Appendix A

Work plans

First Semester

During the first semester, the focus was primarily on understanding the formal requirements and expectations for the master's thesis. We conducted a review of the available literature and theoretical background related to dermatotoxicity, QSAR modeling, and machine learning techniques. Based on this research, we outlined the goals and possible direction of the thesis and created an initial plan for the upcoming work.

Second Semester

In the second semester, we began actively working on the thesis document. we wrote the initial version of the problem analysis and established the general structure of the thesis. We also experimented with the LLNA dataset and trained a simple Random Forest model. As part of this phase, we predicted the sensitization index which was specific for this LLNA dataset, which helped refine the modeling

direction.

Third Semester

This semester to using a real-world dataset provided by the thesis supervisor. Prior to that, we attempted to build our own dataset, but it was basically a subset of supervisor's bigger dataset so we switched to theirs. Using the new dataset, we implemented three machine learning models (adding SVC and XGBoost) and tested them across three feature subset sizes (10, 20, 30). we also applied three feature selection strategies (correlation, mutual information, SHAP). Results and observations were documented and annotated directly within the thesis draft.

Fourth Semester

The majority of the work was completed during the fourth semester. I significantly expanded the related work section and revised the theoretical background based on feedback from my thesis supervisor. I implemented two additional models—Gradient Boosting and Logistic Regression—to allow for more comprehensive performance comparison. Hyperparameter tuning was improved using RandomizedSearchCV. A more thorough evaluation was conducted using metrics such as AUC, precision, and recall, alongside confusion matrices. I also added a SHAP-based explainability analysis and finalized the deployment pipeline. Finally, the thesis document was completed and all necessary components—including attachments, plots, and references—were compiled and reviewed.

Appendix B

Technical documentation

In this chapter we go over our implementation and point out the important parts. At the beginning we mention the environment, in which we worked. We also provided how we set up and ran the project.

Setup and running the project

The project was running on running on a Google Colab, with most of the required libraries pre-installed. Other necessary installations and imports for running the training process in Jupyter Notebook. Deployment of the Streamlit app was done locally by running *streamlit run model-deploy.py*. Running this command starts the app on *localhost:8501*.

Working environment

We worked with Python (Python 3.9.5) in Jupyter notebook running in Google Colab with the following specifications RAM: 12.67 GB Disk: 107.72 GB. The model deployment was tested on local machine (notebook HP Pavillon with 16GB RAM).

This is the list of all Python libraries used as some where not compatible with each other we had to go for specific version.

Library	**Version**
matplotlib	3.7.1
numpy	1.24.3
pandas	1.5.3
seaborn	0.12.2
scikit-learn	1.5.2
xgboost	1.7.6
shap	0.44.0
imbalanced-learn	0.11.0
pubchempy	1.0.4
rdkit	2023.9.2
xsmiles	0.2.0
streamlit	1.30.0
joblib	1.3.2

Table B.1: Overview of used Python libraries

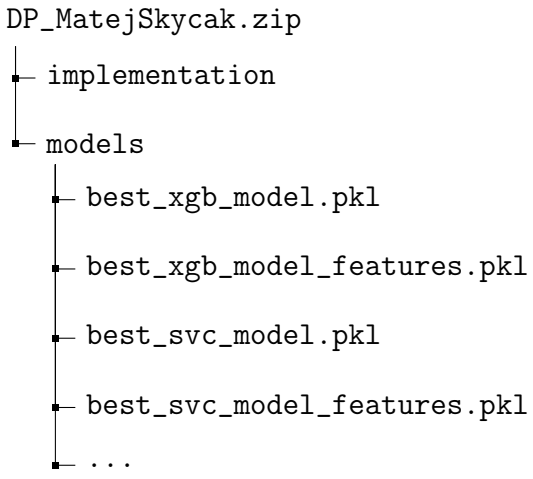
Appendix C

Description of a digital content

In this chapter all digital files, which are appended to this this thesis, are described. We included a tree hierarchy of the content and a short description of what each file contains.

Thesis evidence number: FIIT-182905-111652

Tree file structure for the thesis



implementation - a file containing all code we produced during EDA, pre-processing and training phase. The code for installing and importing required libraries, dataset import, EDA phase, pre-processing, models training, tuning and for evaluation.

models - this folder contains the best fine-tuned models we trained including the *JSON* file with the features selected so it can be reproduced in the deployment. Also, if scaler was used it was also saved.