**Write your abstract here. It should be no longer than one page.**

**Remember to submit a separate copy of the abstract, headed by the**

**dissertation title and your name, with a copy of the academic integrity form.**

**INTRODUCTION**

**What is Solubility?**

The ability of a chemical compound known as a solute to dissolve in a solid, liquid, or gaseous solvent and create a homogenous solution of the solute in the solvent is known as solubility. Fundamentally, a substance's solubility is influenced by the solvent being employed, together with temperature and pressure. When more solute is added, its concentration in the solution does not rise, this is known as the saturation concentration, and it indicates how much of a substance is soluble in a certain solvent.

The degree of solubility varies greatly, ranging from endlessly soluble (totally miscible) substances like ethanol in water to substances that are poorly soluble, such silver chloride in water. Poorly or very poorly soluble substances are frequently referred to as being "insoluble."

The simultaneous and opposing processes of dissolution and phase joining produce solubility, which happens when a system is in dynamic equilibrium (e.g., precipitation of solids). When the two processes move forward at the same rate, soluble equilibrium happens. A so-called supersaturated solution, which is metastable, can result when equilibrium solubility is exceeded under certain circumstances.

Typically, the solvent is a liquid, which may be a single material or a combination of two liquids. Speaking of a solution in a gas is uncommon; solid solutions are more common.

**Solubility: ADMET Property**

Solubility is an important ADMET property for drug development. Drug discovery and the evaluation of environmental hazard depend greatly on the qualities of drug candidates, pesticides, and industrial chemicals known as ADMET (absorption, distribution, metabolism, excretion, and toxicity).

**Absorption:**

The method by which a medicine enters the bloodstream is called absorption. The two most popular methods of delivery are intravenous and oral, while there are numerous more options. The absorption phase of a medicine is bypassed if it is given intravenously since the drug enters circulation right away. However, because it allows patients to self-administer, many medications are dosed orally. Molecular weight, topological polar surface area (TPSA), **solubility**, ionisation, and other physicochemical characteristics of a medicine can all affect how well it is absorbed.

**Distribution**:

The reversible movement of a substance from one part of the body to another is referred to as distribution. A medicine must be transported into interstitial and intracellular fluids once it has been absorbed or administered directly into the systemic circulatory system. The medicine may be administered in various amounts to various organs or tissues, and it may stay in those organs or tissues for various lengths of time.

**Metabolism:**

The process of metabolism involves turning xenobiotic substances, which are typically more lipophilic, into hydrophilic metabolites that can be excreted from the body. Enzymes are involved in the metabolism of drugs, and it may take a number of research investigations to pinpoint the main metabolites and pertinent metabolic pathways.

The term "metabolism" can also refer to the totality of chemical processes that take place within living things, such as digestion and the movement of substances inside and outside of cells. In this case, the set of processes listed above that take place inside cells are referred to as intermediary (or intermediate) metabolism.

The three major purposes of metabolism are the transformation of food's energy into cellular energy, the synthesis of proteins, lipids, nucleic acids, and certain carbohydrates, and the removal of metabolic wastes.

**Excretion:**

Excretion is the process through which an organism gets rid of metabolic waste. The permanent removal of a material from the body is known as excretion.

The process of removing a drug provided to a person from the body, known as drug excretion, is the last phase in the ADME (Absorption, Distribution, Metabolism, and Excretion) cycle. Drugs excreted by metabolic biotransformation can also be removed in their original, unmetabolized form. Drugs are prepared for excretion by metabolic biotransformation. More hydrophobic medications typically change into a more polar, water-soluble molecule that is quickly excreted.

Most of the time, the parent drug and all of its metabolites are finally eliminated from the body. Drugs are often eliminated by the kidney's or liver's production of urine or bile and faeces, although they can also be expelled through perspiration, tears, or breathing.

The process of finding a new medicine nowadays involves a number of difficult and expensive steps, including illness selection, target identification, lead or hit discovery and optimization, as well as preclinical and clinical trials. About 50% of prospective medicinal substances failed in clinical trials or were pulled off the market prior to ten years ago due to intolerable side effects and subpar ADMET characteristics. In the early stages of drug discovery, filtering and optimization of ADMET characteristics are thoroughly examined. **Figure xyz** shows Illustrates pipeline of data curation, data organization, predictive model building, and the application fields of admetSAR(structure−activity relationship database)

Diagram

Description automatically generated

**Fig xyz: AMDET properties & example of the admetSAR application fields, predictive model construction pipeline, and data organisation and curation pipeline.**

**Importance of Solubility:**

Due to its simplicity, high patient compliance, cost effectiveness, lack of sterility restrictions, and flexibility in dosage form design, oral consumption is the most practical and frequently used method of drug delivery. Because of this, many generic medication manufacturers are more likely to produce bioequivalent oral drug formulations.

Other dosage forms, such parenteral formulations, also heavily depend on solubility. When taken orally, poorly water-soluble medications may need high dosages to attain therapeutic plasma concentrations. The main issue in developing formulations for new chemical entities as well as generics is low water solubility. Any medicine that is to be absorbed must be present at the absorption site in the form of an aqueous solution. The preferred solvent for liquid medicinal compositions is water. Most medications have poor aqueous solubility and are either weakly basic or mildly acidic.

Over 40% of the NCEs (new chemical entities) created by the pharmaceutical sector are essentially water insoluble. One of the most difficult areas of drug development, particularly for oral-drug delivery systems, continues to be the enhancement of drug solubility and, consequently, its oral bio-availability.

**Solubility Prediction: Not an easy task**

Predicting solubility has various benefits. Drugs used orally must have a high enough water solubility to be bioavailable. Putative drug molecules that are water insoluble cannot be examined in biological assays, have poor pharmacological profiles, and may come out of solution under storage conditions. Under physiological settings, poor solubility and related poor pharmacokinetics can frequently lead to costly late-stage failure in drug discovery. It is believed that issues with the pharmacokinetics of lead compounds cause up to 40% of drug discovery initiatives to be discontinued. But because of several factors, estimating solubility is not an easy task. First of all, the lack of a high-quality data set of trustworthy and reproducible solubility measurements is one reason why solubility is a challenging attribute to predict. Measurement methods, solvent and solute purity, and poor experimental solubility data quality are all factors.

**Solubility Data Extraction:**

The aqueous solubility dataset for the Open Notebook Science Challenge was used to obtain data on solubility in water and ethanol. Data on water molecules' solubilities, melting points, SMILES codes, and StdInChIKey codes (unique identifiers) were gathered. For solubility, units were changed to LogS = log10C (C = concentration in mol/L). An existing database called ONS Challenge offered information on solubility and SMILES codes. If molecules included charges or radicals, they were likewise eliminated. Additionally, missing unit data points were eliminated. A median value was calculated for molecules that had several data points.

For those who had multiple data points and LogS > 1, a determination was made regarding whether any of the data points (e.g., LogS > 2) were incorrectly recorded. The molecule was eliminated if there was no way to clearly remove the incorrect value. The Water set wide analysis revealed a LogS range of -12 to 2.

Molecular weights were determined using the Chemical Identity Resolver Python interface, CIRpy, from the matching SMILES code. The molecular weight was limited to ≤ 504.  This was done in accordance with Lipinski's rule, which states that tiny organic molecules make up the majority of pharmaceutically active substances. Additionally, this keeps computation durations within reason.

