# PREDICTION OF MOLECULAR SOLUBILITY USING MACHINE LEARNING MODELS

*Dinh Long Huynh – 000624-8397*
*Course: Preclinical and Clinical Data Analysis in Predictive Drug Discovery and Development – 3FG289*

## ABSTRACT

The study used a dataset of 2000 compounds including 17 features and experimental solubility to create a predictive model for estimating molecular solubility. The dataset was divided into two sets, a training set and a testing set, with a 7:3 ratio. The training set was then cross-validated to find the best parameters for three different regression models: Multiple Linear Regression (MLR), K-Nearest Neighbors (KNN) Regression, and Random Forest (RF) Regression. The cross-validation was also used to measure the model's uncertainty. Finally, each optimized model was used to predict solubility on the testing set. The RF Regression model performed the best, with an R-squared score of 0.66, outperforming the MLR and KNN Regression models, which scored 0.32 and 0.58, respectively.

***Keywords:*** *solubility, predict, model, MLR, KNN, RF, machine learning*

## INTRODUCTION

Solubility is a crucial factor in the field of medicinal chemistry. It is given utmost importance while considering a drug-like molecule for any drug discovery project. However, during the early stages of drug discovery, there are millions of compounds that need to be taken into account, making it a challenging task, especially when handling compounds lacking solubility data in the database. This also poses a challenge for new candidate compounds that are developed through computer simulations or virtual screening. Therefore, over the course of 100 years, many computational models have been developed to predict molecular solubility before doing the experiments[1]. There are two common approaches for the model, including descriptors-based, the model that based on physiochemical features and fingerprint-based, in other words, the QSPR (quantitative structure – property relationship)[1,2].

This project used the descriptor-based approach, using 17 numerical descriptors to build the regression model. The goal of the project was to find the model that fit best with the training set while having the highest accuracy in generating future predictions.

## METHODS

### 1. Standardized data

The initial data (df_raw) was preprocessed to remove categorical variables, such as ID, Name, InChI, InChIKey, SMILES, Group, and Occurrences. The resulting data frame, named "df_drop," emerged from this elimination process.

Subsequently, the “df_drop” underwent standardization using mean centering and unit standard deviation methods. Mean centering involves subtracting each observation from the mean of the observations of the variable, effectively centering the new means around zero. Following this, unit standard deviation divides each observation by the standard deviation of the observations of the variable. This process ensures that the derived standard deviation becomes equal to 1. The new observation values are determined through these two methods, providing a standardized representation of the data.

$$\frac{x_i - \mu_i}{\sigma_i}$$

$x_i$: observation of variable i
$\mu_i$: mean of observations of variable i
$\sigma_i$: standard deviation of observations of variable i

In our project, we used the StandardScaler() function in Python to handle these processes. The aim of the standardization step is to improve the calculation rate in the machine learning algorithm. This also ensures similarity in each dimension for distance similarity metrics, which are related to the k-nearest neighbor regressor that we plan to use.

### 2. Examine data

The data examination step centered around the primary data frame, involving two key procedures. First, an examination of the distribution of various features, with a specific focus on Lipinski's features, was conducted. This included an analysis of the histogram of the features of interest to investigate the symmetry or skewness of the distribution. Especially, the target value, which is solubility, was additionally applied Q-Q plot to check the normal distribution, because this is the important assumption for linear regression. Simultaneously, the second procedure assessed the Pearson Correlation Coefficient between each pair of features, thus the creation of a correlation matrix for all pairings are carried out. The correlation matrix, encompassing a total of 17 characteristics, enables the identification of pairs of attributes exhibiting positive (positive coefficient) or negative (negative coefficient) correlation. Additionally, it helped gauge the level of their association, where a larger absolute value indicates a stronger correlation.

### 3. Split data

The original data was divided into two subsets: a training set and a testing set, with proportions of 0.7, and 0.3, respectively. The training set was utilized to construct the model, serving as the foundation for model fitting and model cross-validation. Subsequently, the testing set came into play for plotting and comparing between models. The model also underwent testing using the testing set to assess its accuracy for future predictions.

### 4. Mutiple Linear Regression

The project began by using Multiple Linear Regression (MLR) to analyze the predictive relationships between all features in the dataset and the target values. Then, feature selection techniques were applied to improve model interpretability and reduce dimensionality, hoping to improve accuracy, including Correlated-based method and the SelectKBest method.

In the Correlated-based method, features were removed based on their Pearson Correlation Coefficients. If two features had high correlation coefficients, one of them was eliminated. The cut-off threshold for the correlation coefficients was set at various levels, including 0.6, 0.65, 0.7, 0.75, 0.8, 0.85, 0.9, 0.95, 0.975, and 1. If the correlation coefficient between two features was higher than the cut-off threshold, they were assumed that have a high correlation. While in the SelectKBest method, the f-regression algorithm was used to determine the statistical significance of features, retaining only the top k most important features. The range of k was investigated, from 1 to 17.

### 5. K-Nearest Neighbor Regression

The K-Nearest Neighbor (KNN) regression and classification shared the same core concept: identifying the nearest k data points to a new data point using distance metrics such as Euclidean Distance. KNN regression calculates the mean of the target values of the k nearest points and assigns this value to the new data point. We performed the KNN regression with k nearest neighbors fall withitn the range of 2 to 30, to determine the optimal number of k.

### 6. Random Forest Regressor

To achieve the highest accuracy for the final test, it was recommended to use the Random Forest (RF) Regressor. Although it was time-intensive, it was one of the most accurate classifiers and regressors available. This was the reason why we kept it for the last trial after all of the previous approaches could not achieve desirable accuracy.

The Random Forest Regressor is the further development of the Decision Tree algorithm. Each feature is considered from the root to the leaf of the tree, and an appropriate cut-off threshold for each leaf is set to categorize data points effectively. When new data points are introduced, they are put through the Decision Tree and distributed into a group. The average target value of this group is the regressed value for the new data point.

