regrese s elastic net regularizací

Lineární regrese je jedním z nejjednodušších typů modelů. Tento model využívá metodu nejmenších čtverců (OLS, z anglického ordinary least squares). Pomocí této metody lze vypočítat, jak dobře daná přímka nebo křivka predikuje vztah mezi jednotlivými features $X$ a $y$. K tomu optimalizuje dva parametry $w$ a $w_0$. Účelovou funkci tohoto modelu lze vyjádřit rovnicí:

$$J(w) = \frac{1}{2n_{samples}} \|y - Xw - w_0\|_2^2 + \alpha\rho\|w\|_1 + 0.5\alpha(1-\rho)\|w\|_2^2$$

kde $X$ je matice o dimenzích $n \times m$, $y$ je vektor o dimenzích $n \times 1$, $w$ a $w_0$ jsou vektory o dimenzích $m \times 1$.

První člen je standardní OLS škálované velikostí datasetu. Druhý člen obsahuje L1 (lasso) regularizaci a třetí zase L2 (ridge) regularizaci. L1 regulace je definována jako suma absolutních hodnot: $\|x\|_1 \coloneqq \sum_{i=1}^{n} |x_i|$. L2 regulace neboli Euklidovská norma/vzdálenost je definována jako odmocnina sumy čtverců hodnot: $\|x\|_2 \coloneqq \sqrt{\sum_{i=1}^{n} x_i^2}$.

Ve svém programu optimalizuji dva hyperparametry tohoto modelu: $\alpha$ a $\rho$ (pro $\rho$ se ve Scikit-learn používá označení `l1_ratio`). Hyperparametr $\alpha$ určuje celkovou sílu regularizace (jak L1, tak L2). Vyšší hodnota $\alpha$ způsobí silnější regularizaci, potenciálně vedoucí k jednodušším modelům. $\rho$ kontroluje vyvážení mezi L1 a L2 regularizací. Při $\rho = 1$ je regularizace čistě L1. Pro $\rho = 0$ je regularizace čistě L2. Pro hodnoty mezi 0 a 1 jde o kombinaci obou forem regularizace.

Dále má tento model ještě jeden hyperparametr – `fit_intercept`. Je to boolean, který určuje, zda bude v účelové funkci použit intercept parametr $w_0$, nebo zda bude tento parametr roven 0. Pokud je `fit_intercept=True`, pak hodnota $w_0$ je optimalizována zároveň s $w$. Hodnotu hyperparametru `fit_intercept` však ve svém programu neoptimalizuji – jeho hodnota je vždy `True`.

### 3.2.2 Ridge regrese s kernel trikem

Ridge regrese je druh OLS, který také implementuje L2 ridge regularizaci a tzv. kernel trik sloužící k tomu, aby model byl schopen naučit se nelineární závislosti v datech pomocí lineárních funkcí. Tohoto lze docílit využitím speciálního druhu funkce, tzv. kernelu, značeno $\mathbb{K}$. Vstupem této funkce jsou vždy dva řádky z matice $X$, potažmo ten samý řádek dvakrát. Účelovou funkci KRR (kernel ridge regression) lze vyjádřit následovně:

$$J(w) = \frac{1}{2}\sum_{i=1}^{n}[y_i - \sum_{j=1}^{n}\alpha_j \mathbb{K}(x_i, x_j)]^2 + \frac{\alpha}{2}\sum_{j=1}^{n}\alpha_j^2$$

$\alpha$ zde značí – obdobně jako u lineární regrese – sílu regularizace, a tudíž je prvním hyperparametrem.

U KRR optimalizuji také další dva hyperparametry: `kernel` a `gamma`. Hyperparametr kernel určuje jaká funkce bude v modelu použita jako kernel. V mé implementaci využívám tři různé typy kernelu: `linear`, `laplacian` a `rbf`. Lineární kernel je z nich nejjednodušší a neimplementuje hyperparametr `gamma`. Je definovaný jako $\mathbb{K}_{linear}(x_i, x_j) = x_i^T \cdot x_j$. Laplaceovský kernel však regularizační hyperparametr `gamma` implementuje, a to následovně: $\mathbb{K}_{laplacian}(x_i, x_j) = \exp\left(-\frac{\|x_i - x_j\|_2}{\gamma}\right)$. Velikost $\gamma$ určuje rychlost exponenciálního klesání funkční hodnoty kernelu. Poslední druh kernelu, který ve svém programu používám, je tzv. Gaussovský kernel neboli radiální basická funkce (RBF): $\mathbb{K}_{RBF}(x_i, x_j) = \exp\left(-\gamma\|x_i - x_j\|_2^2\right)$. U tohoto kernelu hyperparametr $\gamma$ mění dopad Euklidovské vzdálenosti datových bodů na výslednou hodnotu kernelu.

Hlavní výhodou KRR a podobných modelů je, že dokážou aproximovat nelineární závislosti v datech pomocí transformování dat do vícedimenzionálního prostoru pomocí kernelu.

### 3.2.3 Posílená regrese rozhodovacího stromu

Základní myšlenkou posílené regrese, anglicky boosting, je využití několika jednoduchých modelů – tzv. souboru slabých žáků (z angl. ensemble of weak learners). Tyto modely jsou však samy o sobě sotva více přesné než hádání. Slabí žáci také nepredikují data přímo, nýbrž jsou trénováni na predikcích předchozího modelu ze souboru. Jako slabí žáci jsou v modelu posílené regrese rozhodovacího stromu, jak již název napovídá, rozhodovací stromy. V mém programu označuji tento model zkratkou GB (z angl. gradient boosting). Algoritmus lze rozdělit do čtyř kroků:

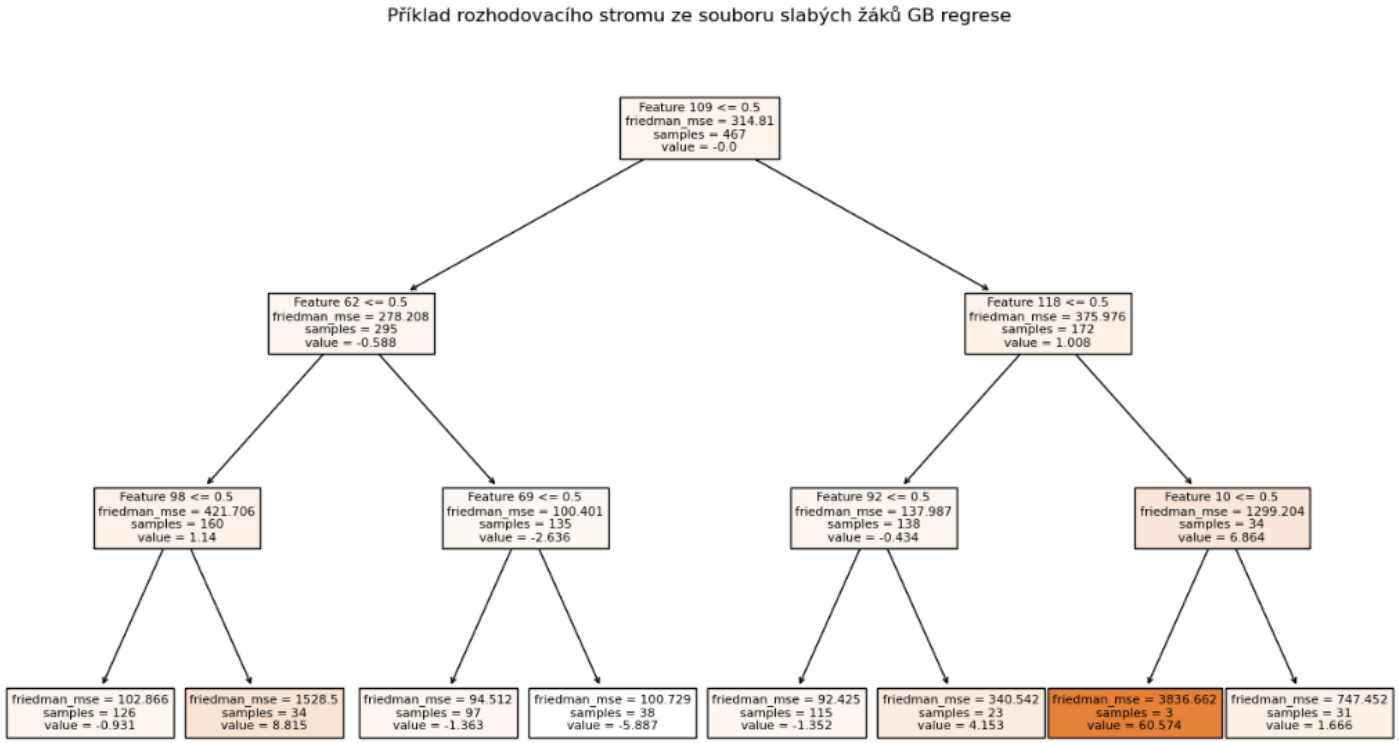
1. Pomocí jednoduchého modelu uděláme prvotní predikci dat. Tímto modelem je často konstantní funkce rovna aritmetickému průměru nebo mediánu trénovacích hodnot.

2. Reziduální hodnoty skutečných hodnot a predikcí předchozího modelu jsou použity na trénování dalšího modelu. Predikce modelu jsou určovány na základě rozhodovacího stromu (způsob generování stromu je popsán níže).
3. Krok dva je několikrát opakován, počet iterací udává hyperparametr `n_estimators`. Každý další slabý žák opravuje chyby souboru předchozích modelů.
4. Výsledná predikce je suma predikcí všech slabých žáků, každá z predikcí je vážená hyperparametrem `learning_rate`.

Generování rozhodovacího stromu během druhého kroku probíhá následovně: algoritmus vybírá nejlepší možný rozklad dat do dvou skupin na základě hodnoty reziduálních chyb. To znamená, že hledá takovou podmínku (např. „je-li hodnota feature_1 menší než X, pak jdeme do levého podstromu, jinak do pravého“), pomocí které model co nejefektivněji predikuje vstupní hodnoty. Hloubku stromu (počet rozdělení dat na polovinu) udává hyperparametr `max_depth`, který zde může být přirozeným číslem, anebo `None`. Pokud je `None`, pak jsou všechny větve rozdělovány, dokud do výsledné kategorie nespadá méně prvků než předem určený limit. S tímto limitem v mém programu nepracuji – ponechávám jeho předdefinovanou hodnotu.

Predikované hodnoty pro nová data jsou poté získány tak, že každý datový bod projde stromem od kořene k jednomu z listových uzlů. V každém listovém uzlu je přidána hodnota, která odpovídá průměrné reziduální hodnotě v daném uzlu. Celková predikce modelu GB pro daný datový bod je pak součtem příspěvků všech listových uzlů, které datový bod prošel.

Na obrázku 9 je graficky znázorněn jeden z rozhodovacích stromů posílené regrese rozhodovacího stromu. Tento konkrétní model byl trénovaný na train splitu z obach, bez škálování eliminačních poločasů. Vytvořil jsem jej jen pro demonstraci, protože při použití škálovaných eliminačních poločasů jsou hodnoty na diagramu příliš malé. Každý uzel reprezentuje jednu skupinu dat. Podmínka „Feature číslo <= 0.5“ určuje podle hodnoty daného bitu ECFP, které molekuly spadají do daného splitu. Konečná predikce je součtem hodnot označených jako „value“ v uzlech, kterými molekula projde. Ještě jednou zdůrazním, že predikce tohoto modelu není přímo eliminační poločas molekuly. Nýbrž se jedná o hodnotu, kterou by nejspíš predikoval předchozí slabý žák. Friedman_mse je míra toho, jak heterogenně daný split rozděluje data. Tato hodnota by ideálně byla co nejnižší, protože poté by split byl homogenní. To by znamenalo, že by pro podobné molekuly byla predikována stejná hodnota. Uzly jsou vybarveny podle velikosti Friedman_mse – čím vyšší hodnota, tím sytější odstín.