Chart, histogram

Description automatically generated

**Fig xyz**

The initial models were constructed using 22 descriptors.

|  |  |  |
| --- | --- | --- |
| No. | Name | Description |
| 1 | E0\_gas | Zero-point energy of optimised gas structure (Hartrees) |
| 2 | E0\_solv | Zero-point energy of optimised solution structure (Hartrees) |
| 3 | DeltaE0\_sol | Solvation energy calculated as E0\_solv - E0\_gas (Hartrees) |
| 4 | G\_gas | Gibbs free energy of optimised gas structure (Hartrees) |
| 5 | G\_solv | Gibbs free energy of optimised solution structure (Hartrees) |
| 6 | DeltaG\_sol | Solvation energy calculated as G\_solv - G\_gas (Hartrees) |
| 7 | HOMO | Energy of the HOMO of gas phase structure of the solute |
| 8 | LUMO | Energy of the LUMO of gas phase structure the solute |
| 9 | LsoluHsolv | The gap in energy between the LUMO of the solute and the HOMO of the solvent |
| 10 | LsolvHsolu | The gap in energy between LUMO of the solvent and HOMO of the solute |
| 11 | Gas\_dip | Dipole of gas structure |
| 12 | Solv\_dip | Dipole of solution structure |
| 13 | O\_charges | Sum of charges on solution structure oxygen atoms |
| 14 | C\_charges | Sum of charges on solution structure carbon atoms |
| 15 | Most\_neg | Charge on most negative atom of solution structure |
| 16 | Most\_pos | Charge on most positive atom of solution structure |
| 17 | Het\_charges | Sum of charges on solution structure non-hydrogen/carbon atoms |
| 18 | Volume | Molar volume (cm-3.mol) |

**Figure xyz** depicts the general process for calculating descriptors. The procedure is discussed in detail below.

Diagram

Description automatically generated

**Fig XYZ**: Flowchart for calculation of descriptors

Charged surface descriptors was generated via standard clustering techniques. The descriptor space was visualised using Principal Component Analysis (PCA). A technique for lowering dimensions is PCA. It also demonstrates how many descriptors are necessary to explain a specific proportion of the variation in the dataset, i.e., whether the descriptor collection contains a large number of associated or redundant descriptors. The PCA plots, especially in the Water set side dataset, demonstrate a link between the descriptors and water. The scree plots demonstrate that the majority of the variance in the dataset has been described after 14 components. In our dataset, descriptor melting point has not been used which gives us 13 components. **(what to write more about this? Why was it not given?)**

Chart, bar chart, histogram

Description automatically generated

**Fig xyz : Using principal component analysis, the datasets' 50–60% of the variation was described in the descriptor space.**

The aqueous solubility dataset for the Open Notebook Science Challenge was used to obtain data on solubility in water. The table **XYZ** shows 13 descriptors along with the predictor variable(LogS) used in this work. The variable to predict is the concentration. Units were converted to LogS = log10C (C = concentration in mol/L) for solubility. The saturation concentration—the point at which adding more solute has no effect on the solution's concentration—is used to gauge how soluble a material is in a certain solvent.

|  |  |
| --- | --- |
| Descriptor variable name | Description |
| MW | Molecular weight |
| Volume | Molar volume (cm-3.mol) |
| SASA | Solvent Accessible Surface Area |
| Sol\_dip | Dipole of solution structure |
| Most\_pos | Charge on most positive atom of solution structure |
| Most\_neg | Charge on most negative atom of solution structure |
| C\_charges | Sum of charges on solution structure carbon atoms |
| DeltaG\_sol | Solvation energy calculated as G\_solv - G\_gas |
| O\_charges | Sum of charges on solution structure oxygen atoms |
| G\_sol | Gibbs free energy of optimised solution structure |
| Het\_charges | Sum of charges on solution structure non-hydrogen/carbon atoms |
| Lsolu\_Hsolv | The gap in energy between the LUMO of the solute and the HOMO of the solvent (eV) |
| Lsolv\_Hsolu | The gap in energy between LUMO of the solvent and HOMO of the solute (eV) |
| LogS | log10C (C = concentration in mol/L) |

Here, we present an effective method for predicting solubility in organic solvents and water by combining machine learning models and statistical analysis.

**METHODS**

**Statistical Analysis:**

Dataset has 13 descriptors and 1 target variable (LogS). The variables have been statistically analyses for further research. There are no missing data in the dataset.

Chart, histogram

Description automatically generated

The **figure xyz** shows the histogram of LogS shows that the distribution is negatively skewed as most of the data points are on the right side. The isolated bars towards the far left end shows that we have outliers in the target variable. The value of LogS ranges from -12.790 to 1.580 with a mean of -3.006 and median of -2.949. We have most of the data points between -5 and 0.

Chart, scatter chart

Description automatically generated

Fig: Plot showing non-linearity between Most\_neg and Solubility

There are loads of non-linearity in the data. Most of the variables shows non-linearity with the predictor variable (LogS). But we also have some variables like G\_sol, volume, MW and SASA which shows linearity with the predictor variable. The figure **XYZ** shows how data points are clustered and doesn’t show any linearity with the predictor variable. On the other hand, figure **XYZ** shows that we also have some variables which does show some linearity with the predictor variable.

Chart, scatter chart

Description automatically generated

Fig: Plot showing some linearity between volume and Solubility

Further analysis also shows multicollinearity in the dataset. Figure **XYZ** shows Gsol, Volume, SASA and MW are strongly correlated. Similarlity, Most\_pos, Most\_neg & O\_charges are strong correlated. It also makes some sense by understanding the description of the variables like Most\_pos & Most\_neg. Presence of correlated predictors impacts the ability to identify strong predictors. Therefore, a variety of linear and non-linear machine learning models have been employed to assess the efficacy and comprehend the outcomes. Given that each model has a distinct edge over the others.

Chart, bubble chart

Description automatically generated

Figure **XYZ**: Correlation plot between all the features

To solve the problem of noisy data, multicollinearity and many variables showing non-linearity while some showing linearity in the data, two steps have been taken:

1. **Introducing an extra metric for performance evaluation:**

Pearson’s R2 and root mean square error (RMSE) are frequently used metric to assess how accurately predicted value correspond to actual value.

\begin{equation\*}

R^{2} =

\left(\frac{ \sum\_{i=1}^{n}(x\_i-\bar{x})(y\_i-\bar{y}) }{%

\sqrt{\sum\_{i=1}^{n}(x\_i-\bar{x})^2}\sqrt{\sum\_{i=1}^{n}(y\_i-\bar{y})^2}}\right)^{2}

\end{equation\*}

Text, letter

Description automatically generated

where x̅ and y̅ are the mean values of x and y. xpred and ypred are the predicted values of x and y. n is the number of instances and ŷi is the predicted value of yi, which is the correct value.

In the case of solubility prediction, as experimental error is around LogS±0.5-0.7, % LogS within ± 0.7 (%LogS±0.7) with experimental value is a useful measure of molecules within the experimental error. The % LogS within ± 1.0 (%LogS±1.0) of experimental is a more arbitrary decision but does allow easy comprehension of molecules within an order of magnitude of the experimental value.

