Machine Learning Report

3rd Home Assignment - Group 8

João Lobato, 62611 (20hr) | Miguel Landum, 35019 (20hr) | Rute Patuleia, 51780 (20hr) | Tiago Assis, 62609 (20hr)

Introduction

The present project aims to find the best regression model for predicting the activity of molecules on dopamine D2 receptors. The target variable for this model is the molecular activity, with values ranging from 0 to 1. Our initial step involved visually inspecting the data, which revealed that the initial dataset comprises 7337 rows and 2132 unknown features, some of numerical and continuous nature; and others of categorical (0 or 1) nature. There were also no null values to be imputed.

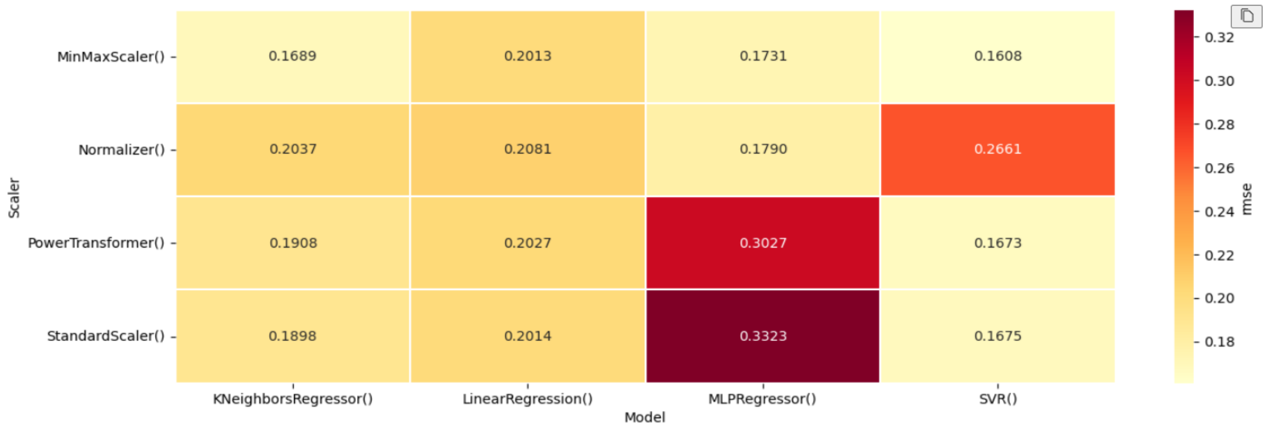
Objective

This work aimed to find the best regression model for an unknown dataset by testing and comparing many feature selection techniques and machine learning models for predicting molecular activity. These models included linear models, tree-based models, k-nearest neighbors (KNN), support vector machines/regressors (SVR), and multi-layer perceptrons (MLP). Various ensemble algorithms were also employed, such as random forests (RF), bagging, gradient boosting, adaptive boosting, and extreme gradient boosting (XGB), as well as voting and stacking methods.

Data Processing

As many regression models based on the distance between data points are sensitive to data scaling, several scaling techniques were tested and evaluated, namely standardization scaling (*StandardScaler*), range scaling (*MinMaxScaler*) between 0 and 1, power transform (*PowerTransformer*) and unit norm scaling (*Normalizer*).

The evaluation process comprised several steps. For each combination of models and scalers, the models were submitted to cross-validation, where the data was scaled in each fold and the performance evaluated through the *rmse* (Figure 1).



Scaler	KNeighborsRegressor()	LinearRegression()	MLPRegressor()	SVR()
MinMaxScaler()	0.1689	0.2013	0.1731	0.1608
Normalizer()	0.2037	0.2081	0.1790	0.2661
PowerTransformer()	0.1908	0.2027	0.3027	0.1673
StandardScaler()	0.1898	0.2014	0.3323	0.1675

Figure 1. Heatmaps representing the rmse for the different models with several combinations of scaling techniques.