To build the Random Forest, there are two main steps: bootstrapping data and feature selection. Boostrapping data horizontally separates data into many random subsets, the hyperparameter ‘bootstrap’ can receive the logical values, to control this process. Feature selection vertically selects several random features for building the trees, all the remaning hyperparameters were related to the tree buliding step. Then, the new data point is then put into each tree to obtain each predicted value. The final prediction is the mean of predicted values from all trees in the Random Forest.

When using the Random Forest Regressor, it is mandatory to set the n_estimators parameter, which determines the number of trees in the forest. The investigation that how number of trees affect the evaluation scores was conducted, with the number of tree were 50, 100, 150, 200, 300, and 400. Additionally, the bootstrap, max_depth, max_features, min_sample_leaf, and min_sample_split can be adjusted as hyperparameters to refine the model. To ensure fair comparisons between models, the random state of the Random Forest should remain constant throughout the optimization process.

### 7. Model evaluation and Uncertainty evaluation

In addition to determining evaluation scores such as Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and R-squared (R2), our project also examines the uncertainty of each score by using K-fold cross-validation on the training set. Our training set had the medium size, which is 1400-observation large, then the number of folds being equal to 10 was chosen, which means that the training data was divided into 10 subsets. The regression algorithm was then looped 10 times. At $k^{th}$ iteration, the $k^{th}$ subsets was used as a validation set, while all the remaining subsets were used as the training set. Evaluation scores were collected at each iteration, and the mean, standard deviation, and 95% confidence interval were calculated to assess the uncertainty of the models.

We used the GridSearchCV() function in Python to perform a task where the cv was set to 10. It's important to note that the scoring argument was a dictionary consisting of r2, mae, and mse, each key having the corresponding value of make_scorer function. During the GridSearchCV function, all evaluation scores were accounted for, but while finding the best parameters for RF model, only RMSE was taken into consideration. This was done because there were more than one parameter and more than one scoring function, so the graphical work cannot be applied to consider all three evaluation scores at the same time. Plus, RMSE is the most important metric for the task at hand, hence the refit argument was set to 'rmse'. After running GridSearchCV, the mean and standard deviation of evaluation scores can be extracted from the internal

array cv_results_. The refit step was necessary for evaluating feature importance. Similarly, the internal array feature_important can be accessed to perform the task.

### 8. Feature Importance

The optimal RF model was introduced with feature importance determination to evaluate which descriptor was the most important for the predicting task. After running the RF on the testing set, the data for feature importance was extracted and rearranged in descending order. Finally, a bar chart was plotted to show all the importance of descriptors.

## RESULTS – DISCUSSION

### 1. Data examination

This data has 2000 compounds, and the solubility of them was derived in the logarithm form - logS. Distribution of solubility is nearly normal distribution, not so many outliers (**Fig. 1A**). To ensure this point, the Q_Q plot between solubility and the corresponding normal distribution, which had the same mean and standard deviation to the solubility, was made. The plot was nearly linear, with the slight outliers and the beginning and the end. The compound has the logS < 0 is considered as the poorly soluble subtance. In this data set, there are 92.4% compound has the low solubility in water.

Distribution of almost features are right skew, except molecular logP (**Appendix 1**). However, it doesn't violate the normal distribution assumption of MLR. This asumption is mostly about the distribution of target value and the residuals.

The correlation matrix showed that solubility only has the medium correlation with logP (-0.53), while all the remaining features has low or no correlation with

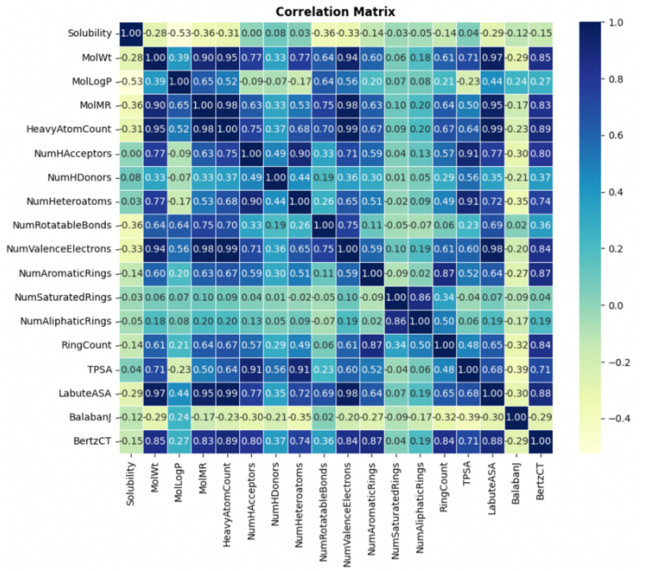



**Figure 2.** Correlation Matrix

solubility, which predictively indicate for the low linear relationship between the target value and feature values (**Fig. 2**). Considering the correlation between features, there are 6 features that have high correlation with others (greater than 0.9), including MolMR, HeavyAtomCount, NumHeteroAtoms, NumberValanceElectrons, TPSA, and LabuteTPSA. Another point that worth mentioning is that the definition threshold of Pearson Coefficient. Further investigation with various cut-off thresholds (from 0.6 to 1) would be investigated in the MLR part.

### 2. Multiple Linear Regression

The MLR performed poorly with a low R2 score of 0.32 and high RMSE and MAE scores of 0.82 and 0.62 respectively (**Table 1**). To improve the model, we implemented a feature selection method since there were many highly correlated features.