1. **Wisdom of crowd:**

‘In 1906, the great statistician Francis Galton observed a competition to guess the weight of an ox at a country fair. Eight hundred people entered. Galton, being the kind of man he was, ran statistical tests on the numbers. He discovered that the average guess (1,197lb) was extremely close to the actual weight (1,198lb) of the ox. This story was told by James Surowiecki, in his entertaining book The Wisdom of Crowds.’ [The Parable of the Ox. John Key, Financial Times 2012]

This famous story illustrates the power of collective wisdom. Crowds acting together can be capable of intelligence that goes beyond what any individual can achieve on their own. Just look at how colonies of termites build fantastically complex nest structures that no individual could conceive of alone. Consider how many individuals can contribute to accurate predictions of the future through financial and betting markets. These are instances where the aggregation of individuals with limited information and/or simple models of the world produces a much richer collective system. Fundamentally, all intelligence that we know of is collective intelligence. None of the neurons in your brain has any intelligence of its own. Likewise with the transistors on a microchip. It is only through the interactions and aggregation of these fundamental units that intelligence is produced.

Hence, we have used many models to predict the solubility. The Wisdom-of-Crowd technique can be used to make up for incorrect forecasts and produces superior outcomes.

**General flow of the analysis:**

Diagram

Description automatically generated

Figure **xyz**: chart depicting the general flow of the analysis of some of the models. All the models used have the similar flow.

* Bootstrapping approach is used to split the data into 70% training and 30% testing subsets in every ML models. It creates balanced splits of the data or random stratified splits. It is additionally shuffled into a random order to remove any bias in the dataset's ordering. The importance of this split is that it helps to evaluate the performance and accuracy of the models by testing it with testing data.
* Scaling is performed on the data. The goal was to lessen the impact of data outliers on model performance and to create datasets with consistent scales and units.
* 10-fold cross validation is used to ensure that every observation from the original dataset has a chance of showing up in the training and test sets. It is done by splitting the training data into k groups.
* A range of machine learning model parameter settings were tested to see which values best fit the model and maximize prediction accuracy, which could be determined by picking the least RMSE value.

**Non-linear Models used:**

**Decision tree:**

A decision tree creates tree-like models for classification or regression. It incrementally develops an associated decision tree while segmenting a dataset into smaller and smaller sections. The outcome is a tree containing leaf nodes and decision nodes. The topmost decision node in a tree which corresponds to the best predictor called root node.

Decision trees can handle both categorical and numerical data. The root node is the original set S at the start.  The approach determines the entropy (H) and information gain (IG) of the extremely under-utilized attribute of the set S for each iteration.  The property with the lowest entropy or greatest information gain is then chose. To create a subset of the data, the set S is then divided by the chosen attribute. Only traits that have never been chosen previously are taken into account as the algorithm iterates over each subset.

A screen shot of a computer

Description automatically generated with low confidence

Fig **XYZ**

However, the model is highly unstable because even a minor change in the data causes a significant change in the optimal decision tree's structure. As they have a tendency to closely fit all the samples within training data, there is also a risk of overfitting. But decision tree as the model is very easy to interpret and easy to visualize.

Hyperparamters

\caption{Hyperparameters of Decision Tree Model}

\label{dtreehyper}

**Random Forest:**

**The random forest model makes use of a multiple decision trees. The random forest method expands the bagging strategy by creating an uncorrelated forest of decision trees utilising both feature randomness and bagging.**

**These are based on ensembles of trees, as indicated by the name "forest." The term "random" is used because they generate random numbers via bootstrapping.**

**The figure \ref{rf} shows the workflow of random forest. Initially, multiple subsets of training data is created. A model is built from every subset, and it is done with replacement which is called as row sampling. Row sampling with replacement is known as bootstrap. Here, result from each independently trained model is obtained. The final prediction is based on majority vote. Hence, the result of every model is combined. This process of combining and obtaining output based on majority voting is called as aggregation. \\**

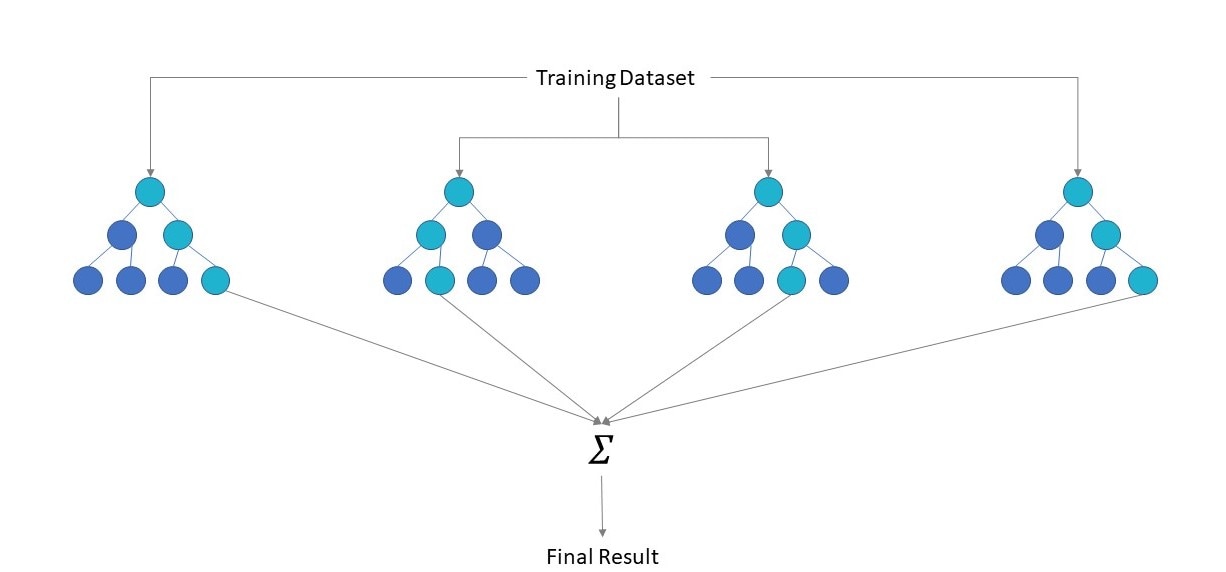
**One of the most important features of random forest is diversity. All features are not considered while building individual model of tree. Hence, each tree is different.**

There are three key hyperparameters for random forest algorithms that must be set prior to training. Node size, tree count, and sampled feature count are a few of them. From there, classification or regression problems can be solved using the random forest classifier.

Numerous decision trees are used in the random forest model. The bagging approach is extended by the random forest algorithm, which builds an uncorrelated forest of decision trees using both feature randomness and bagging.

**\caption{Hyperparameters of Random Forest Model}**

**\label{rfhyper}**



**Fig XYZ**

Each decision tree in the ensemble that makes up the random forest method is composed of a bootstrap sample of data that is taken from a training set with replacement. The out-of-bag (oob) sample is one third of that training sample that is set aside as test data. Feature bagging adds more diversity to the dataset and reduces correlation across decision trees by injecting another randomization instance. The prediction's determination varies depending on the problem kind. The individual decision trees are averaged for a regression job to predict. The out-of-bag sample is finally used for cross-validation, which completes that prediction.