*Obr. 9: Grafické znázornění struktury jednoho z rozhodovacích stromů v GB regresi.*

### 3.2.4 Regrese náhodného lesa

Obdobně jako předchozí model využívá regrese náhodného lesa (RF) soubor slabých žáků a každý z těchto žáků je taktéž rozhodovací strom. Mezi GB a RF je však několik klíčových rozdílů:

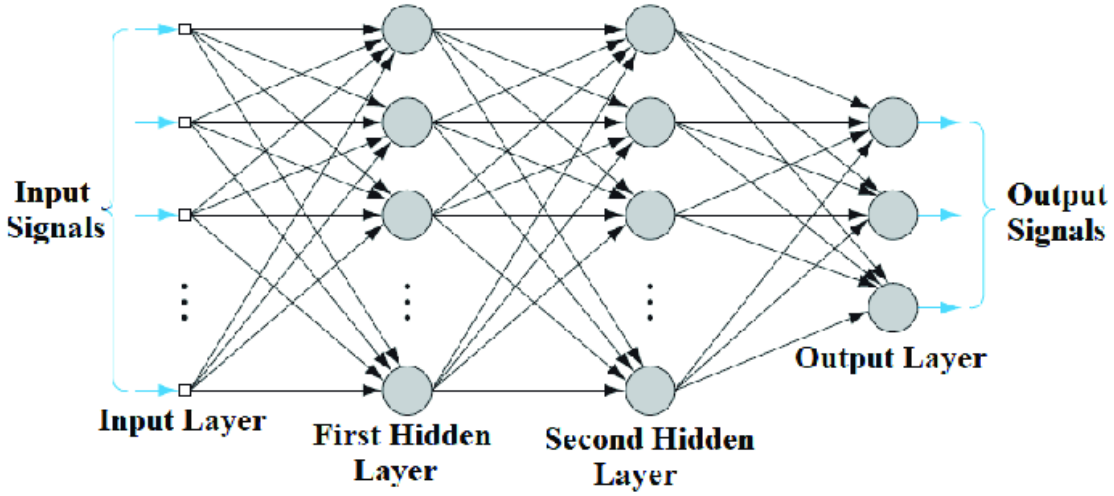
| | Posílená regrese rozhodovacího stromu (GB) | Náhodný les (RF) |
|---|---|---|
| Trénovací data | Každý strom je trénován na predikcích předchozího stromu | Každý strom je trénován na části dat (tzv. bootstrapping) |
| Konečná predikce | Suma predikcí, které jsou vážené hyperparametrem `learning_rate` | Průměr všech predikcí |
| Features použity pro generování stromu | Všechny features | Náhodně zvolené podmnožiny features |
| Paralelizace během trénování | Stromy jsou typicky trénovány sekvenčně | Stromy lze trénovat paralelně |

*Tab. 1: Klíčové rozdíly mezi posílenou regresí rozhodovacího stromu GB a náhodným lesem RF.*

V mém programu optimalizuji tři hyperparametry tohoto modelu: `n_estimators`, `max_depth` a `max_features`. `n_estimators` a `max_depth` mají obdobný význam jako u GB. `max_features` v mém programu může být buď „`sqrt`“, nebo „`log2`“. Tento hyperparametr určuje část ze všech features, které budou použity při generování jednotlivých stromů; množství těchto features bude tedy $\sqrt{n\_features}$, nebo $\log_2(n\_features)$. Pokud tyto hodnoty nejsou celým číslem, pak množství features, které jsou zvoleny pro každý split je $\lfloor max\_features \cdot n\_features\_in\_ \rfloor$, kde $n\_features\_in\_$ je počet features ve vstupních datech.

### 3.2.5 Vícevrstvý perceptron

Vícevrstvý perceptron (MLP) je typ dopředné umělé neuronové sítě, který lze přirovnat k lidskému mozku. Tento model se skládá z několika vrstev, přičemž každá vrstva obsahuje uzly, nazývané neurony. Analogicky si lze představit, že jednotlivé neurony v první vrstvě MLP fungují jako receptory, přijímající vstupní informace. Tyto signály jsou následně přenášeny skrz váhy do následujících (tzv. skrytých) vrstev, což odpovídá zpracování informací v mozku. Během učení jsou váhy mezi neurony aktualizovány tak, aby síť lépe predikovala závislou proměnnou na základě vstupních dat. Je důležité poznamenat, že "signál" mezi neurony je ve skutečnosti reálné číslo, a výstup každého neuronu je vypočítán nelineární funkcí součtu jeho vstupů. Neuronové sítě jsou schopny efektivně rozpoznávat a zpracovávat složité vzory v datech.




*Obr. 10: Příklad vícevrstvého perceptronu s dvěma skrytými vrstvami [36]*

## 3.3 Optuna

Pro optimalizaci hyperparametrů jsem využil open-source knihovnu Optuna. [37] Studie v Optuně reprezentuje jedno spuštění optimalizačního procesu. Během studie jsou zkoumány různé body v hyperparametrickém prostoru s cílem nalézt konfiguraci, která maximalizuje nebo minimalizuje danou účelovou funkci.

*Trial* je jednotlivý experiment uvnitř studie, který zahrnuje konkrétní nastavení hyperparametrů a vyhodnocení odpovídajícího výsledku účelové funkce. Během studie jsou spuštěny různé trialy s různými kombinacemi hyperparametrů, a Optuna se snaží nalézt optimální konfiguraci.

Samplers jsou algoritmy, které určují, jakým způsobem budou zkoumány různé body v hyperparametrickém prostoru. V mém případě jsem se rozhodl použít `TPESampler` (Tree-structured Parzen Estimator Sampler). [38] Tento algoritmus prochází prostor možných hodnot hyperparametrů tak, že se soustředí na okolí předešlých konfigurací se slibnými výsledky. Díky tomu může model rychleji konvergovat k optimálním hodnotám hyperparametrů, ale zároveň není výpočetně náročný.

Optuna dále využívá pruners. Ve svém programu implementuji `HyperbandPruner`. Pruners jsou mechanismy, které selektivně ukončují neefektivní experimenty, čímž urychlují celkový proces optimalizace. `HyperbandPruner` implementuje hyperband algoritmus, který vybírá slibné experimenty a spouští je s různými alokačními schématy výkonu počítače, aby určil ten nejlepší trial.

## 3.4 Směrodatná odchylka a směrodatná odchylka chyb

Pro každý trial jsem vyhodnocoval dvě klíčové metriky: RMSD a STD. Směrodatná odchylka chyb RMSD (root-mean-square deviation) vyjadřuje odmocninu průměrné odchylky mezi predikcemi modelu $\hat{y}$ a skutečnými hodnotami $y$: $RMSD = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y - \hat{y})^2}$. STD (standard deviation) je směrodatná odchylka, která udává míru variability predikcí: $STD = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(|y_i - \bar{y}|^2)}$.

V rámci jednoho trial jsem provedl tři běhy s různými trénovacími a validačními daty, abych získal robustnější výsledky. Každý běh obsahoval pět cross-validačních splitů, a nakonec jsem spočítal průměrné hodnoty RMSD. STD jsem počítal pouze z predikcí všech dat, tedy až po křížové validaci. RMSD jsem dále používal jako hlavní metriku pro porovnávání výsledků jednotlivých modelů na konkrétním datasetu.

## 3.5 Spearmanův korelační koeficient a koeficient determinace

Pro další zkoumání přesnosti modelů jsem používal Spearmanův korelační koeficient a koeficient determinace $R^2$.

Spearmanův korelační koeficient je statistická míra, která kvantifikuje sílu a směr monotónního vztahu mezi dvěma proměnnými. Na rozdíl od Pearsonova korelačního koeficientu nezávisí na lineárním charakteru vztahu a je schopen detekovat i nelineární vztahy. Hodnota Spearmanova korelačního koeficientu je v rozsahu od -1 do 1, kde -1 znamená úplně negativní monotónní vztah, 1 znamená úplně pozitivní monotónní vztah a 0 znamená žádný monotónní vztah. Matematicky je Spearmanův korelační koeficient $\rho$ definován následovně: $\rho = 1 - \frac{6\sum_{i=0}^{n} d_i^2}{n(n^2-1)}, d_i = rank(y_i) - rank(\hat{y}_i)$. Funkce $rank(a_i)$ je definována jako index prvku $a_i \in A$, kde $A$ je vektor, ve kterém jsou hodnoty seřazené dle velikosti tak, že největší je na první pozici a nejmenší na pozici poslední. Rád bych zdůraznil, že tato funkce je odlišná od hodnosti matice, která se standardně také značí jako $rank$.

Koeficient determinace neboli $R^2$ vyjadřuje, jak dobře se model přizpůsobuje datům. $R^2$ je poměr mezi rozptylem vysvětleným modelem a celkovým rozptylem dat: $R^2 = 1 - \frac{\sum_{i=1}^{n}(y-\hat{y})^2}{\sum_{i=1}^{n}(y-\bar{y})^2}$. Hodnota $R^2$ je v rozsahu od 0 do 1, kde 0 značí, že model nijak nevysvětluje variabilitu dat, a 1 značí, že model plně vysvětluje variabilitu.

## 3.6 Reprezentace molekul pro strojové učení

Teoreticky je možné provádět strojové učení přímo pomocí struktury molekul ve formátu SMILES. Nevidím však důvod trénovat modely přímo na SMILES. Systém SMILES má příliš složitou strukturu, než aby bylo možné aplikovat standardní postupy. Domnívám se, že by takto trénovaný model neměl reálné využití, jelikož konverze SMILES do jiného formátu je vesměs triviální.

### 3.6.1 Struktura molekul pomocí extended-connectivity fingerprintů

První z nich je tzv. extended-connectivity fingerprint (ECFP). Pro ECFP se také používá označení Morgan fingerprint nebo circular fingerpring. ECFP je bit vektor o dané délce. Každý bit v ECFP reprezentuje určitou podstrukturu v molekule. Zde je třeba zdůraznit, že tyto podstruktury nemusí být shodné s chemickými funkčními skupinami. Algoritmus generování ECFP lze shrnout ve třech krocích:

1. Přiřazení identifikátoru v podobě přirozeného čísla každému nevodíkovému atomu molekuly. Toto číslo je generováno pomocí hashovací funkce na základě vlastností atomu, jako jsou např.: o jaký prvek se jedná, na kolik dalších atomů je navázaný atd.
2. Identifikátory každého atomu jsou iterativně upravovány podle atomů, na které je navázán.

3. Odstranění identifikátorů ekvivalentních částí molekuly. Dvě části molekuly jsou ekvivalentní, pokud obsahují přesně identické množiny vazeb a jejich hashové identifikátory jsou stejné.

Výsledné pole identifikátorů projde konečnou hashovací funkcí, která pomocí něj vygeneruje bit vektor, tj. ECFP[7]. [8, 39]

### 3.6.2 Vlastnosti molekul pomocí molekulárních deskriptorů

Dalším způsobem strojového učení, pomocí kterého jsem trénoval modely, byly molekulární deskriptory. To jsou experimentálně zjištěné hodnoty, které popisují konkrétní chemické nebo fyzikální vlastnosti molekuly anebo její strukturu. Ve své práci jsem využil šest deskriptorů: sdx, sdc, sa, dga, dgp a dgtot.