The results revealed that, independently of the model, the range scaling with *MinMaxScaler* was the best scaling technique for this dataset. It consistently produced the lowest *rmse* values, providing its efficacy in scaling data when required.

Preliminary Testing

Firstly, several machine learning models were compared at a surface level, with mostly default parameters or slight adjustments, using 11-fold cross-validation (this way, the data is divisible into equal parts and each fold has the same amount of data) to assess which ones could better handle this problem (Table 1).

Table 1. Basic model comparisons and relevant statistics. Hyperparameters not specified are assumed to be default. Results sorted by *rmse*.

Model	**Hyperparameters**	**RVE**	**rmse**	**Pearson's correlation**	**ME**	**MAE**
Support Vector Machine	epsilon=0.01 gamma=0.01	0.6813	0.1562	0.8256	0.8686	0.1133
Extreme Gradient Boosting	learning_rate=0.1 max_depth=10 n_estimators=300	0.6683	0.1594	0.8175	0.9157	0.1160
Support Vector Machine	default	0.6625	0.1608	0.8172	0.8303	0.1236
Random Forest	max_depth=40	0.6491	0.1639	0.8078	0.8338	0.1221
Multi-layer Perceptron	alpha=0.5 hidden_layer_sizes=[50]	0.6467	0.1644	0.8043	0.9106	0.1241
Adaptive Boosting	estimator=SVR(C=0.75, epsilon=0.01) n_estimators=30	0.6468	0.1646	0.8028	0.9164	0.1154
Extreme Gradient Boosting	default	0.6456	0.1647	0.8035	0.8175	0.1228
Adaptive Boosting	estimator=DecisionTreeRegressor(max_depth=40)	0.6430	0.1653	0.8019	0.9247	0.1186
Bagging	estimator=KNeighborsRegressor(n_neighbors=15, weights=gaussian (kernel weight=5))	0.6399	0.1661	0.8028	0.9238	0.1176
Bagging	estimator=SVR(epsilon=0.01, gamma=0.03)	0.6290	0.1685	0.8035	0.7925	0.1284
K-Nearest Neighbours	default	0.6276	0.1689	0.7934	0.9025	0.1222
Bagging	estimator=DecisionTreeRegressor(max_depth=40)	0.6108	0.1726	0.7816	0.9358	0.1280
Bagging	default	0.6097	0.1729	0.7809	0.9109	0.1278
K-Nearest Neighbours	n_neighbors=15 weights=gaussian (kernel weight=5)	0.6065	0.1737	0.7886	0.9656	0.1196
Multi-layer Perceptron	default	0.5917	0.1768	0.7772	0.9417	0.1325
Lasso	alpha=0.0001	0.5771	0.1799	0.7617	0.933	0.1376
Adaptive	estimator=KNeighborsRegressor(	0.5679	0.1820	0.7688	1.0000	0.1230

Boosting	n_neighbors=15, weights=gaussian (kernel weight=5))					
Gradient Boosting	max_depth=40	0.4791	0.1997	0.7182	1.0000	0.1409
Linear Regression	default	0.4703	0.2013	0.7190	1.0976	0.1529
Decision Tree	default	0.3203	0.2281	0.6599	1.0000	0.1574
Decision Tree	max_depth=40	0.3096	0.2299	0.6535	1.0000	0.1587

The best preliminary results were achieved using models such as RF, XGB, MLP, and SVR. Several boosting and bagging models performed very well but were computationally expensive to tune and train, while simpler and/or faster models, such as KNN, ended up performing better after the in-depth parameter tuning performed later.
A first, shallower, model tuning using *GridSearchCV* with 11-fold cross-validation was performed on these models to estimate the optimal range of hyperparameters before proceeding to feature selection. It was revealed that the best models ended up being SVR (0.1562 *rmse*), KNN (0.1611 *rmse*), and XGB (0.1595 *rmse*), while MLP (0.1644 *rmse*) and RF (0.1638 *rmse*) lagged a bit behind. Thus, the former models were chosen as the focus of this work to try and achieve the best performance possible while still maintaining good generalization capabilities. More in-depth grid searches, as well as manual adjustments, were later performed during and after feature selection in an iterative way.