As the variance and prediction error are reduced by averaging uncorrelated trees, the random forest model is less susceptible to overfitting.

**GBM:**

Compared to random forests, gradient boosting trees can be more precise. We train them to correct each other's mistakes, which allows them to recognize intricate patterns in the data. In contrast, noisy data may cause the boosted trees to overfit and begin modelling the noise.

GBM is an iterative and dependence-based algorithm. It is a numerical optimization approach that searches for an additive model that minimizes the loss function. The main idea behind this technique is to build new base-learners that are maximally correlated with the loss function's negative gradient, which is associated with the entire ensemble.



As shown in Figure 1, Each step involves fitting a new Decision tree to the current residual and combining it with the prior model to update the residual. To minimize the error in prediction, previously learnt trees are maintained and new trees are introduced one by one. The trees are grown in a sequential order, with each tree using knowledge from preceding trees. Based on the prior trees created, each tree is fitted to a modified version of the original data set. As it does these steps iteratively, the classifier is reinforced by the number of iterations supplied by the user. Therefore, GBM is based on boosting principle which helps to reduce the bias and variance of the model.[1] Boosting is a continual process of creating classifiers that are strengthened by weights from earlier stages' poor classifiers.

Hyperparameters are variables whose values influence the learning process and affect the model parameters that a learning algorithm learns. They are most impactful parameters that regulate the learning process and the model parameters that come from it, as the prefix 'hyper' suggests. We have multiple hyperparameters in GBM as mentioned in Table 2 which we pass to the algorithm along with our data. This helps to control the learning process of the model.

|  |  |
| --- | --- |
| **Tuning hyperparameters** | Description |
| n.trees | Number of Trees. Default is 100. |
| interaction.depth | Integer specifying the maximum depth of each tree (i.e., the highest level of variable interactions allowed). A value of 1 implies an additive model, a value of 2 implies a model with up to 2-way interactions, etc. Default is 1. |
| n.minobsinnode | Integer specifying the minimum number of samples in the terminal nodes of the trees for splitting. |
| shrinkage | A shrinkage parameter applied to each tree in the expansion. Also known as the learning rate or step-size reduction; 0.001 to 0.1 usually work, but a smaller learning rate typically requires more trees. Default is 0.1. |

The model can be more accurate than random forest as we train it to correct each other’s error which makes it capture complex patterns in the data, but it also tends to overfit

The GBM uses a significant amount of memory. The cost of storing a predictive model is determined by the number of boosting iterations used, and in order to avoid overfitting, the optimal number of iterations for an appropriate shrinkage parameter can be quite large.

**Neural Network:**

Artificial neural networks (ANNs) are adaptable artificial systems that draw inspiration from how the human brain works. These are systems with the capacity to alter their internal organization in response to a function objective. Due to their ability to recreate the fuzzy rules governing the ideal solution for problems of this type, they are particularly well suited for dealing with nonlinear difficulties in the dataset.

As we have seen, our dataset contains both linear and non-linear relationships between the variables, and ANN is able to learn from both of these types of interactions. The neural network is a fairly complicated method that is ideally suited for handling huge datasets with many features. As a result, if we only have a small number of data (for instance, fewer than 100), we are more likely to produce an overfit model since neural networks are more complex than trees-based models.

The nodes, also known as processing elements (PE), and connections are the fundamental components of an ANN. Each node has its own output, through which it communicates with other nodes or the environment, as well as its own input, through which it receives messages from other nodes and/or the environment. Last but not least, each node has a function f that it uses to convert its own global input into output (**Fig. XYZ**). The degree to which pairs of nodes are activated or inhibited defines each link. Excitatory connections are shown by positive values, while inhibitory connections are indicated by negative values.

Diagram

Description automatically generated

Over time, the connections between the nodes can change. The entire ANN begins to learn as a result of this dynamic. The "Law of Learning" refers to the process through which nodes adapt to their environment. Thus, one of the fundamental features of ANNs, which are regarded as adaptive processing systems, is the learning process. A method of adjusting an ANN's connections to the data structures that make up the environment and, as a result, a method of "understanding" the environment and the relationships that define it, is the learning process.

A node layer of an artificial neural network (ANN) consists of an input layer, one or more hidden layers, and an output layer. Each node, or artificial neuron, is connected to others and has a weight and threshold that go along with it. Any node whose output exceeds the defined threshold value is activated and begins providing data to the network's uppermost layer. Otherwise, no data is transmitted to the network's next layer.

Diagram

Description automatically generated

**FIG XYZ**

Training data is essential for neural networks to develop and enhance their accuracy over time.

With input data, weights, a bias (or threshold), and an output, each node has its own linear regression model.

∑wixi + bias = w1x1 + w2x2 + w3x3 + bias

Weights are assigned after a determination of the input layer. These weights aid in determining the significance of each variable, with larger weights having a greater impact on the final result than smaller ones. Next, each input is multiplied by its corresponding weight before being added together. The output is then determined by an activation function once the output has been passed through it. This "fires" (or activates) the node, sending data to the network's next layer, if the output rises beyond a predetermined threshold. As a result, the input of one node becomes the output of the following node.

The activation function used in our model to predict solubility is \textbf{Rectified Linear Unit}. It is a non-linear function which is computationally efficient comapred to other activation functions like $\tanh$ and $sigmoid$ as only certain number of neurons are activated.\\

But disadvantages of using ANN includes requirement of greater computational resources and have limited ability to explicitly identify possible causal relationships. An additional drawback of a neural network model's capacity to implicitly account for interactions and nonlinearities is that it may result in an overfitting of the training data set and subpar performance on outside test data sets. In general, there are three major strategies to avoid overfitting: reducing the number of hidden nodes, including a penalty term in the objective function for big weights, or restricting the amount of training using cross-validation.

**KNN:**

The k-nearest neighbours algorithm, sometimes referred to as KNN or k-NN, is a supervised learning classifier that employs proximity to produce classifications or predictions about the grouping of a single data point. It is famous for its ease of use, understandability, and scalability. It is simple to comprehend.

To forecast the values of any new data points, the KNN algorithm makes advantage of "feature similarity." In other words, the value given to the new point depends on how much it resembles the points in the training set. The distance between each training point and the new point is first computed. The k data points that are closest are chosen (based on the distance). The final forecast for the new point is the average of these data points.

Calculating the distance between each training point and the new point is the first step. There are several ways to figure out this distance, but the three that are most widely used are the Euclidian, Manhattan (for continuous), and Hamming distances (for categorical).

1. Euclidean Distance: Euclidean distance is calculated as the square root of the sum of the squared differences between a new point (x) and an existing point (y).
2. Manhattan Distance: This is the distance between real vectors using the sum of their absolute difference.

Text

Description automatically generated

1. Hamming Distance: It is used for categorical variables. If the value (x) and the value (y) are the same, the distance D will be equal to 0 . Otherwise, D=1.

Text, letter

Description automatically generated

Selecting the k value is the next step. This specifies how many neighbours we take a look at before giving any new observation a value.

The model overfits on the training data for a very low value of k (let's say k=1), which causes a significant error rate on the validation set. On the other hand, the model performs poorly on both the train set and the validation set with a high value of k. Hence, the model must be tuned properly to select the best k.