První tři parametry, které jsem pro molekuly vypočítal pomocí Python knihovny Jazzy, jsou sdx (X-H donor strength), sdc (C-H donor strength) a sa (acceptor strength). Všechny tři parametry popisují, jak se daná molekula bude při tvoření vodíkových vazeb nejspíše chovat. Jestliže bylo pro molekuly vypočítáno vysoké sdx, tak molekula bude obsahovat funkční skupiny, které jsou dobrými donory ve vodíkových můstcích. Mezi takové skupiny patří ty, ve kterých je vodík navázaný na prvek s vysokou elektronegativitou. Mohou to být například sekundární aminy ($R_1$-NH-$R_2$, kde $R_1$ a $R_2$ jsou uhlovodíkové zbytky) nebo hydroxylové (-OH) skupiny. Sdc také značí, že molekula obsahuje vodíky, které jsou dobrými donory ve vodíkových můstcích. Tyto vodíky jsou ale navázány na uhlík. Vysoké sa znamená, že molekula celkově obsahuje skupiny, které jsou dobrými akceptory ve vodíkových můstcích. Mezi takové skupiny patří např. primární aminy ($R_1$-$NH_2$), hydroxylové skupiny nebo třeba amidové skupiny (-$CONH_2$).

Další tři parametry popisují, jak molekula interaguje s vodou při hydrataci. Dga (apolar contribution to ΔG of hydration) měří, jak nepolární části molekuly přispívají k energetice hydratace. Oproti tomu dgp (polar contribution to ΔG of hydration) zkoumá, jak polarita molekuly ovlivňuje energetiku hydratace. Dgtot (Total ΔG of Hydration) popisuje, jak nepolární a polární části molekuly společně určují energetiku hydratace molekuly. [40]

[7] ECFP může také označovat pole identifikátorů před hashovací funkcí. V mé práci používám označení ECFP výhradně pro bit vektor.

# 4 PRAXE

Veškerý můj kód – včetně dat, grafů a obrázků – je dostupný na GitHubu na následujícím odkazu: https://github.com/hruska-lab/metabolism-hyperparameter-optimization. Kořenová složka repozitáře obsahuje skripty. Také jsem pro účely této práce napsal modul `cytochrome_P450.ipynb`. Tento modul obsahuje mé vlastní funkce, které jsem využíval v kódu na více místech, a je spolu s ostatními potřebnými soubory obsažen ve složce project_resources. Abych mohl svůj modul importovat a zároveň jej mohl nechat jako Jupyter notebook, musel jsem napsat ještě separátní modul `import_utils.py`. Pomocí tohoto modulu mohu importovat samotný notebook `cytochrome_P450.ipynb`. Struktura repozitáře je blíže popsána v README, které jsem uvedl také v apendixu.

## 4.1 Příprava eliminačního poločasu a clearance látek pro strojové učení

Před užitím farmakokinetických parametrů látek na trénování modelů byly hodnoty upraveny tak, aby nejmenší hodnota v datasetu byla 0, největší 1, a všechny ostatní spadaly mezi ně. Po této úpravě nelze hodnoty interpretovat v kontextu medicíny, ztrácí tedy měrné jednotky. Tohoto jsem docílil pomocí třídy `MinMaxScaler` ze Scipy [41]. Tato třída zachovává relativní vzdálenosti mezi maximální, minimální, a každou další hodnotou. To znamená, že relativní vzdálenosti mezi hodnotami, které nejsou krajní, nebudou zachovány. Bude zachován pouze jejich poměr k minimální a maximální hodnotě. Ze začátku jsem místo konečné implementace s `MinMaxScaler` používal přirozený logaritmus eliminačního poločasu/clearance. Je to totiž jeden ze standartních způsobů, jak zmenšit rozmezí features, a také jej bylo triviální implementovat. Tento postup měl však dvě nevýhody. První nevýhodou bylo, že po přirozeném logaritmu bylo rozmezí hodnot stále relativně velké. Kromě toho také přirozený logaritmus nezachovává žádné poměry mezi hodnotami; ani mezi maximální, respektive minimální a zbytkem hodnot. Bez jakékoliv úpravy byl největší rozdíl mezi maximální a minimální hodnotou v datasetu obach konkrétně mezi hodnotami 4,11 a 855,3. Po přirozeném logaritmu se tyto hodnoty změní na asi 1,41 a 6,75.

## 4.2 Příprava struktur molekul pro strojové učení

Napříč všemi metodami strojového učení, které ve své práci využívám, se jako target features uplatňují eliminační poločasy pomocí třídy `MinMaxScaler`. Jako vstupní features využiji struktury molekul převedené do jedné ze dvou reprezentací – extended-connectivity fingerprint, nebo molekulární deskriptory.

### 4.2.1 Implementace ECFP

Ve svém kódu používám ECFP algoritmus, který je implementován ve funkci `rdkit.Chem.AllChem.GetMorganFingerprintAsBitVect`. Pomocí tohoto algoritmu z RDKit [42] jsem vytvořil funkci `fp_from_smiles`. Vstupem této funkce

je pole SMILES řetězců, které se převedou na korespondující ECFP, a funkce poté vrátí pole s vygenerovanými ECFP. Funkci `GetMorganFingerprintAsBitVect` využívám s následujícími parametry: `useChirality=True, radius=2, nBits=124`. První parametr určuje, zda při generování ECFP bude použita i stereochemie (tj. uspořádání molekuly v 3D prostoru). Defaultně má tento parametr hodnotu `False`. Jenže 3D uspořádání molekul může (obzvláště v medicíně) mít velký dopad na vlastnosti látky. Kvůli tomu jsem se rozhodl tento parametr použít. Parametr radius určuje tzv. cutoff poloměr, který při generování fingerprintu vymezuje, jak velká část molekuly bude reprezentována jedním bitem. Jelikož všechny molekuly, se kterými pracuji, jsou relativně malé, postačí mi cutoff poloměr roven dvěma. Kdybych pracoval s většími molekulami, zvolil bych větší poloměr a k tomu adekvátní množství bitů v poli. Dalším důvodem, proč jsem dal přednost menší velikosti pole, je ten, že se na nich díky jejich relativní jednoduchosti modely trénují rychleji. Pro získání přesnějších výsledků by bylo třeba zvolit větší cutoff poloměr a k němu adekvátně velký bit vektor, což by ale vyžadovalo velmi zvýšit výpočetní náročnost. Domnívám se, že rozdíl v přesnosti modelu v těchto dvou případech by byl malý.

### 4.2.2 Molekulární deskriptory

Dalším způsobem strojového učení vlastností molekul, který jsem používal pro trénování modelů, byly molekulární deskriptory. Ty byly vypočítány prostřednictvím knihovny Jazzy. Ke každé molekule jsem vypočítal šest hodnot, které ji charakterizují, a to sdx, sdc, sa, dga, dgp a dgtot.

Pro práci s molekulárními deskriptory z Jazzy je v mém kódu určen jeden soubor – `Jazzy-data-prep.ipynb` – a jedna funkce – `cytochrome_P450.parse_jazzy_df`. V notebooku jsem implementoval generování samotných molekulárních deskriptorů využitím funkce `jazzy.api.molecular_vector_from_smiles`. Tato funkce však vrací slovník. Output jsem musel zpracovat a výsledné molekulární deskriptory jsem spolu s target hodnotami zapsal do csv souborů. Pokud pro molekulu deskriptory nelze vygenerovat, vrátí tato funkce `JazzyError`. Tento error se mi nepodařilo explicitně zachytit pomocí `except JazzyError`, takže jsem musel použít jen holý `except`. Funkce `parse_jazzy_df` slouží k tomu, abych mohl snáze přečíst vygenerované molekulární deskriptory z csv souborů.

## 4.3 Strojové učení a optimalizace hyperparametrů

V `TDC-ADME.ipynb` využívám Optunu pro optimalizaci hyperparametrů. Optunu jsem implementoval ve třídě `cytochrome_P450.HyperparamTuner`.

### 4.3.1 Strojové učení benchmarků TDC-ADME

V první buňce souboru `cytochrome_P450` importuji knihovny, včetně funkcí `fp_from_smiles`, `parse_jazzy_df` a třídy `HyperparamTuner`. V dalších pěti buňkách deklaruji proměnné, které používám vícekrát, a načítám data pro strojové učení. Ve druhé buňce mimo jiné deklaruji slovníky samplerů a prunerů, které Optuna nabízí, abych je mohl snáze používat ve zbytku kódu. V následujících buňkách postupně načítám TDC-ADME benchmarky, škáluji target features, převádím SMILES molekul do ECFP a načítám molekulární deskriptory z csv souborů.

Kód v poslední buňce zajišťuje samotné využití Optuny. Jako první stanovím druh sampleru a pruneru, který má být při studii využíván, a určím tzv. `n_trials`. Tento parametr vymezuje, kolik bude provedeno iterací při hledání optimálních hyperparametrů. Následuje samotné trénování modelů. To jsem implementoval s multithreadingem pomocí `ThreadPoolExecutor` z modulu `concurrent.futures`, který je součástí základní instalace Pythonu. Následné tři *for cykly* projdou všechny kombinace druhu features (ECFP, molekulární deskriptory), benchmarků (obach, microsome, hepatocyte) a modelů (linear, KRR, GB, RF, ANN). Abych mohl využívat multithreading s Optunou, musel jsem použít méně častý postup při ukládání studie. Standardně se pro tento účel využívá databáze, jako je SQLite. Kvůli multithreadingu však mohou procesy končit nesynchronně, a proto by tento postup nefungoval. Z tohoto důvodu používám třídy `JournalStorage` a `JournalFileStorage` z Optuny, které během studie zapisují její průběh do log souboru.

### 4.3.2 HyperparamTuner

Tato třída obsahuje 5 metod: `sample_params`, `cross_validation_splits`, `evaluate`, `train_test_return`, `objective`; a 6 parametrů: `log_csv_path`, `model_identifier`, `X_train`, `y_train`, `X_val` a `y_val`. V první metodě se pomocí sampleru určí hyperparametry pro danou iteraci modelu. Metoda vrátí třídu konkrétního modelu inicializovanou se zvolenými hodnotami hyperparametrů. Metoda `cross_validation_splits` slouží k přípravě dat pro křížovou validaci. Vrací pole šesti n-tic, kde prvních pět obsahují trénovací a validační data pro křížovou validaci. Poslední prvek n-tice jsou samotná trénovací a testovací data. Metoda `train_test_return` provede křížovou validaci modelu se zvolenými hyperparametry. Po posledním foldu se také ukládají predikce modelu. Tato iterace se provádí pomocí posledního prvku n-tice vygenerované funkcí `cross_validation_splits`. Celý proces se provede třikrát, během každého cyklu se získají predikce modelu na testovacích datech. Z těchto predikcí se vypočítá RMSD a STD pro každou iteraci, a tyto hodnoty se následně zprůměrují. Pomocí funkce `optuna_trial_logging` se do csv souboru zapíše číslo trialu, hyperparametry modelu, průměrné RMSD, průměr predikcí a průměrné STD. Poslední metoda `objective` nejprve volá `sample_params`, poté `train_test_return`

s inicializovaným modelem, a vrátí stejnou hodnotu, kterou vrátí metoda `train_test_return`.

## 4.4 Interaktivní inference

Pro demonstraci toho, že modely lze aplikovat i ve skutečném světě, jsem připravil interaktivní inferenci. Uživatel si zde může zvolit předtrénovaný model. Lze zvolit libovolná trénovací data, tj. libovolnou kombinaci druhu features, datasetu a modelu. Pro účely inference jsem modely trénoval na veškerých datech – trénovacích, validačních i testovacích. Uživatel následně zadá látku, u které by chtěl predikovat hodnotu. Zvolil-li uživatel dataset obach, bude predikován eliminační poločas látky. Pokud však zvolil hepatocyte nebo microsome, model bude predikovat clearance. V souboru `inference-model-prep.ipynb` jsou modely trénovány a následně ukládány. Samotná inference je implementována v `interactive-inference.ipynb`. Demo slouží k prokázání, že strojové učení lze využívat i pro nové, neznámé látky.