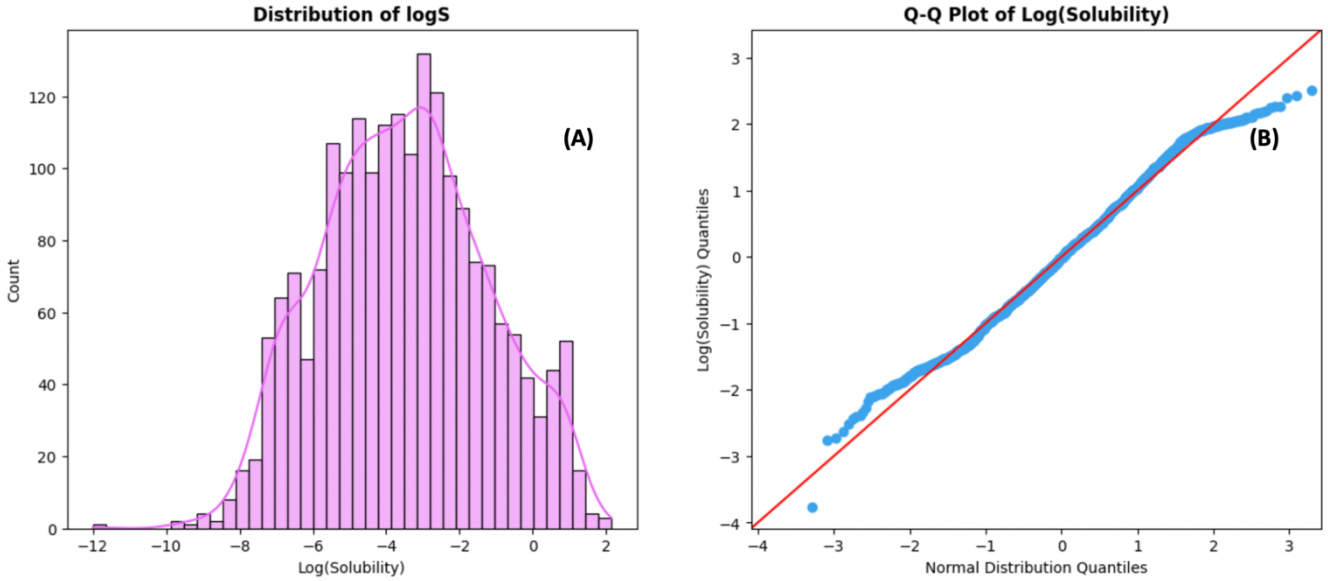



**Figure 1. (A)** Distribution of Solubility (log scale)
**(B)** Q_Q plot of Solubility and the corresponding Normal distribution

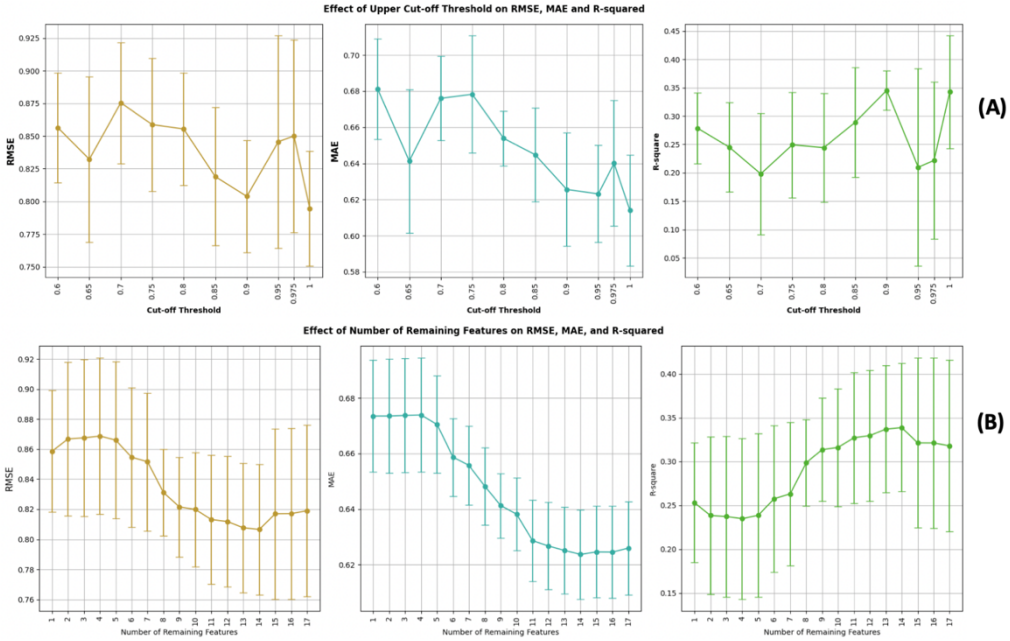



**Figure 3. (A)** The effect of Cut-off threshold on evaluation scores, based on Correlation-Based Features Selection method. **(B)** The effect of Number of Remaining Features on evaluation scores, based on Select K-best Features Selection method.

Following that idea, two types of feature selection methods was investigated. The first method was the correlation-based feature selection, which showed there is no trend in both means and standard deviation of evaluation scores. The highest accuracy can be obtain with the cut-off threshold equal to 1 leading to R2 at 0.34 (**Fig. 3A**), which mean remain all the features. The second method was the Select K-Best features, which produced a similar result with the highest R2 score of 0.34 when 14 most important feautures was select (**Fig. 3B**). However, if we take into account the 95% confidence interval, there is no significant differents in all evaluation scores between remaining 14 most significant features and remaining all the features, which has the R2 at 0.32, with the 95% CI range of 0.09 (**Table 1**). Unfortunately, these scores were still unsatisfactory for future prediction.

After analyzing the results, we concluded that removing features from the MLR model could not improve the model's precision. In the previous papers, the feature selection commonly works when the number of features is fairly high, more than hundreds for example. By way of illustration, in the report of Arash et. al., his team also used the correlated-based feature selection, which resulted in filtering out 177 final descriptors over 811 initial ones[2]. In our case, however, the total number of original features was only 17, leading to a fluctuated trend in evaluation scores when conducting a feature selection.