**SVR:**

**The SVM regression algorithm is referred to as Support Vector Regression or SVR.**

Support vector machines (SVMs) are a set of supervised learning methods used for classification, regression and outliers detection.

Finding a hyperplane in an n-dimensional space that clearly classifies the data points is the goal of a support vector machine algorithm. Support Vectors are the nearest data points to the hyperplane on either side of the hyperplane. These have an impact on the hyperplane's position and orientation, assisting in the SVM construction.

the main idea is always the same: to minimize error, individualizing the hyperplane which maximizes the margin, keeping in mind that part of the error is tolerated.

Hyperparameters in SVR:

1. Hyperplane: A decision boundary used to predict continuous output; hyperplanes Support Vectors are the closest-to-the-hyperplane data points on either side of the hyperplane. These are utilised to draw the necessary line that depicts the algorithm's projected outcome.
2. Kernel: A kernel is a collection of mathematical functions that take data as input and change it into the desired form. In higher dimensional space, these are typically employed to locate a hyperplane.

Chart, scatter chart

Description automatically generated

**FIG XYZ**

The most often used kernels are Sigmoid, Radial Basis Function (RBF), Non-Linear, Polynomial, and Linear. RBF is used as the kernel by default. Depending on the dataset, any one of these kernels may be utilised.

The kernel functions transform the data into a higher dimensional feature space to make it possible to perform the linear separation.

For our dataset, RBF has been used to predict solubility (target variable).

1. Boundary Lines: These are the two lines around the hyperplane at ε (epsilon). The margin between the data points is produced using it.

Support The SVMs and Vector Regression both operate on the same basis. Finding the optimum fit line is the fundamental tenet of SVR. The hyperplane with the most points is the best-fitting line in SVR.

The SVR seeks to match the best line within a threshold value, in contrast to other Regression models that aim to reduce the error between the real and predicted value. The distance between the boundary line and the hyperplane is the threshold value.

where $\sigma$ is the variance and \begin{Vmatrix}X\_{i}-X\_{j}\end{Vmatrix} is euclidean distance between $X\_{1}$ and $X\_{2}$. The maximum value of RBF kernel is 1 when $X\_{1}$=$X\_{2}$ , as $exp(0)=1$. But when the distance between the points is large, the value of the RBF kernel is less than $1$ and quite close to $0$. Since we can see that the points are less similar as the distance between them grows, distance can be considered as an equivalent to dissimilarity.

In other words, as $\gamma$ increases, model tends to overfit for a particular value of Cost parameter. The right value of $\gamma$ and cost is necessary to obatin best bias and variance trade-off.

**Linear Models:**

**Linear Regression:**

**\begin{itemize}**

**\item Model is highly sensitive to outliers.**

**\item It can many times capture noisy data resulting in overfitting.**

**\item Model assumes the independent variables to be uncorrelated. Hence, multicollinearity must be removed before building this model.**

**\end{itemize}**

The concept of linear regression was first proposed by Sir Francis Galton in 1894. Linear Regression is a machine learning algorithm based on supervised learning. It performs a regression task. Regression models a target prediction value based on independent variables. It is mostly used for finding out the relationship between variables and forecasting. Different regression models differ based on – the kind of relationship between dependent and independent variables they are considering, and the number of independent variables getting used.

Linear regression performs the task to predict a dependent variable value (y) based on a given independent variable (x). So, this regression technique finds out a linear relationship between x (input) and y(output). Hence, the name is Linear Regression.

When training the model – it fits the best line to predict the value of y for a given value of x. The model gets the best regression fit line by finding the best θ1 and θ2 values.

θ1: intercept

θ2: coefficient of x

Once we find the best θ1 and θ2 values, we get the best fit line. So when we are finally using our model for prediction, it will predict the value of y for the input value of x.

Linear regression is a statistical procedure for calculating the value of a dependent variable from an independent variable. Linear regression

measures the association between two variables. It is a modeling technique where a dependent variable is predicted based on one or more

independent variables.

Linear regression is a statistical test applied to a data set to define and quantify the relation between the considered variables.

The linear regression analysis uses the mathematical

equation, i.e., y = mx + c, that describes the line of best fit

for the relationship between y (dependent variable) and x

(independent variable).

Linear regression assumes that there exists a linear relationship between the dependent (target variable) and independent variables.

**The**

**Lasso Regression:**

When a set of candidate variables is subjected to typical regression techniques, the number of variables that are eventually included in the model and the model's capacity to use the included factors to explain observed variability are both overestimated (a phenomenon known as optimism bias). When predicting data with a higher or lower degree of "extremity," the model typically performs badly.

There are various regularization regression techniques which can be used to address this problem. Regularization solves the problem of overfitting which results in low accuracy of the model. If a regression model uses the L1 Regularization technique, then it is called Lasso Regression. If it used the L2 regularization technique, it’s called Ridge Regression.

LASSO (Least Absolute Shrinkage and Selection Operator) regression is a shrinkage and variable selection method for regression models. The aim of this model is to identify the factors and accompanying regression coefficients that result in a model with the lowest possible prediction error. This is accomplished by placing a restriction on the model parameters, which "shrinks" the regression coefficients towards zero, or by ensuring that the total of the absolute value of the regression coefficients be less than a specified value (λ). Practically speaking, this limits the model's complexity. After shrinking, variables with a regression coefficient of 0 are not included in the model. Hence, less important features are eliminated from the model.

Due to presence of multicollinearity in our dataset, this model can be helpful to predict solubility. Additionally, its prediction will serve to add to the ensemble of all the models.

Mathematical equation of Lasso Regression:

Text

Description automatically generated

Where,

* λ denotes the amount of shrinkage.
* λ = 0 implies all features are considered and it is equivalent to the linear regression where only the residual sum of squares is considered to build a predictive model
* λ = ∞ implies no feature is considered i.e, as λ closes to infinity it eliminates more and more features
* The bias increases with increase in λ
* variance increases with decrease in λ

In some situations, LASSO regression has been demonstrated to perform better than conventional techniques. It does not, however, eliminate the requirement to test a model using data from a different dataset or provide a fix for the issues of overfitting and optimism bias. Furthermore, the LASSO technique compromises any bias in parameter estimation for a better anticipated overall forecast. Due to the LASSO approach's emphasis on the best combined prediction rather than the quality of the estimation and interpretation of the contribution of individual variables, the regression coefficients may not be consistently interpretable in terms of independent risk factors.

Log lambda represents the penalizing factor for the sum of absolute values of coefficient.

[**https://glmnet.stanford.edu/articles/glmnet.html**](https://glmnet.stanford.edu/articles/glmnet.html) **use it**

**Ridge Regression:**

**Ridge regression reduces the overfitting by adding a penalty term to the error function that shrinks the size of the coefficients. Ridge regression is similar to ordinary least squares regression, but the penalty term ensures that the coefficients do not become too large. This can be beneficial when there is a lot of noise in the data, as it prevents the model from being too sensitive to individual data points. Ridge regression is often used in conjunction with other machine learning methods, such as cross-validation, to further reduce overfitting. Ridge regression is also less sensitive to outliers than linear regression.**

Ridge regression is one of the more popular, albeit controversial, estimation procedures for combating

* Performs L2 regularization, i.e. adds penalty equivalent to **square of the magnitude** of coefficients

Multicollinearity

Lambda is the penalty term. λ given here is denoted by an alpha parameter in the ridge function. So, by changing the values of alpha, we are controlling the penalty term. The higher the values of alpha, the bigger is the penalty and therefore the magnitude of coefficients is reduced.