# 5 VÝSLEDKY

## 5.1 Důležitosti hyperparametrů

Prvním z výstupů mé práce je porovnání důležitosti jednotlivých hyperparametrů modelů pro jednotlivé datasety. Tato veličina je automaticky vypočítávána Optunou během studie. Po ukončení studie lze tyto důležitosti získat pomocí funkce `optuna.visualization.plot_param_importances` nebo také `get_param_importances`, pokud uživatel chce vytvořit vlastní vizualizaci. Zvolil jsem funkci `plot_param_importances`, protože grafy, které tato funkce vytvoří, jsou pro mě dostačující.

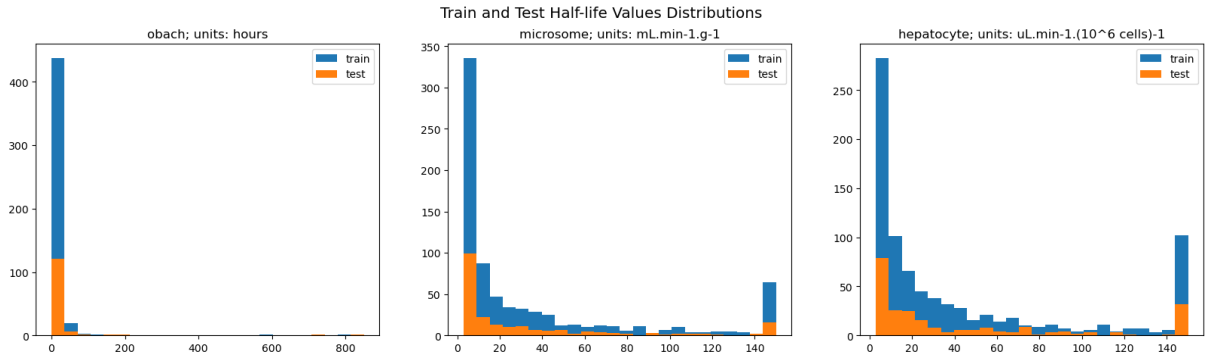

*Obr. 11: Důležitosti hyperparametrů všech modelů pro dataset obach.*


*Obr. 12: Důležitosti hyperparametrů všech modelů pro dataset microsome.*

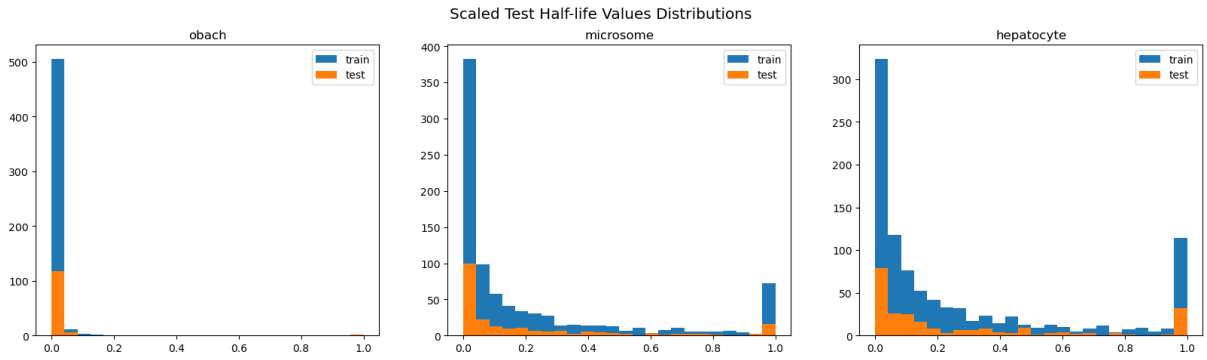

*Obr. 13: Důležitosti hyperparametrů všech modelů pro dataset hepatocyte.*

## 5.2 Distribuce hodnot eliminačních poločasů a clearance

Zkoumal jsem distribuce hodnot eliminačního poločasu v jednotlivých datasetech, a to jak před škálováním, tak po něm. Obach měl před škálováním výrazně větší rozpětí hodnot: od 0,065 po 850,0; oproti minimální hodnotě 3,0 a maximální 150,0 u microsome i hepatocyte. Všechny tři datasety jsou nerovnoměrné. Microsome a hepatocyte mají podobnou distribuci, v obou datasetech je po škálování asi 50 % hodnot menších než 0,1. U obach je však 96 % procent hodnot menších než 0,1 a 50 % hodnot je menších než 0,005. Podobně je tomu i u percentilu. U microsome je osmdesátý percentil roven 0,35, pro hepatocyte má srovnatelnou hodnotu, a to 0,54. Obach se od zbylých dvou datasetů v tomto také liší, u něj byl osmdesátý percentil roven 0,021, tedy asi dvacetkrát méně. Distribuci eliminačních poločasů jsem prověřoval i zjišťováním, jaké procento hodnot je menších než 0,1 a jaká je hodnota osmdesátého percentilu.




*Obr. 14: Distribuce hodnot eliminačního poločasu před škálováním. Jednotky v grafech jsou postupně: hodiny, $mlmin^{-1}g^{-1}$, $\mu lmin^{-1}(106\ buněk)^{-1}$.*



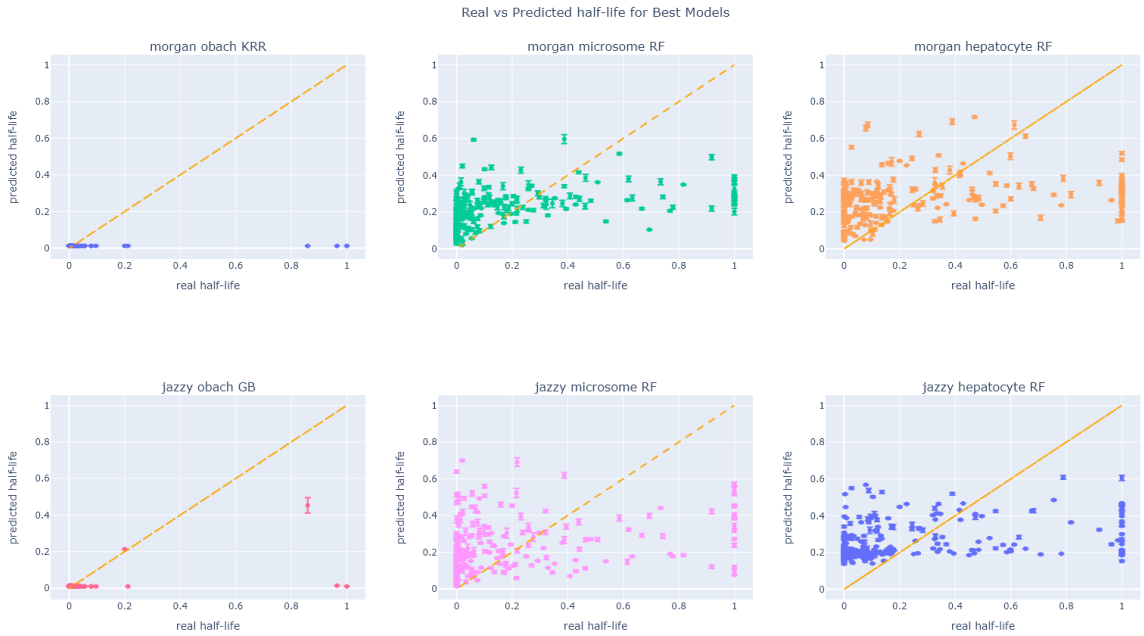

*Obr. 15: Distribuce škálovaných hodnot eliminačního poločasu (bez jednotek).*

| | Obach | Microsome | Hepatocyte |
|---|---|---|---|
| Procento škálovaných hodnot menších než 0,1 | 96,15 % | 56,82 % | 46,91 % |
| Hodnota osmdesátého percentilu (zaokrouhleno na tři desetinná čísla) | 0,021 | 0,347 | 0,537 |

*Tab. 2: Porovnání distribuce škálovaných hodnot v jednotlivých datasetech.*

## 5.3 Analýza vztahu skutečných a predikovaných hodnot

Vytvořil jsem korelační diagramy predikovaných a skutečných hodnot eliminačního poločasu. U hodnot, pro které je definována standardní deviace, je také znázorněna. Vyobrazená diagonála značí, kde by veškerá data v ideálním případě ležela.