Therefore, we concluded that MLR may not be a suitable model for the dataset. This was consistent with the quite low correlation between solubility and most of the features. We also decided not to explore other feature extraction methods such as Partial Least Squared Regression or Principle Components Analysis since they may lower the accuracy of the regression methods. Instead, we considered models that could handle the non-linear relationship between solubility and descriptors, which turned our attention to the non-parametric models.

## 3. K-Nearest Neighbors Regression

The KNN regressor calculates the mean of the k-nearest neighbors to predict the outcome of new data

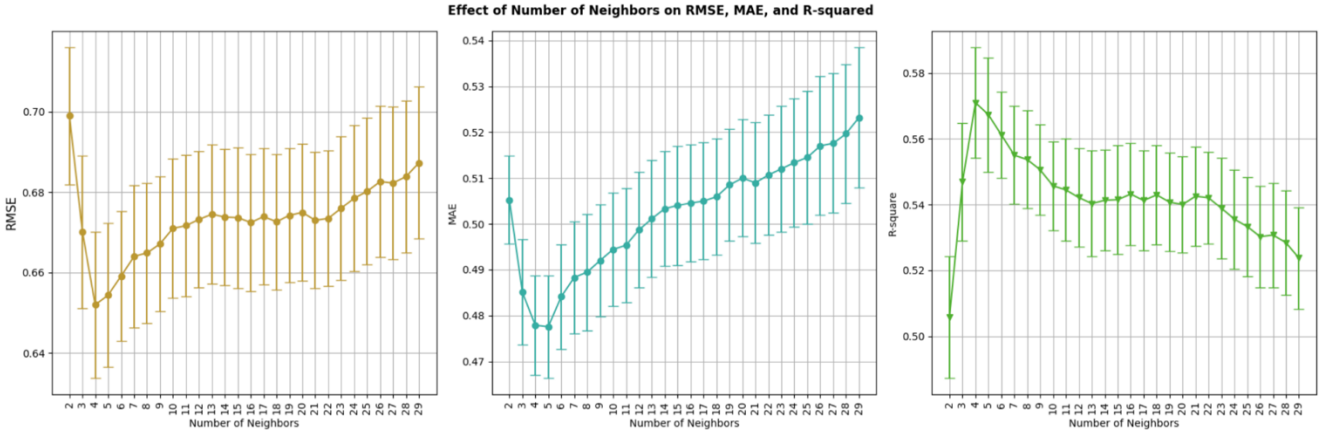



**Figure 4.** The effect of Number of Neighbors on evaluation scores, based on KNN regression

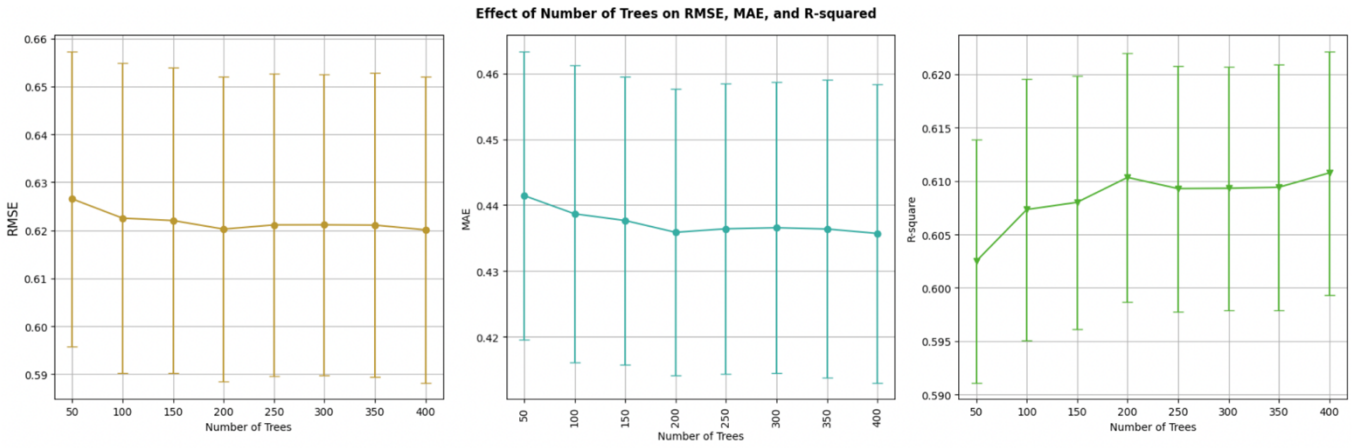



**Figure 5.** The effect of Number of Trees on evaluation scores, based on Random Forest regression

points. As it can handle non-linear relationships, it yields a significantly higher R2 score compared to MLR. The RMSE and MAE scores are consequently lower. The optimal R2, RMSE, and MAE scores achievable were 0.57, 0.65, and 0.48, respectively. The optimal number of neighbors in this model was found using the GridSearchCV function in Python, which turned out to be 4 nearest neighbors (**Fig. 4**).

## 4. Random Forest Regression

RF Regressor method yielded the highest R2 scores among the three methods used, delivering an R2 score of 0.61 for the optimal number of trees. Using GridSearchCV to determine the optimal estimators, the number of trees was established at 200. **Figure 5** also showed that the number of trees between 50 to 400 had a modest effect on evaluation scores. In this case, the choice of the number of trees depends on the size of the dataset. For our project's 2000 compounds, we selected the optimal option with 200 trees. However, for ultra-large datasets, such as those with 50 or 100 trees may be selected to save calculation time.

In the Random Forest algorithm, a randomized feature selection step was already in place, so feature selection or extraction strategies was not applied to Random Forest. Instead, we apply the hyperparameter tunning for Random Forest, aiming to obtain the highest possible precision. The parameters that be taken into account were n_estimators, bootstrap, max_depth, max_features, , min_sample_leaf, and min_sample_split. The best estimators were then determined, which are 184, False, 13, 'log2', 5, and 4, respectively (**Table 1**).