* It shrinks the parameters. Therefore, it is used to prevent multicollinearity
* It reduces the model complexity by coefficient shrinkage
* Check out the free course on [regression analysis](https://www.mygreatlearning.com/academy/learn-for-free/courses/regression-analysis-with-excel-hands-on?gl-blog_id=20944).

Bias and variance trade-off is generally complicated when it comes to building ridge regression models on an actual dataset. However, following the general trend which one needs to remember is:

1. The bias increases as λ increases.
2. The variance decreases as λ increases.

As we have known that our dataset suffers from multicollinearity. Ridge regression is a model-tuning technique that is used to analyze any multicollinear data. L2 regularization is done using this technique. The predicted values deviate significantly from the actual values when the multicollinearity problem is present, least-squares are unbiased, and variances are substantial.

**Ridge regression, however, also has certain drawbacks. First, if the data set is large, it could be computationally expensive. Second, because the Ridge term or L2 norm modifies the coefficients, ridge regression findings might be challenging to interpret. This is due to the cost function having a quadratic term, which makes optimization more challenging.**

**Result & Discussion:**

**Decision Tree:**

Recursive Partitioning or rpart library in R has been used to build decision tree model. It provides a powerful framework for growing classification or regression trees. The use of anova method leads to regression trees. A large decision tree is built initially by choosing a very small value for cp which stand for ‘complexity parameter.’ It is used to determine the optimal tree size and control the size of the decision tree. Tree building terminates if the cost of adding another variable to the decision tree from the current node exceeds the value of cp. We could alternatively argue that tree construction will stop unless the overall lack of fit is reduced by a factor of cp. Putting CP value as zero will build a tree with its maximum depth. Hence, it will build a very large tree.

Chart

Description automatically generated with medium confidence

The lowest X Relative Error or Cross Validation error gives the optimal value of CP. The optimal value of CP obtained is 0.0007681 with 40 number of splits. After that, the cross-validation increases slowly. All the features are used to predict the solubility.

Hence, the tree is pruned using optimal cp value. The **figure xyz** shows the final tree model with 40 splits.

Chart

Description automatically generated with medium confidence

**Figure xyz** shows the variable importance given by decision tree model.

The predicted LogS in each terminal node is displayed alongside the number of observations from the training dataset that are specific to that node. After prediction using the test dataset, the following metrics are obtained:

RMSE: 1.19

\%LogS\pm0.7

LogS+-0.7 = 0.4813

**Random Forest:**

The ranger method has been used to train the random forest model. The ranger package is required for it. The tuning parameters present in this method are:

mtry: It is defined as the number of predictors/features sampled as candidates at each split. In simple words, it refers to number of variables to possible split at in each node. Lower values of mtry lead

to more different, less correlated trees, yielding better stability when aggregating. However, lower values of mtry also lead to trees that 2 perform on average worse, since they are built based on suboptimal variables (that were selected out of a small set of randomly drawn candidates): possibly non-important variables are chosen. We have to deal with a trade-off between the stability and accuracy of the single trees.

If mtry is large, however, these less influential variables might not have the chance to contribute to prediction because stronger variables are preferably selected for splitting and thus “mask” the smaller effects. On the other hand, if there are only a few relevant variables out of many, which is the case in many genetic datasets, mtry should be set high, so that the algorithm can find the relevant variables

While tuning all the range of features has been considered to see the effect. Hence, the range considered is from 1 to 13.

Splitrule: It is defined as splitting rule. Two splitting rules have been considered for tuning. The computational efficiency is improved by choosing randomized split rules. Hence, extremely randomized trees, also known as extratrees has been considered as one of the split rule to check the performance. Not only performance is improved by extratrees but it also adds extra an extra layer of randomness to the trees. Apart from that, Variance is also considered as one of the split rule.

min.node.size: It is the minimum number of observation in the terminal node. Small values of node size results in tree with large depth and hence, more splits are performed until the terminal nodes. It has been observed that increasing the node size decreases the computational time. This parameter helps to control the size of the tree. The range used to tune the model is from 1 to 15.

Graphical user interface, chart

Description automatically generated

The figure shows the plot of the model which has been tuned. As seen from the plot, extratrees starts at a greater value of RMSE when the only one predictor is randomly selected. But we see a drop in RMSE as soon as more than 1 predictors are selected randomly. It is also interesting to see that the RMSE increases with increase in the node size after deceasing for a specific value of node size. With split rule as variance, the difference between the RMSE of more than one randomly selected predictor is quite less but decreases with its increase. Overall, extratrees could go lower in rmse than variance as split rule.

The optimal value of the parameter obtained for best RMSE are:

Mtry: 13, hence all the predictor variables are considered.

Splitrule: ExtraTrees

Min.node.size: 3

The metrics obtained after predicting test data:

RMSE: 0.887

LogS+-0.7: 0.578

**GRADIENT BOOSTING METHOD:**

1. Best parameter combination chosen using RMSE as the metric is shown below.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Shrinkage | Depth | Minobsinnode | Trees | RMSE | R2 | MAE |
| 0.02 | **11** | **1** | **1050** | **0.90** | **0.86** | **0.68** |

1. Predicted against Observed concentration values from the GBM model, with the model fit indicated by the blue linear trend line and the loess trend in red giving an indication of the variation in prediction from this.

Chart, scatter chart

Description automatically generated

1. Prediction Accuracy Measures:

|  |  |  |  |
| --- | --- | --- | --- |
| RMSE | R2 | MAE | LogS+-0.7 |
| 0.88 | **0.87** | **0.67** |  |

**Artificial Neural Network:**

Using Rectified Linear Unit activation function to train the model. Visualizing the training loss vs. validation loss or training accuracy vs. validation accuracy over a number of epochs is a good way to determine if the model has been sufficiently trained. This is important so that the model is not undertrained and not overtrained such that it starts memorizing the training data which will, in turn, reduce its ability to predict accurately.

Hyperparameters such as the number of nodes per layer of the Neural Network and the number of layers in the Network can make a significant impact on the performance of the Model. Visualization of the fitness of the training and validation set data can help to optimize these values and in building a better model.

The training data set is fed to the three-layered Neural networks; with the first two layers having four nodes each and the output layer with just one node. The loss and accuracy data of the model for each epoch is stored in the history object

Adam is a replacement optimization algorithm for stochastic gradient descent for training deep learning models.

A picture containing text

Description automatically generated

Plot shows the training and validation accuracy and loss at each epoch.

Prediction Accuracy Measure by ANN model:

|  |  |  |  |
| --- | --- | --- | --- |
| RMSE | R2 | MAE | LogS+-0.7 |
| 0.88 | **0.87** | **0.67** |  |

**KNN:**

The KNN model has been trained by tuning one hyper-parameter which is ‘K’, which refers to number of nearest neighbours. A range of values of K, from 1 to 50 has been considered to tune the model.

The **figure xyz** shows how the RMSE of the model varies with increase in the number of neighbors.