Feature Selection

By focusing on the most informative features, a subset of the original variables was selected to improve model performance and reduce dimensionality. Reducing the number of irrelevant or redundant features can lead to a simpler and more interpretable model that results in an improved generalization, as the model is less likely to overfit to noise in the training data.
Several methods were tested for feature selection: using RF, which can display each feature's importance score, ranking the features based on their contribution to the model's performance, and providing information that can be used for feature selection; and using both Principal Component Analysis (PCA) and Kernel PCA, which can provide a new subset of uncorrelated features by performing linear combinations of the original variables, as the dataset shows very high multicollinearity (*condition number* of 15e+18).
The three methods were compared using the best models as benchmarks, and, although PCA and Kernel PCA solve the problem of multicollinearity, strictly linear models are still not up to par with the models in the study, which resulted in the RF being the best-performing approach for feature selection (Tabela 2).

Table 2. Comparison between the several feature selection methods tested using the *rmse* as the evaluation metric.

	Feature selection method	**RF**	**PCA**	**Kernel PCA**
Model		**rmse**		
SVR		0.15475	0.16291	0.16298
XGB		0.15344	0.16823	0.16789
KNN		0.15826	0.17471	0.17491

In this way, the hyperparameters for the RF model, namely *'max_depth'*, *'min_samples_leaf'*, *'min_samples_split'*, and *'n_estimators'*, were tuned (with the same procedure as before) to best fit the training data. It was found that only the values for *'max_depth'* and *'n_estimators'* had a significant impact on the model's performance, with the *rmse* values stabilizing at 45 and around 200, respectively, while the remaining hyperparameters did not display a concrete pattern. Thus, the model was trained with hyperparameters *max_depth=45* and *n_estimators=225*.

Furthermore, the *threshold* value in *SelectFromModel* for feature selection dictates what variables are kept and discarded according to their relative importance. Features with relative importance higher than the threshold are kept, while others are discarded. Thus, several threshold levels were tested, and tuning was performed according to the minimum *rmse* achieved (Figure 2).

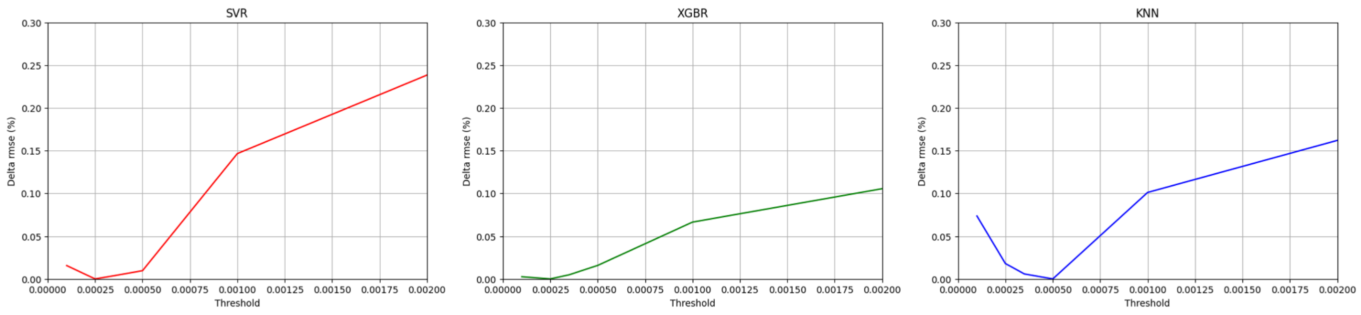


Figure 2. Plots of the percentage decrease in *rmse* based on the threshold value for feature selection using Random Forests.

