CHEM452/CHBE413

Chemical Data Science and Engineering

Homework 5

Due Date: October 2nd, 2025

1. **K-Nearest Neighbors Classification of Solubility (dataset =** “*solubility\_classification.csv*”)**.**

A simple classification model is the K-nearest neighbors (KNN) method. In this model, you use the “K” molecules most similar to the molecule you are trying to classify and compute its class as the most probable class from the K-nearest neighbors. More concretely, given a positive integer K and a test observation x0, the KNN classifier first identifies the K points in the training data that are most similar to x0, represented by N0. It then estimates the conditional probability for class j as the fraction of points in N0 whose response values equal j:

I is an indicator function that registers ‘1’ when yj = j, and 0 when it does not. KNN then classifies the test observation x0 to the class with the largest probability. You will be using this method to determine whether the molecules in *solubility\_classification.csv* are soluble in water.

1. (**4 points**) Select the following features (columns) for use in your classification model: [*MolLogP,HeavyAtomCount,NumHAcceptors,NumHDonors,NumHeteroatoms,NumRotatableBonds,NumAromaticRings,NumSaturatedRings,NumAliphaticRings,RingCount,TPSA,BalabanJ*] and use [*Solubility*] as the output. Using a 80/20 training/test split use *sklearn.model\_selection.cross\_val\_score* and *sklearn.neighbors.KNeighborsClassifier* with 5-fold cross-validation to compute the mean and standard deviation of your KNN classifier’s accuracy as a function of K, plotting the results as a function of K (with error bars). I would suggest searching K=1 to K=50 as a generous range. What is the best value of the hyperparameter “K”? What is the performance of K-means with the optimal K on the held-out test set? When you report the error on your final held out test set, use the optimal hyperparameters you found using cross-validation to train one model on the entire training set (all K folds) at once – then apply this model to the held-out test set to make the final estimate.
2. (**2 point**) Using what you know about the Bias-Variance tradeoff, write a few sentences describing how the selected value of “K” in K-nearest neighbor regression influences the bias-variance tradeoff for this model.
3. **(4 points – This is more challenging! Skip it and come back to it last.)** Using the Tanimoto Similarity between Morgan fingerprints (1024 bits, radius 2), write a Python function from scratch to perform K-nearest neighbor classification for a given value of “K”. To do this I would suggest that you write a function that (1) computes the Tanimoto similarity between the molecule you wish to make a prediction on, and all of the molecules in your training set, (2) identifies the “K” molecules most similar to your target molecule using Tanimoto similarities, and (3) computes the most common class amongst those K molecules as the class label. Perform an 80/20 training test split, fit your K-means classifier to the training data using K=7, and then report the classification accuracy on the test data. No need for cross-validation here. You can’t use scikit-learn’s KNN method.
4. **Logistic Regression of Molecular Toxicities (**dataset: **“**Tox21.csv”)

The Tox-21 dataset represents a collaboration between federal agencies to develop new ways to rapidly discover how chemical substances impact health. The Tox-21 dataset includes numerous metrics for molecular toxicity, but here we will focus on the p53 protein. The p53 protein is expressed in cells when they undergo DNA damage, which can transform a normal cell into a cancerous one. When DNA damage occurs, there is a significant increase in p53 expression which is a good indicator of cell health. The Tox21 data was generated by thresholding the values of p53 gene expression under controlled conditions as a function of molecular structure to determine if a molecule was an agonist (activator) of the p53 pathway or not.

1. (**1 point**) Note that not every toxicity measurement has been performed for every molecule in this database. For this problem we will select only the column related to the p53 gene (‘sr-p53’) and the rows containing measurements for that gene. Report .info() of your final dataframe once you have done this.
2. (**1 point**) Use a Morgan Fingerprint with a radius of 3 to convert each SMILES string in your dataset to a 1024-bit vector; this vector will serve as the feature for your logistic regression model.
3. (**4 points**) Using an 80/20 training/test split on the p53 dataset, perform 5-fold cross-validation of your logistic regression model (*sklearn.LogisticRegression*) on the training set using the prediction accuracy metric. Use the regularization strength (*C* – try 10-2, 10-1, 100, 101, 102) as the hyperparameter you are optimizing in your cross-validation. Set *max\_iter* = 1000 in the LogisticRegression object. Make a plot of the prediction accuracy on the training and test datasets as a function of *C* (log-scale), and then report the prediction accuracy of your cross-validated model with the optimal choice of *C* on the held-out test set. When you report the error on your final held out test set, use the optimal hyperparameters you found using cross-validation to train one model on the entire training set (all K folds) at once – then apply this model to the held-out test set to make the final estimate.
4. (**2 points**) How good is the accuracy of the model you have trained? Assume that the absolute simplest classification algorithm you could employ would be to compute the total fraction of ‘1’ labels in your dataset relative to the total number of datapoints and draw a random number with equivalent probability that classifies the test data either ‘1’ or ‘0’. What prediction accuracy would you expect this model to obtain on your training and test data? Try it! Is your logistic regression model better than the most ignorant of guesses?
5. (**1 point**) In classification it is often useful to quantify how your classification model was in error. Compute the number of true positives, false positives, true negative, and false negatives resulting from your model. What was the most common type of misclassification error for your model?
6. (**1 point**) Given the heavy bias of your dataset, suggest a way that you might modify this dataset in future work to try to eliminate this bias, without collecting more data.