## 5. Comparison of models

The training step was conducted on training set, aiming to find the optimal parameters of each algorithms. It was followed by the testing steps, which was responsible for comparing between optimal models. We applied the found optimal conditions for MLR, KNN, and RF to the testing set to see how well the model can predict the solubility.

**Table 1.** Summary table of MLR, KNN and Random Forest model evaluation for solubility predictions

| Method | | Best parameter | Training set | | | Testing set | | |
|---|---|---|---|---|---|---|---|---|
| | | | R2 | MAE | RMSE | R2 | MAE | RMSE |
| MLR | All features | - | 0.32 ± 0.12 | 0.62 ± 0.02 | 0.82 ± 0.01 | 0.32 | 0.65 | 0.83 |
| | Correlation-based Features Selection | Cut-off threshold = 1 | 0.34 ± 0.03 | 0.61 ± 0.03 | 0.79 ± 0.04 | - | - | - |
| | SelectKBest Features Selection | n_features = 14 | 0.34 ± 0.07 | 0.62 ± 0.02 | 0.80 ± 0.04 | - | - | - |
| | | n_features = 17 (all features) | 0.32 ± 0.09 | 0.62 ± 0.02 | 0.82 ± 0.06 | - | - | - |
| KNN Regressor | | n_neighbors = 4 | 0.57 ± 0.02 | 0.48 ± 0.01 | 0.65 ± 0.02 | 0.58 | 0.47 | 0.65 |
| Random Forest | | (1) n_estimators = 200<br>(2) n_estimators = 184<br>bootstrap = False<br>max_depth = 13<br>max_features = 'log2'<br>min_sample_leaf = 5<br>min_sample_split = 4 | 0.61 ± 0.01 | 0.44 ± 0.02 | 0.62 ± 0.03 | 0.66 | 0.43 | 0.59 |

*(1): Apply to training set*
*(2): Apply to testing set*

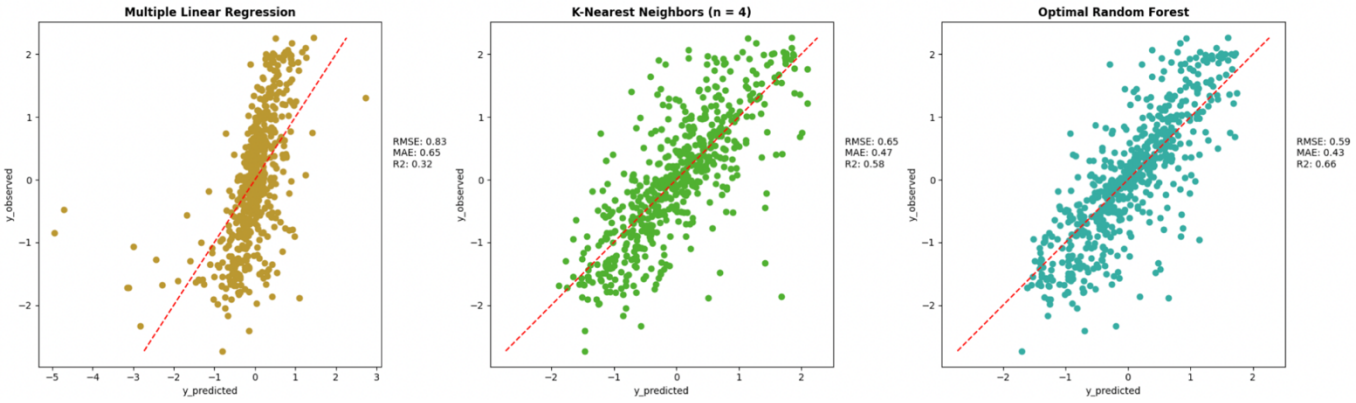



**Figure 6.** Scatter plot of prediction and observation

The results were consistent compared to the training set, with the R2 of MLR, KNN, and RF were 0.32, 0.58, and 0.66, respectively (**Fig. 6**). The possible advantages of MLR might be the interpretability and calculation time-saving. However, the MLR's prediction accuracy was significantly low, compare to the KNN and RF regressors. Another evidence support for the fact that MLR was not an appropriate model is the low correlation between solubility with most of the descriptors in the dataset.

Away from the parametric model, KNN and RF performed remarkably better, with the R2 scores being 1.8 and 2.1 times higher than MLR, respectively. Among them, RF was better than KNN based on all evaluation scores. It suggested that the RF was the best model among the three, despite the high running time. RF required a large time to tune the hyperparameters, however, when they have been already in hand, RF can easily run for the 2000-compound dataset.

## 6. Feature Importance

After creating the best RF model, we applied feature importance determination to it. The analysis showed that molecular logP was the most significant factor in predicting solubility (**Fig. 7**), which represents for the distribution of neutral molecule between nonpolar and polar solvent. This finding is consistent with the fact that logP has the strongest correlation with solubility (0.53, **Fig. 2**), because higher logP means lower concentration in water compared to octanol, indicating higher lipophilicity, less interaction with water (polar solvent), and hence lower aqueous solubility.

The second most important feature was molar refractivity, which indicates a molecule's polarity[3]. It has a strong correlation with solvent accesible surface (LabuteASA), molecular weight, and number of valence electron (**Fig. 2**). Therefore, all of them were the next. important factors that affect solubility (**Fig. 7**). This indicates that polarity is a crucial factor in predicting solubility. Indeed, if the molecule has a high level of polarity, the interaction between molecules in the crystal lattice will be strong, leading to the solid-state limit of solubility. Conversely, if the molecules have low polarity,