The RMSE drastically decreases when K increases from 1 to 4 and linearly increases after that. For k=4, we obtain lowest RMSE value.

Chart, line chart, scatter chart

Description automatically generated

Prediction Accuracy Measure by KNN Model:

|  |  |  |  |
| --- | --- | --- | --- |
| RMSE | R2 | MAE | LogS+-0.7 |
| 0.88 | **0.87** | **0.67** |  |

RMSE: 1.03

LogS+-0.7 =

But when compared to tree-based models, it doesn’t perform well at all.

**Linear Regression:**

Linear Regression is performed using lm() function in R. All the features, also known as independent variables, have been used to train the model. The summary of the model is shown in **figure xyz.**



|  |  |  |
| --- | --- | --- |
| **Variables** | **Estimate** | **Standard Error** |
| (Intercept) | -3.01645 | 0.04572 |
| Most\_neg | -1.54276 | 0.11708 |
| G\_sol | 1.46982 | 0.36084 |
| volume | 1.32771 | 0.3553 |
| MW | -1.08376 | 0.24915 |
| Most\_pos | -1.01573 | 0.13045 |
| SASA | -0.88627 | 0.34832 |
| Het\_charges | -0.77541 | 0.14838 |
| O\_charges | -0.56263 | 0.10154 |
| C\_charges | -0.19204 | 0.13136 |
| Lsolv\_Hsolu | 0.12075 | 0.07021 |
| Lsolu\_Hsolv | 0.06003 | 0.08107 |
| DeltaG\_sol | -0.02071 | 0.04961 |
| sol\_dip | 0.01723 | 0.06711 |

The following key points to note from the summary:

* The residuals is providing information about the difference between actual values and predicted values of Solubility (LogS) overall. The residuals are normally distributed.
* The Coefficients in **table xyz** give more idea about the features.
* The estimate shows an average increase in solubility with one unit change in features. We can see some features with positive estimate values have a positive impact on the solubility and others have a negative impact. Hence, the solubility increases with an increase in features having positive estimate values.
* The Standard error tells how precise the estimate of the coefficient is. The smaller the standard error, precise is the estimate. G\_sol is one of the most significant features in predicting solubility as seen from the magnitude of the estimate but also has the highest standard error. Hence, even when the magnitude of the estimate of G\_sol is the second highest in the list, it’s not the second most important feature in predicting solubility.
* The varImp() function of the caret package in R is used to find the feature importance of the Model.

Prediction Accuracy Measure by Linear Regression model:

|  |  |  |  |
| --- | --- | --- | --- |
| RMSE | R2 | MAE | LogS+-0.7 |
| 0.88 | **0.87** | **0.67** |  |

**SVR:**

Library e1071 has been used to build the SVR model. Epsilon regression has been performed on the model.

The default value for epsilon (ϵ) is 0.1

The SVR model has been trained and tuned using multiple values of cost, gamma & epsilon. The cost parameter is used to avoid overfitting. Initially ϵ=0, 0.1, 0.2, ... ,1 and cost = 2^2, 2^3, 2^4, ... ,2^9 was used which trained 88 models. Then the graph has was plotted to see the performance. The darker the region is the better our model is. This way narrowing down to the range of optimal values was calculated. For example, from the plot below we can see the performance is best for epsilon in the range of 0.16 to 0.17 and cost between 4 and 6.

Graphical user interface

Description automatically generated

Hence, after tuning the model in that range. The following plot is obtained:

Chart, histogram

Description automatically generated

The final Parameters used:

SVM-Type: eps-regression

SVM-Kernel: Radial

Cost: 4

Gamma: 0.0769

Epsilon: 0.164

Prediction Accuracy Measure by KNN Model:

|  |  |  |  |
| --- | --- | --- | --- |
| RMSE | R2 | MAE | LogS+-0.7 |
| 0.88 | **0.87** | **0.67** |  |

RMSE obtained after prediction: 0.9827

**LASSO REGRESSION:**

[**https://glmnet.stanford.edu/articles/glmnet.html**](https://glmnet.stanford.edu/articles/glmnet.html) **(use it)**

The glmnet package has been used to carry out the Lasso regression. The lasso model is trained using a range of lambda values, and 10-fold cross-validation is performed to find the lambda value that results in the lowest test mean squared error (MSE). This yields the optimal value of lambda.

Diagram

Description automatically generated

The plot shows that the coeffiecient of the features becomes zero after certain point as the value of lambda keeps on increasing. Again we need to find an optimal value of lambda where the model performs the best.

Cross-validation is performed to find it.

Graphical user interface, chart

Description automatically generated

**Fig xyz**

**The figure xyz** shows that the lambda value which minimizes the test MSE is 0.001. Lastly, the final model is trained using the optimal lambda value.

A picture containing text, receipt

Description automatically generated

Unlike ridge regression which shrinks the coefficients towards zero, lasso model has the potential to remove the less influential features from the model by shrinking the coefficients completely to zero. But here as we see in **table xyz**, none of the coefficient is zero which tells that all the features are important in some or the other way for the prediction. **DeltaG\_sol, sol\_dip, Lsolu\_Hsolv** have coefficient values close to zero.

Prediction Accuracy Measure by KNN Model:

|  |  |  |  |
| --- | --- | --- | --- |
| RMSE | R2 | MAE | LogS+-0.7 |
| 0.88 | **0.87** | **0.67** |  |

**RIDGE REGRESSION:**

The glmnet package has been used again to carry out the Ridge regression. A range of lambda values is used to optimize the model. The minimum value of Lambda is 0.001 and the maximum is 100.

Chart

Description automatically generated

The plot explains the following points:

* The y-axis shows regularized coefficients after penalization is applied.
* The logarithm of the penalization parameter Lambda is displayed on the x-axis. Larger the value of lambda, the higher the shrinkage or reduction of the coefficient magnitude.
* The number of predictor variables is indicated by the number 13 at the top. Since ridge regression does not perform the feature selection, all the variables are used in the model.
* The colourful curves show how the coefficient of predictor variables changes as the penalty term increases. As the Lambda increases, the coefficients get small towards zero. This way more noise is getting removed. But there are two points where the best value of lambda lies. These are shown by red and blue vertical dotted lines.
* The red dotted vertical line shows the minimum value of lambda (λ=0.005) which results in the smallest cross-validation error. This optimal value of lambda is used in the final model to predict solubility.
* The blue dotted vertical line shows the largest value of lambda within the 1 standard error of the minimum value of lambda(λ=0.16). This value is only used for a higher level of penalization. Using this value of lambda makes the model more regularized.

Graphical user interface

Description automatically generated with medium confidence

Cross Validation helps in finding the optimum value of lambda. The vertical division lines tell where the optimum parameters lie. The Y-axis shows the standard error.

The minimum MSE for Cross-validation: 1.35

Lambda for corresponding Min. MSE: 0.005

Minimum MSE for 1 Std. Error: 1.45

Corresponding Lambda for 1 Std. Error: 0.16

Hence, the log of Lambda for corresponding Min. MSE & 1 Std Error are -5.27 and -1.84 respectively as shown in the **fig xyz.**

The metrics obtained from the final model:

Prediction Accuracy Measure by KNN Model:

|  |  |  |  |
| --- | --- | --- | --- |
| RMSE | R2 | MAE | LogS+-0.7 |
| 0.88 | **0.87** | **0.67** |  |

**Note:**

In general, we can find that the tree-based model always predicts better than various linear regression models. The most important variables like Most\_neg, Most\_pos, and Het\_charges shows no linear trend when compared to a variable like **volume**. It can be seen from the plots below.