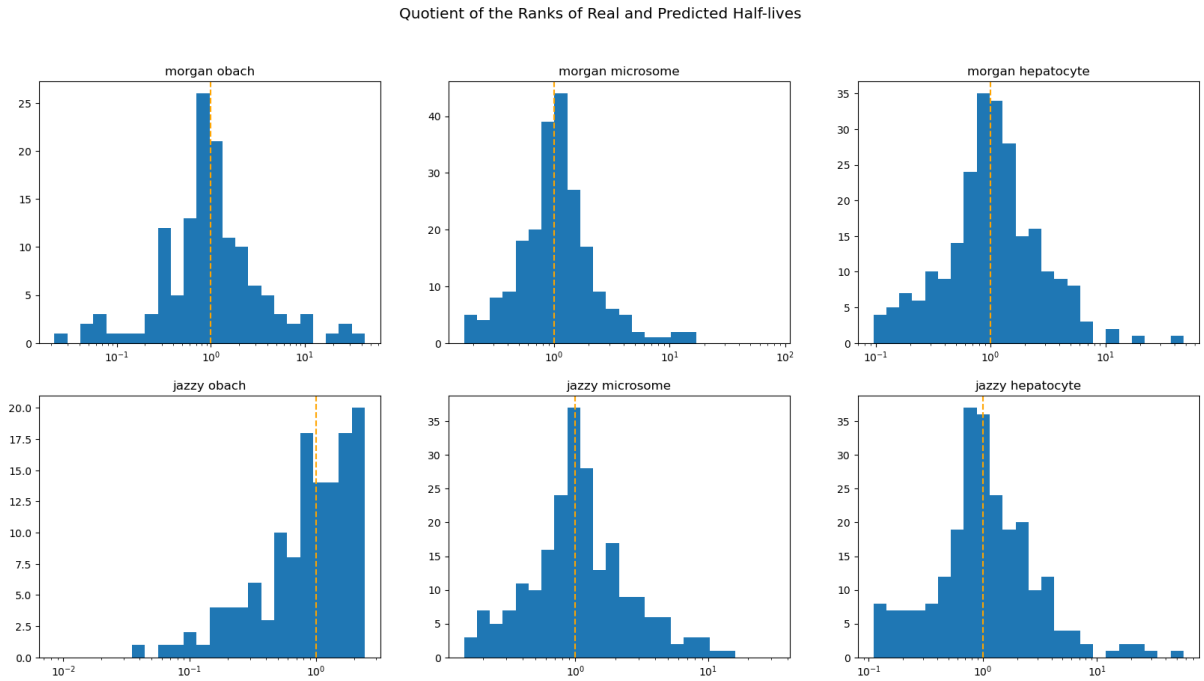

*Obr. 16: Vztah predikovaných a reálných hodnot eliminačního poločasu.*

### 5.3.1 Poměr mezi skutečnými a predikovanými pozicemi

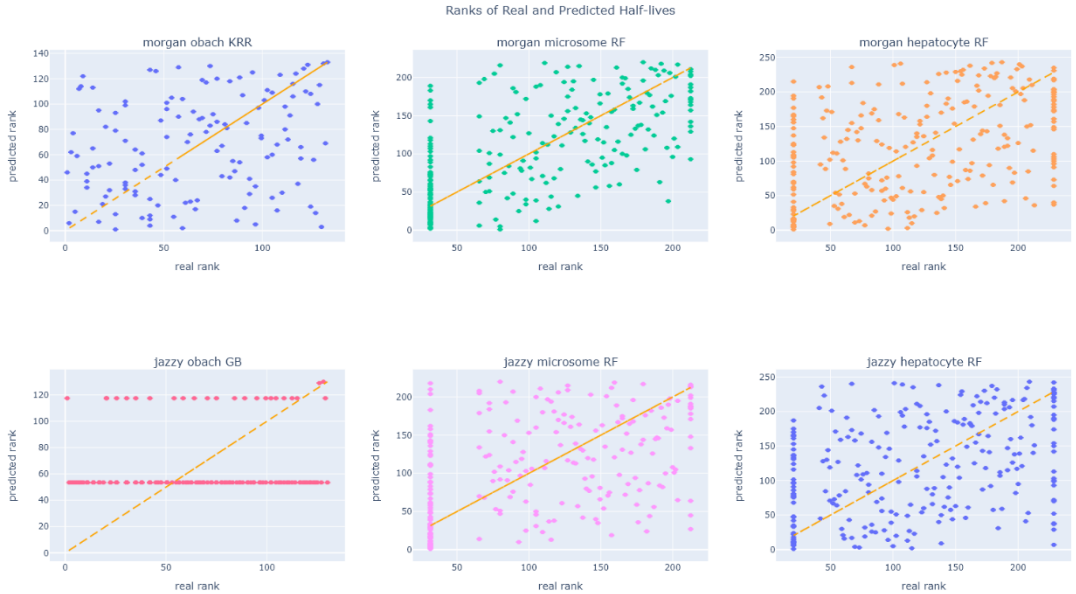
Pomocí poměru mezi pozicemi skutečných a predikovaných hodnot eliminačního poločasu můžeme zkoumat přesnost predikcí modelu. Porovnával jsem pořadí reálných a predikovaných hodnot, oba seznamy hodnot byly přitom seřazeny podle velikosti. Aby byla lépe vidět distribuce hodnot této veličiny, vyobrazil jsem hodnoty na histogramech s logaritmickou osou x. Pokud by se predikované hodnoty shodovaly s hodnotami skutečnými, byly by v identickém pořadí, a poměr mezi jejich pozicemi by byl roven jedné. Proto jsem hodnotu jedna na ose x na histogramech označil oranžovou čárou, a také jsem vypočítal průměrnou absolutní deviaci od jedné. V ideálním případě by byla tato hodnota rovna nule.

| | Obach | Microsome | Hepatocyte |
|---|---|---|---|
| Morgan | 1,809 | 1,192 | 1,349 |
| Jazzy | 0,569 | 1,128 | 1,560 |

*Tab. 3: Průměrné absolutní deviace od jedné mezi pozicemi skutečných a predikovaných pozic hodnot.*




*Obr. 17: Histogramy všech hodnot poměru mezi pozicemi skutečných a predikovaných eliminačních poločasů. Osa x je logaritmická. Oranžovou vertikální čárou je označena hodnota x=1.*

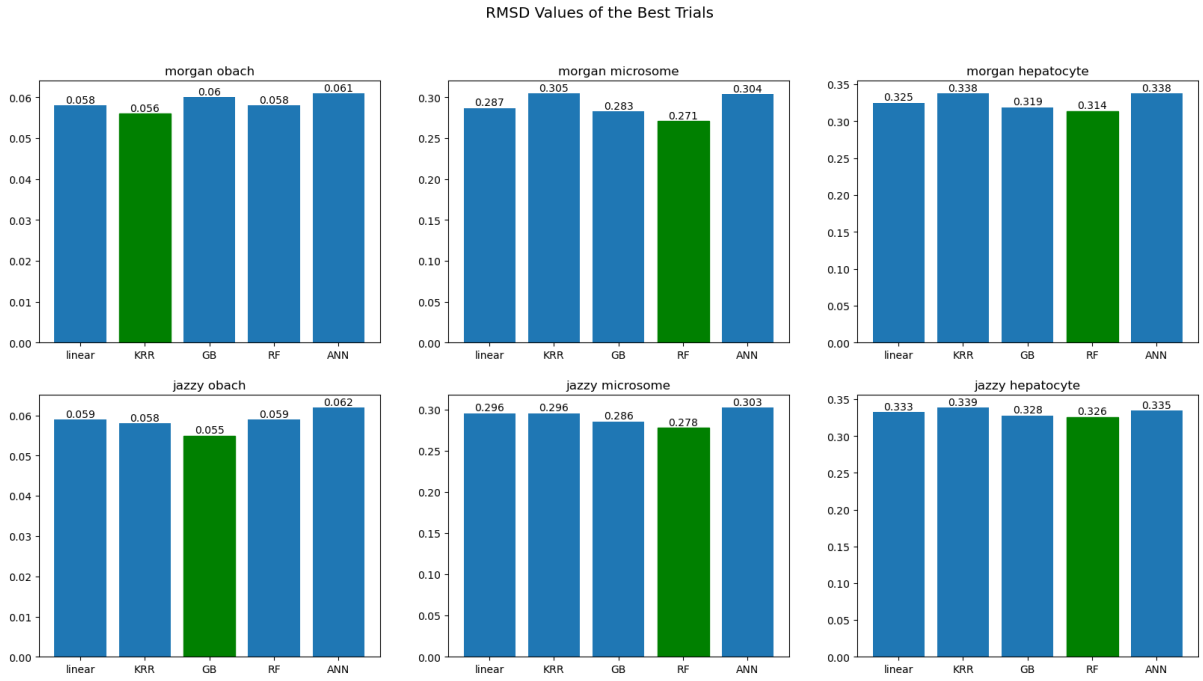



*Obr. 18: Porovnání pořadí skutečných a predikovaných hodnot eliminačního poločasu. Oranžová diagonála značí, kde by pořadí bylo u skutečných i predikovaných hodnot shodné.*

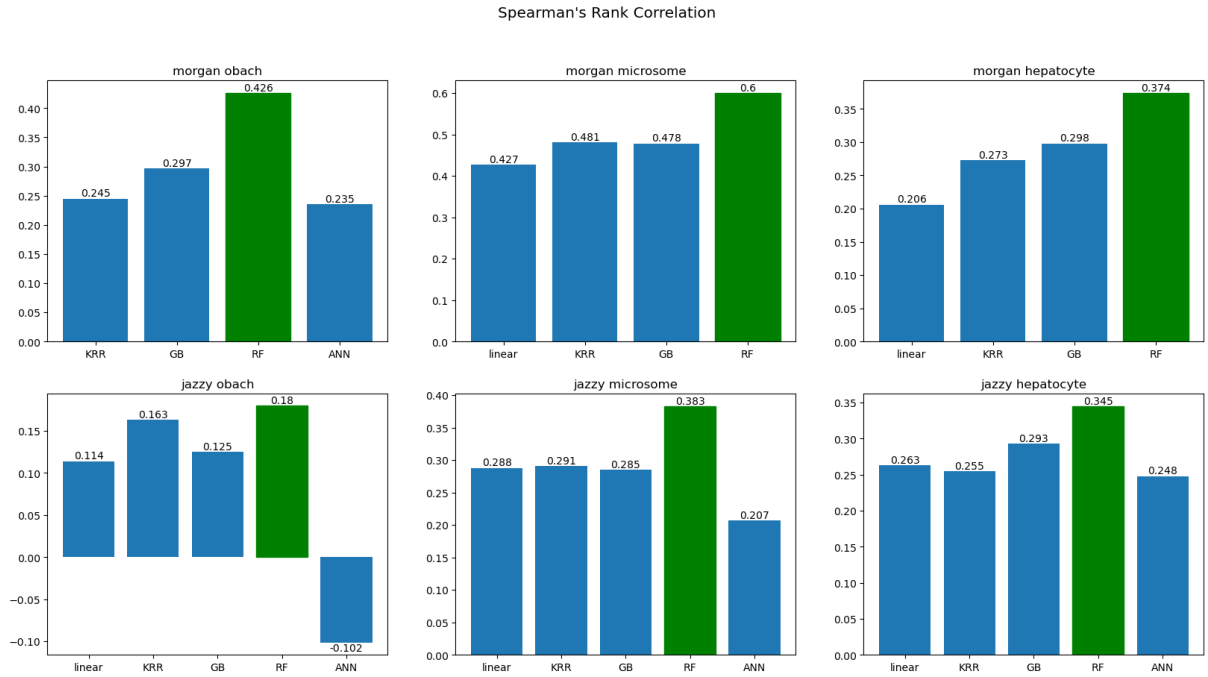
## 5.4 Celková přesnost modelů

Přesnost predikcí modelů jsem zkoumal pomocí tří různých metrik: odmocnina střední hodnoty čtverce chyby (RMSD z anglického *root-mean-square deviation*), Spearmanův koeficient pořadové korelace a koeficient determinace (neboli $R^2$).

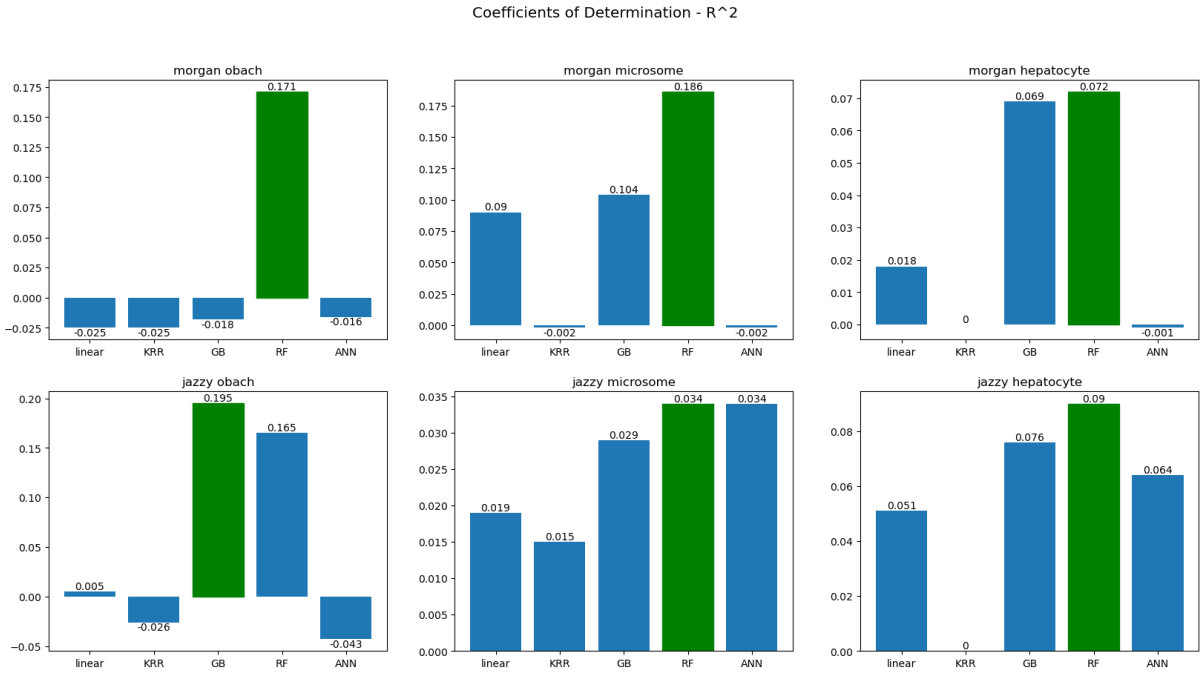
RMSD se lišilo nejvíce mezi Obach a datasety od Astra-Zeneca (tj. microsome a hepatocyte). Všechny modely měly na všech datasetech srovnatelné výsledky, avšak GB a RF měly konzistentně nízké RMSD, a ANN jedno z nejvyšších. Spearmanův koeficient měl vždy nejvyšší RF. Koeficient determinace měl takřka vždy nejlepší také RF, až na jazzy obach, kde jej mělo nejvyšší GB. U některých datasetů chybí hodnota Spearmanova korelačního koeficientu, a to buď u lineární regrese, nebo ANN. Model se naučil konstantní funkci, tzn. pro každý testovací datový bod predikoval stejnou hodnotu. V takovém případě není Spearmanův korelační koeficient definovaný.