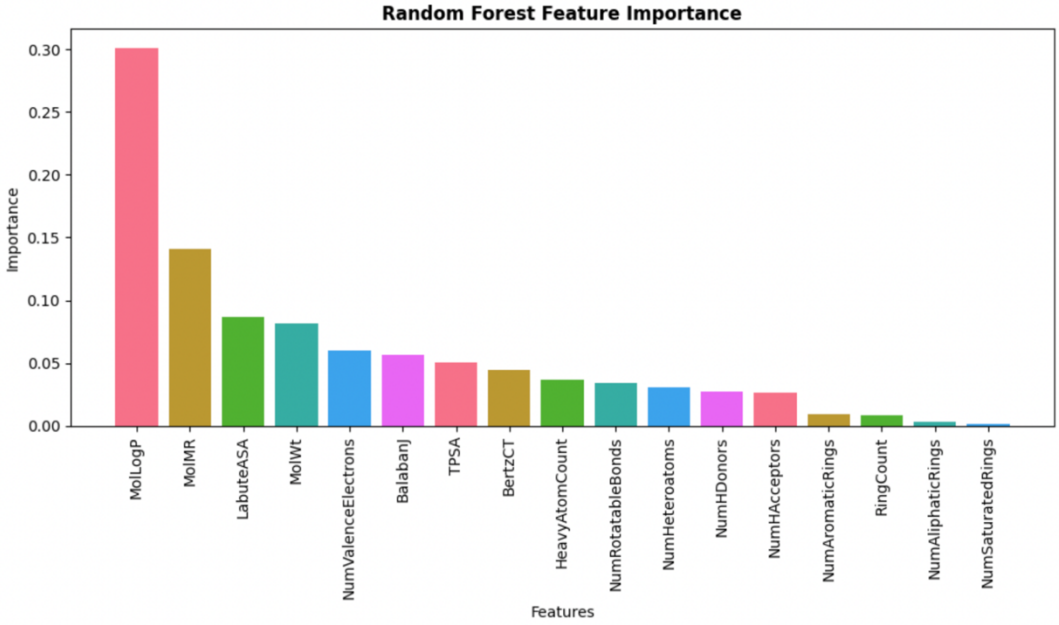



**Figure 7.** Features' Importance in Random Forest Model

they will have difficulty interacting with the polar solvent, such as water, leading to the solvation limit of solubility.

## CONCLUSION

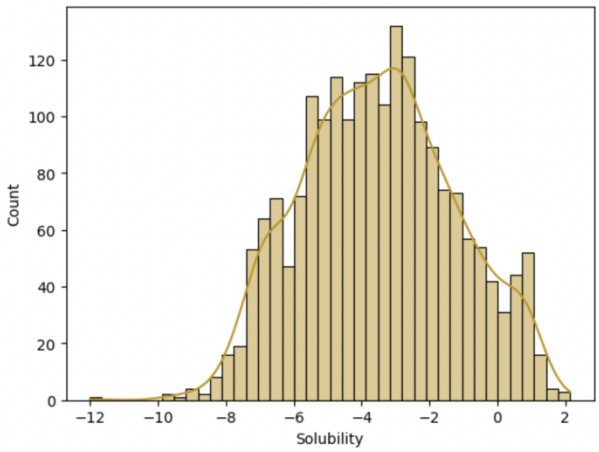
In the course of this project, an in-depth exploration was conducted on three regression models - MLR, KNN, and RF - for the purpose of predicting molecular solubility. The utilization of 17 physiochemical features, termed descriptor-based regression, formed the basis for these models. Despite the application of two feature selection strategies, no significant enhancement in model performance was observed. Following exhaustive experimentation, the RF model, characterized by 184 estimators, no bootstrap, a max depth of 13, 'log2' for max features, a minimum sample leaf of 5, and a minimum sample split of 4, emerged as the optimal choice. The performance metrics for this model included RMSE of 0.59, MAE of 0.43, and R2 of 0.66.

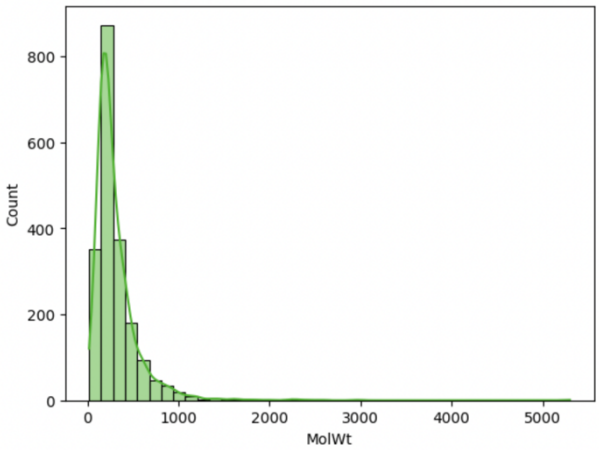
The limitation posed by a restricted number of features emerged as a notable challenge in the context of descriptor-based regression, contributing to a relatively low R-squared, even with the optimal RF model. To address this, two potential approaches for further investigation were identified to enhance prediction performance: the exploration of more influential features within the dataset and the consideration of fingerprint-based regression as an alternative approach.

