ENGR 301
Engineering Applications of Data Science

Laboratory 7
Topic: Materials Engineering; Subtopic: Reaction Prediction

Introduction: Glass making is a part of materials science and engineering that stretches back at least 5,000 years to ancient Egypt and Mesopotamia. At the most basic level, making glass involves heating sand (silicon dioxide, SiO_2), soda ash, and limestone until it melts, then rapidly cooling it. Over the centuries, glass science has evolved, and modern glass making is a complicated process involving a number of complicated synthesis routes. Various additional additives are used to improve the properties of glass, such as hardness, color, or strength. Therefore, by tuning the ingredients at the initial stages, scientists and engineers can make different glasses for different types of applications. In this lab, you will use information about the type and amounts of these initial reactants to predict what type of glass is formed.

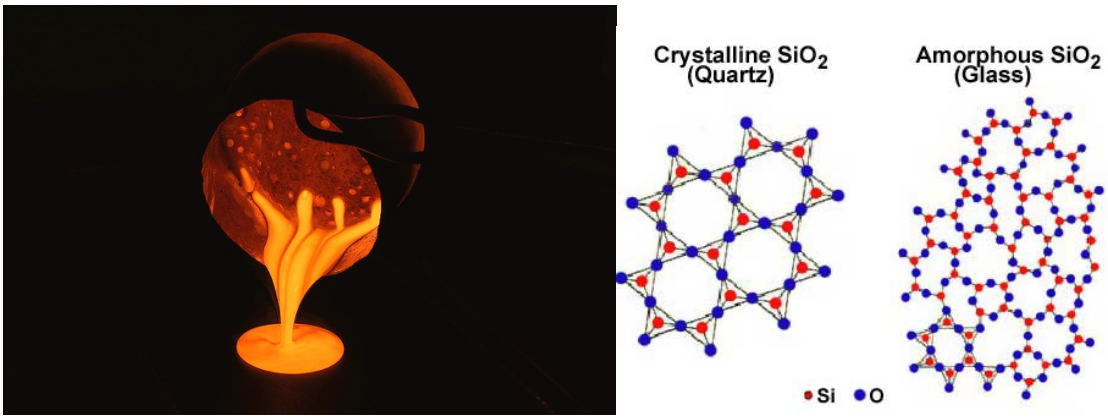

Figure 1. Left: Molten glass (SiO_2, soda ash, and limestone). Right: The atomic structure of crystalline quartz and amorphous glass.
(Images courtesy: https://fineartamerica.com/featured/nuclear-waste-as-molten-glass-pacific-northwest-national-laboratoryus-department-of-energyscience-photo-library.html, https://physicsopenlab.org/2018/02/13/crystalline-and-amorphous-solids/)

Discipline Specific Information: For this laboratory, you are given the amounts of various elements (Na, Mg, Al, Si, K, Ca, Ba, and Fe) used in making a particular sample of glass. The refractive index (RI) and type of glass produced are also given for each sample. There are 215 samples. The goal here is to predict what type of glass is produced given the ingredients; although this may seem straightforward, it is a non-trivial classification problem.

There are seven different types of glass:

1: building windows (float processed)
2: building windows (non-float processed)
3: vehicle windows (float processed)
4: vehicle windows (non-float processed) (none in this database)
5: containers

6: tableware
7: headlamps

Note that in this dataset there are no entries for type 4.

The data was taken from the UCI ML repository:
https://archive.ics.uci.edu/ml/datasets/Glass+Identification

Tasks

Part 1 – Exploratory Data Analysis

1. The movement data for this part is provided on Canvas in a CSV file called "glass.csv"

2. What is the distribution of the different glass types? Is this balanced or imbalanced?

3a. Plot a correlation heat map, a pairplot, and a boxplot. Describe in detail what you notice about the data.

3b. When doing the remainder of the model training, it is likely best to drop the refractive index feature. Why do you think this would be the case?

4. Although we already know how many categories there are, it is instructive to perform a k-means clustering analysis.

4a. Create a for loop in which you loop perform a k-means clustering with k values of 1 to 12 and record the WCSS value for each. Remember that:
- KMeans is part of the sklearn.cluster package
- you must initialize some empty WCSS array
- the number of clusters hyperparameter is "n_clusters"
- you must drop the "Type" column during the fit
- the WCSS of a give KMeans model is found by calling model_name.inertia_

4b. Plot the k value versus the WCSS. What do you think is the ideal number of clusters (k)? Why? Does match the number of types we know exist?

Part 2 – Baseline Models

Although the data set is imbalanced (we will fix this later), it is worthwhile to train some baseline models and see how they perform.

1. Assign x and y, split the data, and perform scaling on the x data.

2. We will first train a support vector classifier (SVC) model, starting by determining the best hyperparameters.

Remember that SVC is part of the sklearn.svm package.

2a. Using GridSearchCV, determine the best model by scanning over these hyperparameters:
- a rbf, poly, and sigmoid kernel
- a regularization (C) of 0.1, 1, 10, 100, and 1000
- a gamma of 0.001, 0.01, 0.1, 1, and scale
- a degree of 1, 2, and 3
- 3-fold cross validation

2b. Create and fit an SVC model with these hyperparameters. Use it to make predications with the test data. What is the model score and accuracy?

3. We will next do the same with a random forest model.

Remember that the RandomForestClassifier is part of the sklearn.ensemble package.

3a. Using GridSearchCV, determine the best model by scanning over these hyperparameters:
- a number of decision trees ranging from 100 to 400 in increments of 50
- a maximum features of sqrt or log2
- a max depth of 4 to 12 in increments of 2
- 3-fold cross validation

3b. Create and fit a random forest model with these hyperparameters. Use it to make predications with the test data. What is the model score and accuracy?

4. Finally, we will do the same with a multilayer perceptron neural network.

Remember that the MLPClassifier is part of the sklearn.neural_network package.

4a. Using GridSearchCV, determine the best model by scanning over these hyperparameters:
- an activation function of relu or logistic
- hidden layer sizes of one layer of 10 neurons, one layer of 10 neurons, two layer of 20 neurons, one layer of 10 neurons, and three layers of 20 neurons
- a regularization (alpha) of 0.0001, 0.005, and 0.01
- 3-fold cross validation

Make sure convergence happens. You may need to increase the max iterations to 5000 to achieve convergence.

4b. Create and fit a MLP model with these hyperparameters. Use it to make predications with the test data. What is the model score and accuracy?

<u>Part 3 – Fixing the Imbalance and Model Improvement</u>

<u>Part 3a – SMOTE</u>

1. As we saw from the first part, the data is highly imbalanced.

2. Using SMOTE, balance the data set. Don't forget to start from a fresh instance of calling x and y from the original data frame.

Following this, split the data and scale it.

3. Re-optimize the hyperparameters of and re-train each of the models from Part 2. Do the optimal hyperparameters change? How does the model score and accuracy change?

Part 3b – Outlier removal

4. Look at the boxplot from Part 1. Do you notice any outliers? Starting from a new data frame, using pandas, remove the outliers in Ca and K. You can decide how to do this, but eye-balling it from the pair plot is probably the easiest.

5. Using SMOTE, re-balance the data set. Don't forget to start from a fresh instance of calling x and y from the data frame from which the outliers were removed.

Following this, split the data and scale it.

6. Re-optimize the hyperparameters of and re-train each of the models from Part 2. Do the optimal hyperparameters change? How does the model score and accuracy change?