*Obr. 19: Hodnoty RMSD pro všechny modely u všech datasetů; model s nejlepším RMSD je označený zeleně.*



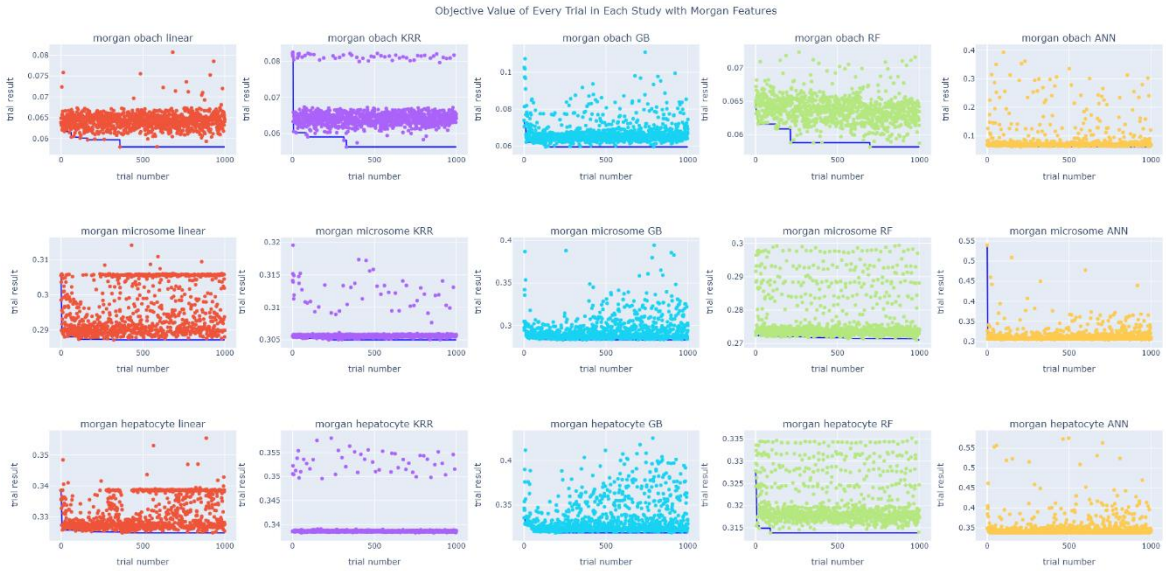

*Obr. 20: Velikosti Spearmanova koeficientu mezi skutečnými a predikovanými hodnotami; model s nejlepším koeficientem je označený zeleně.*



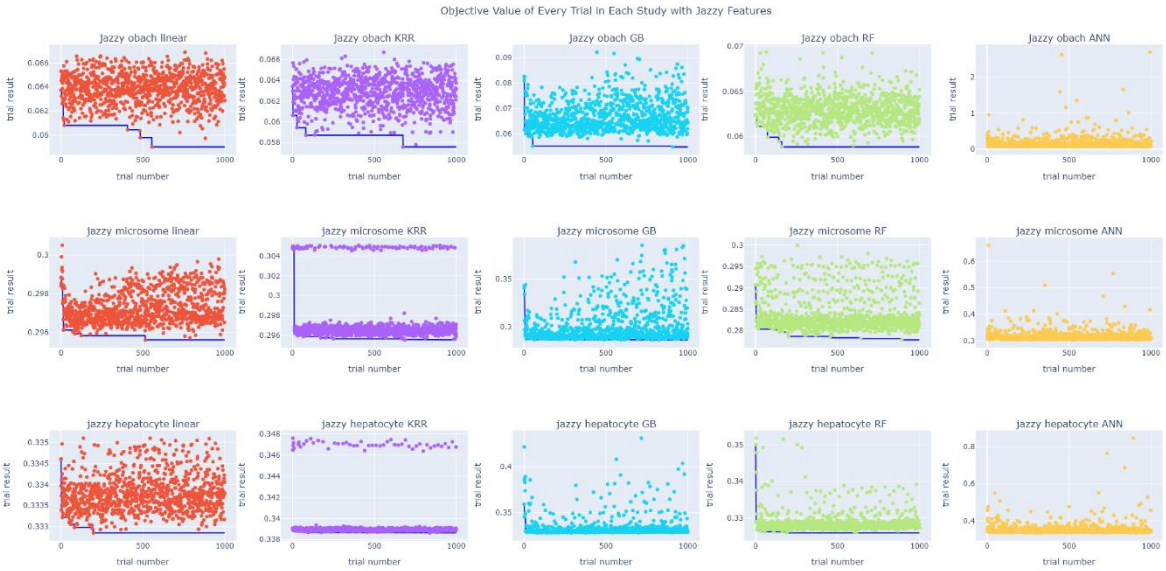

*Obr. 21: Hodnoty koeficientu determinace R2.*

## 5.5 Výsledky všech Optuna trials

Celkový průběh studie lze zkoumat pomocí grafu výsledných hodnot jednotlivých trials, tj. jaké RMSD měl model s konkrétními hyperparametry. V tomto grafu je podstatné, jak se v průběhu studie vyvíjí nejnižší hodnota, která byla některým trial dosažena. Tyto hodnoty jsou v grafu spojeny čárou.




*Obr. 22: Shrnutí všech trials pro všechny modely na všech datasetech s Morgan features.*




*Obr. 23: Shrnutí všech trials pro všechny modely na všech datasetech s Jazzy features.*

# 6 DISKUZE

Existuje mnoho způsobů, jak přistupovat ke strojovému učení s modely ze Scikit-learn. Ve své práci jsem nejprve využíval pouze nástroje dostupné v samotném Scikit-learn i pro optimalizaci hyperparametrů. Poté jsem pro tento účel implementoval Optunu, a u té jsem již zůstal. Existuje i třetí způsob, a to manuální optimalizace na základě zasvěceného odhadu. Každý z těchto přístupů má své výhody a nevýhody, a hodí se pro jiné situace.

Zasvěcený odhad je vhodný při trénování velkých modelů, u kterých může trvat trénování i několik měsíců. V takovém případě je zasvěcený odhad logicky jedinou praktickou možností. Když však trvá natrénování jednoho modelu maximálně několik desítek sekund, pak má optimalizace hyperparametrů smysl. Dále je zde otázka, zda využít nástroje dostupné přímo v Scikit-learn, nebo využít knihovnu určenou přímo pro optimalizaci hyperparametrů, jako je Optuna.

Scikit-learn umožňuje pomocí krátkého programu relativně lehce implementovat optimalizaci hyperparametrů s `GridSearchCV`, viz funkce `param_tuning` a `mol_predict_and_std` v `cytochrome_P450.ipynb`. Předpokládal jsem, že tyto výsledky nejsou optimální, a lze je zlepšit implementací sofistikovanějšího algoritmu optimalizace hyperparametrů. Přesnost predikcí modelů se mezi iteracemi poměrně výrazně lišila, což podpořilo můj předpoklad. Pro přístup s Optunou i bez ní jsem prověřoval Spearmanovy koeficienty pořadové korelace pro dataset obach s ECFP. Na těchto datech měly modely bez použití Optuny následující výsledky: KRR 0,213; GB 0,280; RF 0,259; ANN 0,128. Tyto hodnoty můžeme porovnat s levým horním grafem z obrázku 18. KRR a GB mají srovnatelné hodnoty. Kdežto ANN a zvláště RF mají s Optunou značně vyšší hodnotu Spearmanova koeficientu.

Dále lze diskutovat o výpočetní, respektive časové náročnosti obou přístupů. Časová náročnost je dvojího druhu: jak dlouho trvá sestavit funkční program a jak dlouho je potřeba, aby byl zpuštěný, aby přinesl adekvátní výsledky. Vzhledem k tomu, že pro plné využití nástrojů dostupných skrze Optunu je třeba napsat delší program, můžeme usoudit, že jen Scikit-learn bude z tohoto hlediska méně časově náročný. Z pohledu samotného času, kdy má kód běžet, jsou tyto dva přístupy ekvivalentní. Algoritmický výběr hyperparametrů zabere nepatrné množství času oproti tomu, jak dlouho trvá natrénovat model. U Optuny je naopak mnohem lehčí v tomto ohledu škálovat program – stačí pouze upravit hodnotu parametru `n_trials`. V mém kódu měl tento parametr hodnotu 1000, jelikož jsem se domníval, že do té doby by měl program najít optimální hyperparametry. Jak se však ukázalo, postačila by mnohem nižší hodnota `n_trials`, a výsledky by byly jen o málo horší. Tento fakt lze vyvodit z obrázků 22 a 23, konkrétně z toho, že čára spojující nejlepší trials je velice strmá a velmi rychle klesne. Z tohoto důvodu by bylo vhodnější použít nižší hodnotu `n_trials` (např. v rozmezí 200 až 500) a provést více iterací všech studií s různým rozmezím hyperparametrů.

Celkově si myslím, že je Optuna velice dobrý nástroj pro optimalizaci hyperparametrů. Zasvěcený odhad nebo optimalizace hyperparametrů pomocí Scikit-learn mohou být lepšími variantami pro jiné účely, než byla moje práce.

Optuna nepodporuje jen modely ze Scikit-learn. Já sám jsem ji implementoval také pro NequIP, jenže z časových důvodů jsem musel tuto část mé práce nechat nedokončenou.

Kromě zkoumání, zda je Optuna vhodná pro aplikaci strojového učení v medicíně, jsem porovnával, jak velký dopad mají jednotlivé hyperparamtery na výkon modelu. To se měří pomocí tzv. důležitostí hyperparametrů. Ty každému hyperparametru přiřazují procentuální hodnotu, která reprezentuje, jak velký vliv má hyperparametr na model.

Důležitosti hyperparametrů `ElasticNet` se neliší podle typu features – liší se jen dataset od datasetu. Pro obach s ECFP je poměr důležitostí `l1_ratio` ku `alpha` 1:3. Pro stejný dataset, ale s molekulárními deskriptory, je jejich poměr obrácený, tedy 3:1. U ostatních datasetů s libovolným typem features jsou důležitosti v poměru 1:1. Z toho vyplývá, že při optimalizaci hyperparametrů pro `ElasticNet` třeba se soustředit jak na `l1_ratio`, tak na `alpha`, a pro oba hyperparametry mít podobně velké rozsahy search space.