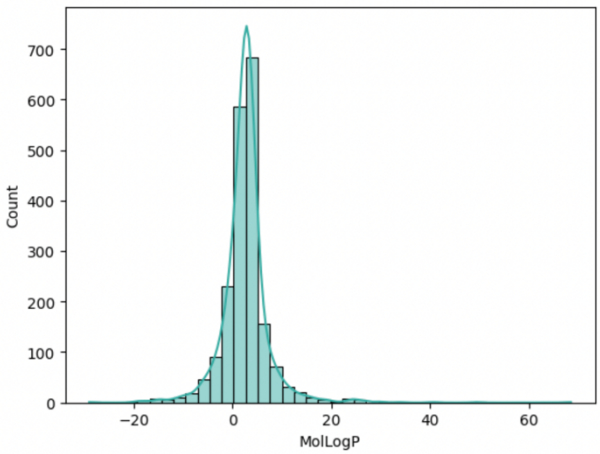
## REFERENCE

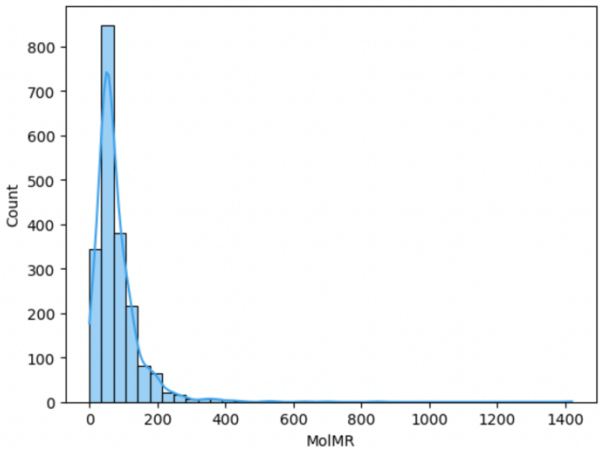
1. Ye, Z. & Ouyang, D. Prediction of small-molecule compound solubility in organic solvents by machine learning algorithms. *J. Cheminformatics* **13**, 98 (2021).
2. Tayyebi, A. *et al.* Prediction of organic compound aqueous solubility using machine learning: a comparison study of descriptor-based and fingerprints-based models. *J. Cheminformatics* **15**, 99 (2023).
3. Le Fèvre, R. J. W. Molecular Refractivity and Polarizability. in *Advances in Physical Organic Chemistry* (ed. Gold, V.) vol. 3 1–90 (Academic Press, 1965).

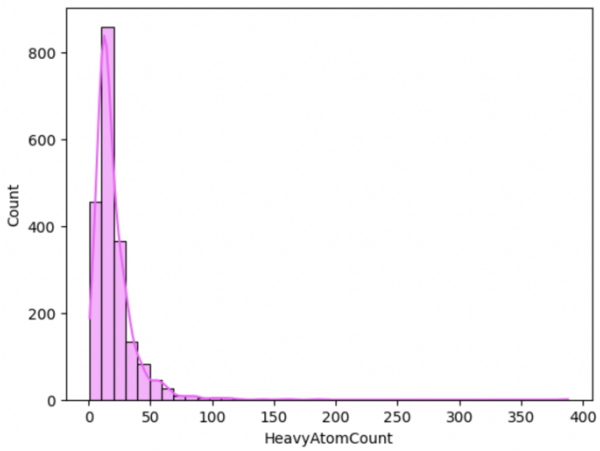
# **Appendix 1.** Distribution of all features