As observed in the figure above, the rmse value decreases with the threshold (a higher threshold equates to fewer features selected) up to a point, where it then sharply increases again. This occurs because an excessive number of irrelevant rows can add noise to the dataset, reducing the performance of prediction models, while having too few features reduces the amount of information retained by the models, also leading to decreased performance.

Finally, the tuned RF model was used to perform feature selection with *SelectFromModel* using *threshold=0.00025*, which resulted in a dimensionality reduction from 2132 to 621 (71% decrease) variables (Table 3).

Table 3. Comparison between the threshold values and the number of features retained during feature selection.

Threshold	0.0001	0.00025	0.00035	0.0005	0.001	0.005	0.01	"mean"	"median"
Number of features retained	1317	621	454	311	138	30	12	331	1066

Model Tuning and Performance Comparison

The best models found in the beginning were tuned with more in-depth hyperparameter ranges, only taking into account the features selected in the previous step.
For the SVR model, it was observed that values of '*C*' around 1 were optimal, while values of '*epsilon*' and '*gamma*' had to have an order of magnitude of 10^{-2}. For KNN, '*n_neighours'* between 10 and 20 were optimal, while a Gaussian '*weight*' with a kernel value between 4 and 9 performed best. For XGB, a '*learning rate*' of magnitude 10^{-2}; '*colsamply_bytree*' and '*subsample*' between 0.7 and 0.9; *max_depth* of 7; and '*n_estimators*' between 3000 and 3500 were the best ranges for optimization. The final models obtained are presented in more detail in Table 4.

Additionally, two ensemble models were tested and employed, namely *VotingRegressor* and *StackingRegressor*, to combine the predictions of multiple base regressors to improve the overall performance and generalization of the model by averaging the predictions of the base models or by training a meta-estimator on the predictions of the base models, respectively, resulting in the best model being the StackingRegressor combining SVR, KNN, and XGB.

Table 4. Tuned hyperparameters and relevant statistics for the final models.

Model	**Hyperparameters**	**RVE**	**rmse**	**Pearson's correlation**	**ME**	**MAE**
Support Vector Machines	C=0.8 epsilon=0.025 gamma=0.02	0.6901	0.1540	0.8309	0.9324	0.1116
K-Nearest Neighbours	n_neighbours=14 weights=gaussian (kernel_weight=7.5)	0.6783	0.1569	0.8237	0.9217	0.1136
Extreme Gradient Boosting	learning_rate=0.01 colsample_bytree=0.7 subsample=0.8 max-depth=7 n_estimators=3000	0.6924	0.1534	0.8325	0.8519	0.1125
Voting	Combining tuned SVR, KNN, and XGB	0.7077	0.1496	0.8416	0.9020	0.1084
Stacking	Combining tuned SVR, KNN, and XGB	0.7080	0.1495	0.8415	0.9077	0.1077

Conclusion

This study aimed to identify the most effective regression model and feature selection technique to predict molecular activity on dopamine D2 receptors. Several models were tested, and the most promising of those included SVR, XGB, and ensemble algorithms such as bagging and adaptive boosting. After model tuning, it was found that the best models to focus on were SVR, XGB, and KNN, taking into account simplicity and time constraints.

Feature selection using Random Forests was performed, reducing the dataset's dimensionality from 2132 to 621 variables, which significantly improved model performances. Subsequently, a more in-depth hyperparameter tuning was conducted on the best models, only taking into account the selected features, thus optimizing their parameters to enhance their predictive performance.

Finally, a stacking ensemble model was used to combine the tuned SVR, KNN, and XGB models to achieve the best predictions considering each model's strengths. This approach achieved surprising results, with a *rmse* of 0.1495, *RVE* of 0.7080, and a strong Pearson's correlation of 0.8415. This demonstrated to be a strong method for improving predictions in this molecular activity task.

Overall, the present work highlights the importance of thoughtful feature selection and hyperparameter tuning for datasets with a high column-to-row ratio and in developing a robust predictive model for molecular activity on dopamine D2 receptors.