U kernel ridge regrese jsem očekával, že nejdůležitějším hyperparametrem bude kernel. Avšak nečekal jsem, o kolik vyšší bude jeho důležitost oproti hyperparametrům `alpha` a `gamma`. Ověřoval jsem frekvenci různých typů kernelu, které byly voleny samplerem během optimalizace. Nejčastěji byly voleny `laplacian` a `rbf`. Oba tyto kernely implementují hyperparametr `gamma`, proto bych se domníval, že i `gamma` bude mít dopad na výslednou přesnost modelu. Je možné, že jsem zvolil špatné rozmezí search space pro `gamma`. Jen v jednom případě nebyl `kernel` jasně nejdůležitějším hyperparametrem, a to u datasetu obach s molekulárními deskriptory. V tomto případě byla jeho důležitost zhruba stejná, jako důležitost `alpha`.

Ze začátku mi nebylo jasné, jak konstruovat třídu pro Optunu, nebo co přesně dělá sampler a pruner. Po pozorném přečtení dokumentace a zhlédnutí příkladů na jejich GitHubu jsem těmto problémům porozuměl. Proto bych doporučil Optunu jako nástroj pro optimalizaci hyperparametrů, i když uživatel nemá s těmito postupy žádné předchozí zkušenosti. Pro rychlou a snadnou optimalizaci hyperparametrů pro malé množství modelů, nezáleží-li přitom na nalezení optimálních hyperparametrů, postačí čistě Scikit-learn.

# 7 ZÁVĚR

Cílem mé práce bylo využít Optunu pro optimalizaci hyperparametrů modelů ze Scikit-learn a stanovit, zda je tento postup vhodný pro predikci vlastností léčiv. Díky robustnější optimalizaci hyperparametrů jsem oproti zasvěcenému odhadu zlepšil přesnost modelů asi o polovinu. Avšak vzhledem k časové a výpočetní náročnosti spolu s relativně nízkou přesností modelů se domnívám, že budoucnost strojového učení v medicíně bude založena spíše na velkých modelech s manuálně nastavenými hyperparametry. Postupy podobné tomu mému, kde se natrénuje větší množství různých jednoduchých modelů, by mohly sloužit spíše ke stanovení, jaký algoritmus strojového učení by mohl být vhodný pro konkrétní aplikaci.

# 8 SLOVNÍČEK POJMŮ A SEZNAM POUŽITÉ NOTACE A ZKRATEK

## 8.1 Pojmy týkající se strojového učení

Hyperparametr – jeden ze členů účelové funkce, určuje průběh učení modelu, uživatel je zadá před trénováním modelu.

Křížová validace – opakování validace, pokaždé s jinou podmnožinou trénovacích dat jako validační dataset.

Model – počítačový program, který je schopen rozpoznat zákonitosti v datech a/nebo data predikovat.

Parametr – jeden ze členů účelové funkce modelu, určuje se během procesu učení, uživatel jej přímo neovlivní.

Regrese – metoda pro zkoumání zákonitosti mezi nezávislou proměnnou (features) a závislou proměnnou (targets).

SMILES – způsob zapsání struktury molekuly do řetězce

Strojové učení – oblast umělé inteligence, soustředí se na vývoj statistických modelů pro předvídání vývoje dat.

Testování – provádí se po natrénování modelu na odlišných datech, tzv. testovací data/dataset, pomocí testování zjišťujeme, jak dobře, a také jak obecně model predikuje data.

Trénování – proces, při kterém se využívá sada dat, tzv. trénovací data/dataset, aby se model „naučil rozpoznávat zákonitosti mezi daty“.

Účelová funkce – matematický zápis modelu; jsou v ní obsaženy parametry, hyperparametry a vstupní data, a také s nimi pracující matematické operace.

Umělá inteligence – druh programů, které v nejširším slova smyslu napodobují člověka. Například strojové vidění, velké jazykové modely.

Validace – trénování modelu jen na části trénovacího datasetu a následné otestování modelu na zbytku tohoto datasetu – tj. validační data/dataset.

## 8.2 Pojmy týkající se medicíny

Aktivní centrum – oblast enzymu, kde dochází ke katalyzované reakci. Může se zde nacházet kofaktor.

Aminokyseliny – skupina molekul, základní stavební jednotky bílkovin

Clearance – množství látky, o kterou je očištěno lidské tělo za jednotku času.

Cytochromy P450 – skupina jaterních enzymů s podobnou strukturou, které katalyzují stejné nebo podobné reakce.

Docking – druh simulace, jehož cílem je predikovat jakým způsobem se ligand naváže na enzym

Eliminační poločas – doba, za kterou se sníží koncentrace látky v těle na polovinu.

Enzym – skládá se z bílkoviny a případně kofaktoru, působí jako katalyzátor biochemických reakcí.

Katalyzátor – chemická látka, které snižuje energii potřebnou pro průběh chemické reakce, a tím ji urychluje.

Kofaktor – nebílkovinná látka nebo iont, který je nutný, aby určité enzymy mohly působit jako katalyzátory.

Ligand – jakákoliv látka, která se váže na protein.

Metabolit – produkt metabolismu.

Substrát – konkrétní molekula, která reaguje při reakci katalyzované enzymem.

Vodíkový můstek – vazba mezi vodíkem jedné molekuly a jiným atomem sousední molekuly

## 8.3 Seznam použité notace

Python objekty (funkce, třídy apod.) a soubory s kódem jsou označeny fontem `Courier New` (např. `ElasticNet`, `gamma`).

$\hat{y}$ predikce modelu

$\bar{y}$ průměrná hodnota vektoru

$J(w)$ účelová funkce modelu s parametrem $w$

$\|x\|_1$ … L1 lasso regularizace

$\|x\|_2$ … L2 ridge regularizace

$\alpha$ … hyperparametr `alpha`, určuje sílu regularizace

$\rho$ … hyperparametr `l1_ratio`, určuje poměr L1 a L2 regularizace; nebo také Spearmanův koeficient pořadové korelace

$\mathbb{K}$ … kernelová funkce

$A^T$ … transpozice matice

$\gamma$ … hyperparametr gamma, určuje chování některých kernelů

$rank(a_i)$ … index prvku $a_i \in A$, kde $A$ je vektor, ve kterém jsou hodnoty seřazené dle velikosti tak, že největší je na první pozici a nejmenší na pozici poslední

## 8.4 Seznam použitých zkratek

AI … Umělá inteligence; Artificial Intelligence

ML … Strojové učení; Machine Learning

TDC … Therapeutics Data Commons

TDC-ADME … TDC-Absorbtion Distribution Metabolism Excretion; česky Absorbce Distribuce Metabolismus Exkrece

CYP … Cytochromy P450

KRR … Ridge regrese s kernel trikem; Kernel Ridge Regression

GB … Posílená regrese rozhodovacího stromu; Gradient Boosting

RF … Regrese náhodného lesa; Random Forest

ANN … Umělá neuronová síť; Artificial Neural Network

RMSD … Směrodatná odchylka chyb; Root-Mean-Square Deviation

STD … Směrodatná odchylka; Standard Deviation

$R^2$ … Koeficient determinace

ECFP … Extended-Connectivity Fingerprint

# 9 SEZNAM OBRÁZKŮ A TABULEK

## 9.1 Seznam obrázků

Obr. 1: Ukázka, jak algoritmus lineární regrese predikuje data. Rovnice y = 2.77x + 4.22 popisuje vztah mezi x a y, který vznikl jako predikce modelu. 2.77 a 4.22 jsou konkrétní hodnoty dvou parametrů tohoto modelu. ..... 8
Obr. 2: Grafické znázornění 5-fold křížové validace. [6] ..... 9
Obr. 3: Obecná struktura aminokyseliny. [10] ..... 10
Obr. 4: ilustrace jednotlivých úrovní uspořádání proteinu; a) primární; b) sekundární (vlevo α-helix, vpravo β-hřeben); c) terciární; d) kvartérní. [11] ..... 11
Obr. 5: Detail struktury enzymu CYP121. [14], upraveno ..... 12
Obr. 6: Interakce ligandu [19] s enzymem u aktivního jádra s hemovou skupinou. [14], upraveno ..... 13
Obr. 7: Grafické znázornění kinetiky enzymů. Na ose x je koncentrace substrátu. Na ose y je reakční rychlost. [20] ..... 14
Obr. 8: Schéma oxygen-rebound mechanismu, který probíhá na cytochromech P450. Ve vrchní části obrázku je znázorněná zjednodušená reprezentace struktury hemové skupiny cytochromů P450. [24] ..... 16
Obr. 9: Grafické znázornění struktury jednoho z rozhodovacích stromů v GB regresi. ..... 22
Obr. 10: Příklad vícevrstvého perceptronu s dvěma skrytými vrstvami [36] ..... 24
Obr. 11: Důležitosti hyperparametrů všech modelů pro dataset obach. ..... 32
Obr. 12: Důležitosti hyperparametrů všech modelů pro dataset microsome. ..... 33
Obr. 13: Důležitosti hyperparametrů všech modelů pro dataset hepatocyte. ..... 34
Obr. 14: Distribuce hodnot eliminačního poločasu před škálováním. Jednotky v grafech jsou postupně: hodiny, $mlmin^{-1}g^{-1}$, $\mu lmin^{-1}(106\ buněk)^{-1}$. ..... 35
Obr. 15: Distribuce škálovaných hodnot eliminačního poločasu (bez jednotek). ..... 35
Obr. 16: Vztah predikovaných a reálných hodnot eliminačního poločasu. ..... 36
Obr. 17: Histogramy všech hodnot poměru mezi pozicemi skutečných a predikovaných eliminačních poločasů. Osa x je logaritmická. Oranžovou vertikální čárou je označena hodnota x=1. ..... 37
Obr. 18: Porovnání pořadí skutečných a predikovaných hodnot eliminačního poločasu. Oranžová diagonála značí, kde by pořadí bylo u skutečných i predikovaných hodnot shodné. ..... 37
Obr. 19: Hodnoty RMSD pro všechny modely u všech datasetů; model s nejlepším RMSD je označený zeleně. ..... 38
Obr. 20: Velikosti Spearmanova koeficientu mezi skutečnými a predikovanými hodnotami; model s nejlepším koeficientem je označený zeleně. ..... 38
Obr. 21: Hodnoty koeficientu determinace R2. ..... 39
Obr. 22: Shrnutí všech trials pro všechny modely na všech datasetech s Morgan features. ..... 40
Obr. 23: Shrnutí všech trials pro všechny modely na všech datasetech s Jazzy features. ..... 40

## 9.2 Seznam tabulek

Tab. 1: Klíčové rozdíly mezi posílenou regresí rozhodovacího stromu GB a náhodným lesem RF. ..... 23
Tab. 2: Porovnání distribuce škálovaných hodnot v jednotlivých datasetech. ..... 35
Tab. 3: Průměrné absolutní deviace od jedné mezi pozicemi skutečných a predikovaných pozic hodnot. 36

# 10 REFERENCE

[1] Pushpakom, S., Iorio, F., Eyers, P. A., Escott, K. J., Hopper, S., Wells, A., Doig, A., Guilliams, T., Latimer, J., McNamee, C. et al. (2019), „Drug repurposing: progress, challenges and recommendations“, Nature Reviews Drug Discovery 18(1), 41-58.

[2] Stokes, J. M., Yang, K., Swanson, K., Jin, W., Cubillos-Ruiz, A., Donghia, N. M., MacNair, C. R., French, S., Carfrae, L. A., Bloom-Ackerman, Z. et al. (2020), „A deep learning approach to antibiotic discovery“, Cell 180(4), 688-702.

[3] J. Jumper et al., „Highly accurate protein structure prediction with AlphaFold“, Nature, roč. 596, č. 7873. Springer Science and Business Media LLC, s. 583–589, čvc. 15, 2021. doi: 10.1038/s41586-021-03819-2.