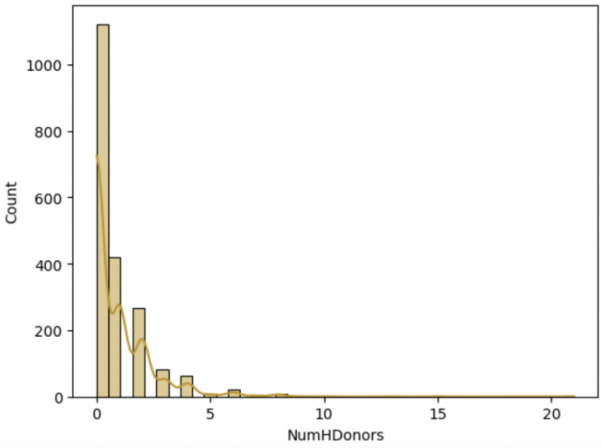


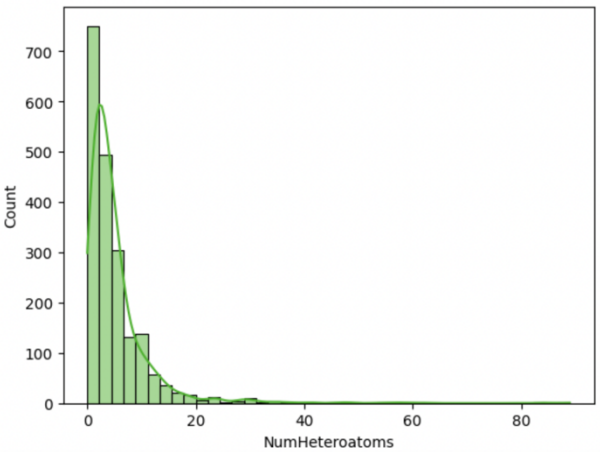



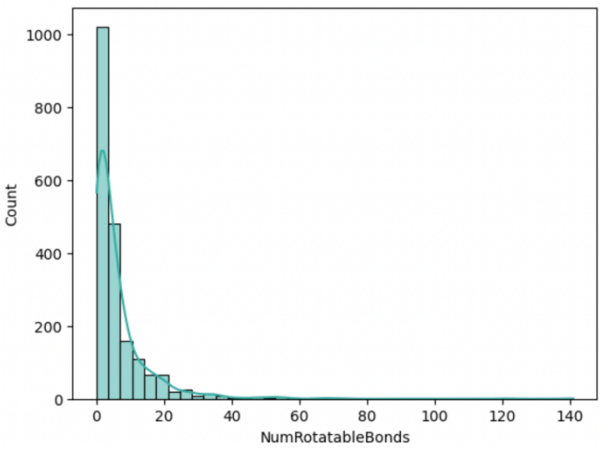



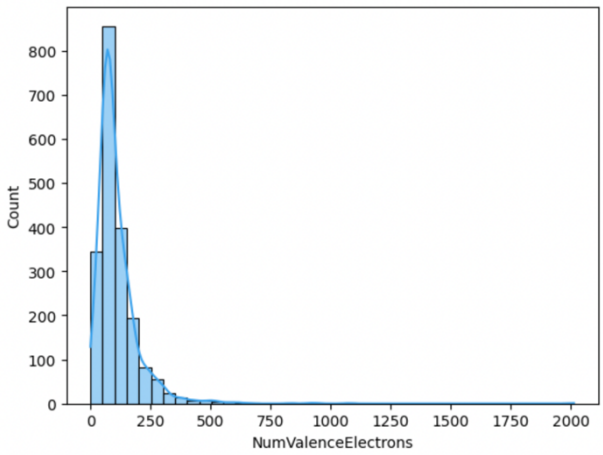



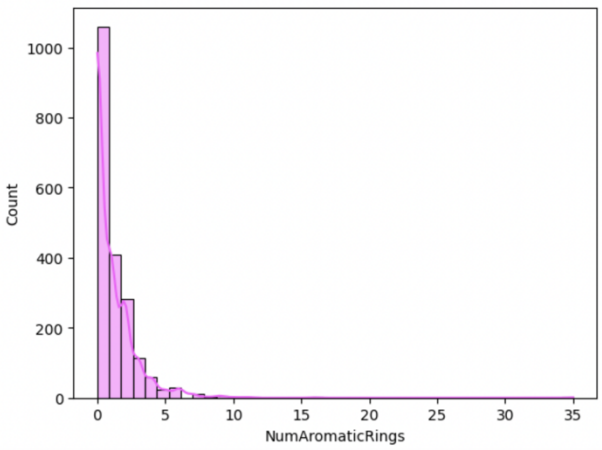



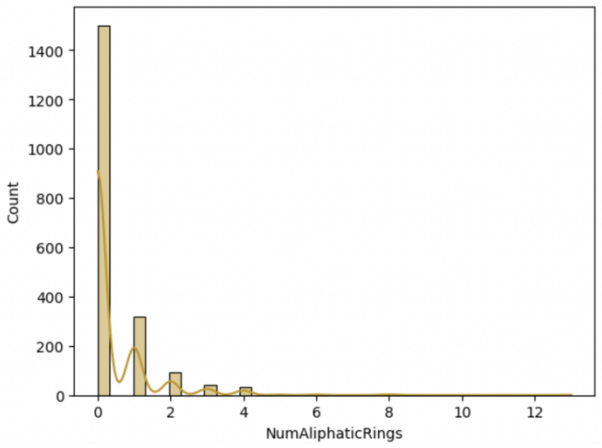



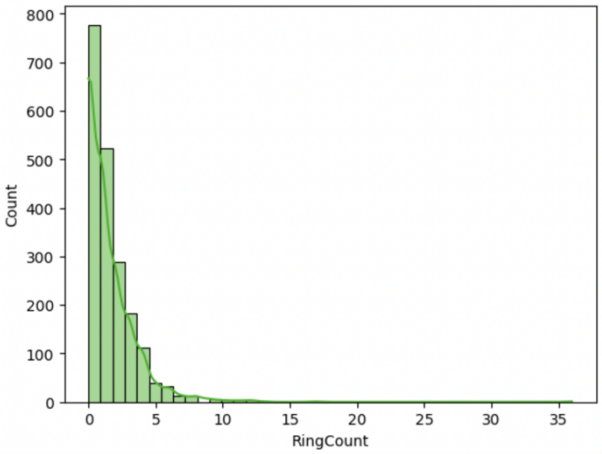



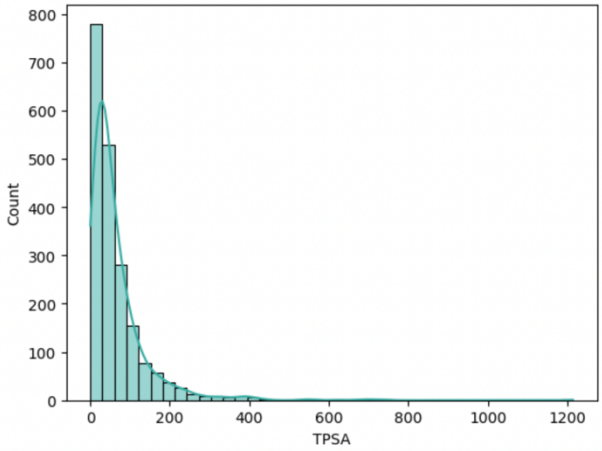



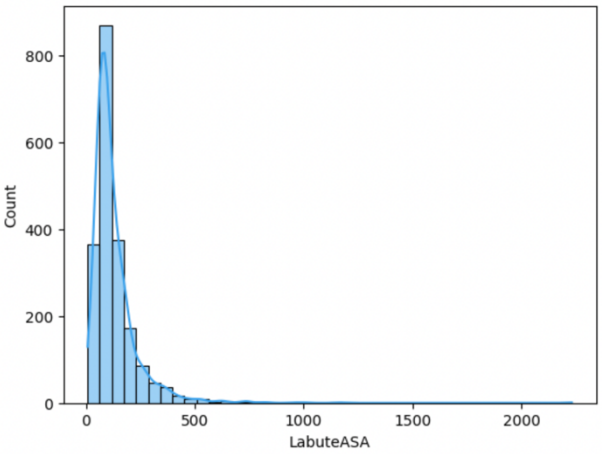

Count
NumValenceElectrons

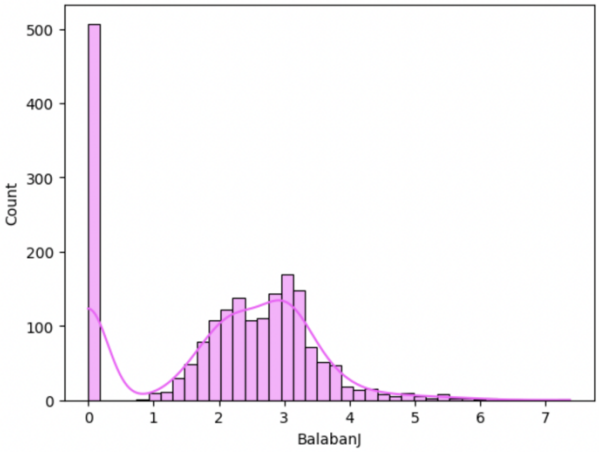

Count
NumAromaticRings


Count
NumAliphaticRings


Count
RingCount


Count
TPSA


Count
LabuteASA


Count
BalabanJ