**Ensembled Models:**

The predictions made by all the models have been averaged to check the performance. The models have been compared using the RMSE as the metric.

Chart, bar chart

Description automatically generated

Figure xyz shows the performance of all the individual models and ensembled models to predict solubility with **RMSE** as the metric.

* Here, we can see that the ensembled model did not perform better than GBM, ANN and Random Forest. The GBM model has still outperformed all the models.
* The decision tree has performed the worst of all the models. This also shows that it is not a good choice for regression problems.
* There isn’t much performance difference between linear, ridge and lasso regression models.
* Out of all the linear models KNN performed the best.

Chart, bar chart

Description automatically generated

Figure xyz shows the performance of all the individual models and ensembled models to predict solubility with **%LogS±0.7** as the metric.

As discussed earlier, %LogS**±0.7** could be a better choice to compare the prediction performance of the models as we have very noisy data. In this case, the ensembled model has turned out to be the 2nd best model. But still, we could see GBM outperforming it. Hence, GBM is still a better choice to predict solubility. Furthermore, it is interesting to see that the SVR model has performed better than ANN & Random Forest model.

Chart, bar chart

Description automatically generated

Figure **xyz** shows the individual performance of simple models and their ensembled models. It is interesting to see how the ensembled model of simple models (as shown in the **figure xyz**) outperforms them. In this case, the average of the prediction made by SVR, KNN, Ridge, Lasso, Linear Regression & Decision Tree model has been taken to create an ensemble model. Models like ANN, Random Forest and GBM are computationally intensive. Hence, this could be a better choice when there is a computational limitation. But this ensembled model still doesn’t give the best prediction accuracy. Hence, this is the only downside of using this model. For prediction accuracy, GBM is still the best one to be used.

**Comparing the best features according to various models:**

Timeline, bar chart

Description automatically generated

Chart, bar chart

Description automatically generated

The **plot xyz** shows how the relative importance of the features is determined by various models. The values are scaled for comparison. We observe the following points from the plot:

* According to most of the models, DeltaG\_sol has the lowest significance in predicting solubility.
* Most\_neg and Volume are the most significant features according to most of the models.
* The Linear models do not consider SASA to be a significant feature while non-linear models do.
* Other Features don’t show any pattern.
* All the models, except for the GBM model, indicate that the relative importance between MW and volume is quite small.

**Comparing all the models using the RMSE metric:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | RMSE | %LogS±0.7 | Rsquared | MAE |
| GBM | 0.84 | 0.645 | 0.877 | 0.64 |
| ANN | 0.882 | 0.585 | 0.873 | 0.694 |
| Random Forest | 0.887 | 0.578 | 0.869 | 0.689 |
| Ensembled | 0.889 | 0.623 | 0.866 | 0.682 |
| SVR | 0.97 | 0.6 | 0.837 | 0.72 |
| KNN | 1.03 | 0.552 | 0.812 | 0.79 |
| Ridge Regression | 1.142 | 0.518 | 0.7728 | 0.858 |
| Lasso Regression | 1.143 | 0.511 | 0.7726 | 0.858 |
| Linear Regression | 1.144 | 0.507 | 0.774 | 0.86 |
| Decision Tree | 1.196 | 0.48 | 0.753 | 0.92 |

Literature review:

Paul G. Francoeur et al. (2021) presented a transformer named SolTranNet, to predict aqueous solubility from a molecule’s SMILES representation. He showed how larger models perform worse at this task, with SolTranNet’s final architecture having 3,393 parameters while outperforming linear.

the entire package PIP installable for PYTHON3 which will allow for both a command line utility and usage in a

PYTHON3 environment. SolTranNet’s dependencies are

RDKit17 (2017.09.1+), NumPy18 (1.19.3), PyTorch19

(1.7.0+), and pathlib (1.0+). SolTranNet also supports

CUDA enabled GPU acceleration via automatic detection

through PyTorch. Installation is done through PIP

ML approaches. SolTranNet has a 3-fold scaffold split crossvalidation

root-mean-square error (RMSE) of 1.459 on AqSolDB

and an RMSE of 1.711 on a withheld test set.

AqSolDB16 is the data set we utilized for

training SolTranNet, Lovric et al.11 provided a set of 829

molecules which they randomly split into a training, validation,

and testing set consisting of 64%, 16%, and 20% of the

molecules, respectively, and evaluated the performance of

random forest, light gradient boosting method, and LASSO

and partial least-squares regression models. In order to optimize the

hyperparameters for SolTranNet, we utilized a two-stage

optimization procedure (Table S1). All hyperparameter

optimizations were performed utilizing the Weights and Biases

platform.23 The first stage was a Bayesian with a hyperband

stopping criteria search over hyperparameters related to the

model optimizer (Figure S1a). The objective of this search was

to minimize the RMSE of the test set for the first fold of the

CCV scaffold split of AqSolDB. This resulted in the selection

of the Huber loss function, the stochastic gradient descent

(SGD) optimizer with a momentum of 0.06, no weight decay,

and a learning rate of 0.04. We then performed a grid search

over the hyperparameters describing the SolTranNet architecture

for 100 epochs over each fold of the CCV scaffold split

of AqSolDB. We additionally evaluated the first

10 models of this search with both 2D and 3D distance

matrices, after which only the 2D versions of the models were

evaluated for the remainder of the sweep. During the grid

search, we evaluated one model for each combination of

hyperparameters. In order to provide a point of comparison,

we evaluated four linear ML models on each fold of the

scaffold split of AqSolDB and the full AqSolDB. These linear

models are LASSO, Elastic Net, partial least-squares, and ridge

regression. Each was implemented through scikit-learn24 and

trained on bit size 2040 RDKit fingerprints for a maximum of

100,000 iterations. In order to select the best performing model, we calculated

the R2 and the root-mean-square error (RMSE) of the

predicted solubility. Since the solubility values in AqSolDB

span several orders of magnitude, the R2 correlation metric is

easier to perform well on. We quantified SolTranNet’s generalization by training five

different models on all of AqSolDB and testing them on our

withheld set (Table S4). An ensemble of five models

outperforms the mean of said models but fails to beat the

best performing seed (Figure S2). As we desire SolTranNet to

be as fast as possible, we elected to deploy a single model with

the best performing seed.

Solubility Prediction from Molecular Properties and Analytical Data

Using an In-phase Deep Neural Network (Ip-DNN)

Atsushi Kurotani, Toshifumi Kakiuchi, and Jun Kikuchi\*

Atsushi Kurotani succeeded at

establishing a solubility prediction tool using a unique machine learning

method called the in-phase deep neural network (ip-DNN), which starts

exclusively from the analytical input data (e.g., NMR information, refractive

index, and density) to predict solubility by predicting intermediate elements,

such as molecular components and molecular descriptors, in the multiple-step

method. For improving the level of accuracy of the prediction, intermediate regression models were employed when performing inphase machine learning.