[4] M. Varadi et al., „AlphaFold Protein Structure Database in 2024: providing structure coverage for over 214 million protein sequences“, Nucleic Acids Research, roč. 52, č. D1. Oxford University Press (OUP), s. D368–D375, lis. 02, 2023. doi: 10.1093/nar/gkad1011.

[5] C. Arnold, L. Biedebach, A. Küpfer, a M. Neunhoeffer, „The role of hyperparameters in machine learning models and how to tune them“, Political Science Research and Methods. Cambridge University Press (CUP), s. 1–8, úno. 05, 2024. doi: 10.1017/psrm.2023.61.

[6] BENNER, Jonas. Cross-Validation and Hyperparameter Tuning: How to Optimise your Machine Learning Model. Online. In: Towardsdatascience.com 6. 8. 2020. Dostupné z: https://towardsdatascience.com/cross-validation-and-hyperparameter-tuning-how-to-optimise-your-machine-learning-model-13f005af9d7d. [citováno 13. 4. 2023]

[7] K. Huang et al., „Therapeutics Data Commons: Machine Learning Datasets and Tasks for Drug Discovery and Development“. arXiv, 2021. doi: 10.48550/ARXIV.2102.09548.

[8] Chemoinformatics: Basic Concepts and Methods. Editor Thomas ENGEL, editor Johann GASTEIGER. Weinheim: Wiley-VCH, 2018. ISBN 978-3-527-33109-3.

[9] REDDY, Michael K. *Amino acid*. Online. In: britannica.com 20. 7. 1998. Upraveno 11. 1. 2024. Dostupné z: https://www.britannica.com/science/amino-acid. [citováno 28. 2. 2024]

[10] „Amino acid generic structure.png“ by PJsg1011 podléhá licenci CC BY-SA 4.0

[11] MOITRA, Karobi. The Interview: Hemoglobin vs. Myoglobin.

[12] „Bílkoviny“ by RNDr. Jan Břižďala podléhá licenci CC BY-SA 4.0

[13] KOOLMAN, Jan a RÖHM, Klaus-Heinrich. Barevný atlas biochemie. Praha: Grada, 2012. ISBN isbn:978-80-247-2977-0.

[14] A. Roujeinikova a D. Leys, „Structure of Mycobacterium tuberculosis CYP121 in complex with the antifungal drug fluconazole“. Worldwide Protein Data Bank, lis. 07, 2006. doi: 10.2210/pdb2ij7/pdb.

[15] „Enzymy“ by RNDr. Jan Bříždala podléhá licenci CC BY-SA 4.0

[16] E. F. Pettersen et al., „UCSF Chimera—A visualization system for exploratory research and analysis“, Journal of Computational Chemistry, roč. 25, č. 13. Wiley, s. 1605–1612, čvc. 2004. doi: 10.1002/jcc.20084.

[17] Eberhardt, J., Santos-Martins, D., Tillack, A.F., Forli, S. (2021). AutoDock Vina 1.2.0: New Docking Methods, Expanded Force Field, and Python Bindings. Journal of Chemical Information and Modeling.

[18] Trott, O., & Olson, A. J. (2010). AutoDock Vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. Journal of computational chemistry, 31(2), 455-461.

[19] M. Fonvielle et al., „Substrate and Reaction Specificity of Mycobacterium tuberculosis Cytochrome P450 CYP121“, Journal of Biological Chemistry, roč. 288, č. 24. Elsevier BV, s. 17347–17359, čer. 2013. doi: 10.1074/jbc.m112.443853.

[20] „Michaelis Menten curve.png“ by Thomas Shafee podléhá licenci CC BY-SA 4.0

[21] „Steady states and the Michaelis Menten equation“ (video). Khan Academy. Citováno 28. 2. 2024. https://www.khanacademy.org/test-prep/mcat/biomolecules/enzyme-kinetics/v/steady-states-and-the-michaelis-menten-equation.

[22] M. Zhao et al., „Cytochrome P450 Enzymes and Drug Metabolism in Humans“, International Journal of Molecular Sciences, roč. 22, č. 23. MDPI AG, s. 12808, lis. 26, 2021. doi: 10.3390/ijms222312808.

[23] MATAL, Jaroslav. Porovnání lidských a prasečích biotransformačních enzymů se zřetelem na cytochromy p450 1a2, 2a19 a udp-glukuronosyltransferasu 1a6. [online]. Olomouc, 2009 [citováno 28. 2. 2024]. Dostupné z: https://theses.cz/id/pw6zsq/695517. Disertační práce. Univerzita Palackého v Olomouci, Lékařská fakulta. Prof. RNDr. Pavel Anzenbacher, DrSc.

[24] X. Huang a J. T. Groves, „Beyond ferryl-mediated hydroxylation: 40 years of the rebound mechanism and C–H activation“, JBIC Journal of Biological Inorganic Chemistry, roč. 22, č. 2–3. Springer Science and Business Media LLC, s. 185–207, pro. 01, 2016. doi: 10.1007/s00775-016-1414-3.

[25] VOKURKA, Martin a HUGO, Jan. Velký lékařský slovník. 10. aktualizované vydání. Jessenius. Praha: Maxdorf, [2015]. ISBN isbn:978-80-7345-456-2.

[26] M. Newcomb, J. A. Halgrimson, J. H. Horner, E. C. Wasinger, L. X. Chen, a S. G. Sligar, „X-ray absorption spectroscopic characterization of a cytochrome P450 compound II derivative“, Proceedings of the National Academy of Sciences, roč. 105, č. 24. Proceedings of the National Academy of Sciences, s. 8179–8184, čer. 17, 2008. doi: 10.1073/pnas.0708299105.

[27] C. de Bruyn Kops, M. Šícho, A. Mazzolari, a J. Kirchmair, „GLORYx: Prediction of the Metabolites Resulting from Phase 1 and Phase 2 Biotransformations of Xenobiotics“, Chemical Research in Toxicology, roč. 34, č. 2. American Chemical Society (ACS), s. 286–299, srp. 10, 2020. doi: 10.1021/acs.chemrestox.0c00224.

[28] T. L. Sandritter, M. McLaughlin, M. Artman, a J. Lowry, „The Interplay between Pharmacokinetics and Pharmacodynamics“, Pediatrics In Review, roč. 38, č. 5. American Academy of Pediatrics (AAP), s. 195–206, kvě. 01, 2017. doi: 10.1542/pir.2016-0101.

[29] Haberfeld H, ed. (2020). Austria-Codex (in German). Vienna: Österreichischer Apothekerverlag. Noradrenalin Orpha 1 mg/ml Konzentrat zur Herstellung einer Infusionslösung.

[30] A. T. Deshkar a P. A. Shirure, „Bedaquiline: A Novel Diarylquinoline for Multidrug-Resistant Pulmonary Tuberculosis“, Cureus. Springer Science and Business Media LLC, srp. 29, 2022. doi: 10.7759/cureus.28519.

[31] J. Hilsted, N. J. Christensen, a S. Madsbad, „Whole Body Clearance of Norepinephrine. THE SIGNIFICANCE OF ARTERIAL SAMPLING AND OF SURGICAL STRESS“, Journal of Clinical Investigation, roč. 71, č. 3. American Society for Clinical Investigation, s. 500–505, bře. 01, 1983. doi: 10.1172/jci110794.

[32] S. C. McLeay, P. Vis, R. P. G. van Heeswijk, a B. Green, „Population Pharmacokinetics of Bedaquiline (TMC207), a Novel Antituberculosis Drug“, Antimicrobial Agents and Chemotherapy, roč. 58, č. 9. American Society for Microbiology, s. 5315–5324, zář. 2014. doi: 10.1128/aac.01418-13.

[33] Obach, R. Scott, Franco Lombardo, and Nigel J. Waters. „Trend analysis of a database of intravenous pharmacokinetic parameters in humans for 670 drug compounds.“ Drug Metabolism and Disposition 36.7 (2008): 1385-1405.

[34] A. Hersey, „ChEMBL Deposited Data Set - AZ_dataset“, EMBL-EBI, úno. 2015. doi: 10.6019/chembl3301361.

[35] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A.,

Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 12, 2825–2830.

[36] A. Pessoa, G. Sousa, L. Maués, F. Alvarenga, a D. Santos, „Cost Forecasting of Public Construction Projects Using Multilayer Perceptron Artificial Neural Networks: A Case Study“, Ingeniería e Investigación, roč. 41, č. 3. Universidad Nacional de Colombia, s. e87737, čer. 27, 2021. doi: 10.15446/ing.investig.v41n3.87737.

[37] T. Akiba, S. Sano, T. Yanase, T. Ohta, a M. Koyama, „Optuna: A Next-generation Hyperparameter Optimization Framework“. arXiv, 2019. doi: 10.48550/ARXIV.1907.10902.

[38] S. Watanabe, „Tree-Structured Parzen Estimator: Understanding Its Algorithm Components and Their Roles for Better Empirical Performance“. arXiv, 2023. doi: 10.48550/ARXIV.2304.11127.

[39] ChemAxon. Dostupné online: https://docs.chemaxon.com/display/docs/extended-connectivity-fingerprint-ecfp.mdf

[40] G. M. Ghiandoni a E. Caldeweyher, „Fast calculation of hydrogen-bond strengths and free energy of hydration of small molecules", Scientific Reports, roč. 13, č. 1. Springer Science and Business Media LLC, bře. 13, 2023. doi: 10.1038/s41598-023-30089-x.

[41] Virtanen, P., et al. „SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python“, in Nature Methods, vol. 17, pp. 261–272, 2020.

[42] Greg Landrum et al., rdkit/rdkit: Release_2023.09.5. Zenodo, 2024. doi: 10.5281/ZENODO.591637.

# 11 PŘÍLOHA 1: README Z GITHUB REPOZITÁŘE

## metabolism-hyperparameter-optimization

Datasety z Therapeutics Data Commons využívané pro trénování a testování modelů: Obach et al., AstraZeneca

Jazzy-data-prep.ipynb → Generování molekulárních deskriptorů pomocí Jazzy a spolu s half-life/clearance ukládání do csv souborů do project_resources/jazzy_splits.

mols-visualization.ipynb → 2D i 3D vizualizace některých molekul, pro správné fungování 3D modelů je potřeba stáhnout Jupyterlab_3dmol extention.

TDC-ADME.ipynb → Strojové učení pomocí modelů ze Scikit-learn na ECFP a molekulárních deskriptorech.

ADME-data-analysis.ipynb → Zpracovávání výsledků strojového učení na benchmarcích TDC-ADME, generování grafů.

inference-model-prep.ipynb → Příprava a ukládání modelů pro inference; tyto modely jsou trénovány jak na trénovacích, tak testovacích datech z TDC-ADME.

interactive-inference.ipynb → Interaktivní inference – predikování vlastností libovolné látky z jejího názvu pomocí zvoleného předtrénovaného modelu.

### project_resources

jazzy_splits/ → csv soubory data splitů s molekulárními deskriptory.

### docking_resources

2ij7.pdb → Krystalografická struktura enzymu CYP121 s ligandem. zdroj

2ij7_edited.{pdb, pdbqt} → Pouze jeden řetězec bez ligandů a bez molekul vody.

2ij7_docked.pdb → 2ij7_edited.pdb s dockovaným ligand_docked.pdb.

2ij7_Compound_1.pdb → CYP121 s aktivovanou formou kyslíku na hemové skupině.

2ij7_*.cxs → ChimeraX session, pomocí kterých jsem získával obrázky dockingu.

ligand.{mol, pdb, pdbqt} → 3,6-bis[(4-hydroxyphenyl)methyl]piperazine-2,5-dione; molekula 1 z obrázku 1 z tohoto článku.

ligand_docked.{pdb, pdbqt} → Dockované ligandy vygenerované pomocí AutoDock Vina.