**Abstract**

In this report we choose to analyze a set of data from the “Glass” dataset. We wish to use supervised learning techniques to predict the glass-type from a set of properties. After exploring the data and providing macro-trends we apply 3 main supervised-learning algorithms to the dataset: logistic regression, knn, and random forest. We then compare the results of the three algorithms and try to understand the reasons for an algorithm’s success.

**Goal:**

Our primary goal is to predict glass type from a set of properties. After selecting the “best” algorithm (best in terms of performance and accuracy) we wish to understand the reason for the algorithm’s success (on the dataset).

**Description of Data Set**

This dataset is composed of 214 observations and 10 components (9 elements and the “type of glass”). The type of glass is a value from 1-9, where 1-2 are building glasses (float and non-float respectively), 3 and 4 are vehicle glasses (float and non-float respectively), 5 is a container glass, 6 is tableware and 8 is headlamp. Regarding the 9 elemnets we have refractive index, sodium (Na), magnesium (Mg), aluminium (Al), silicon (Si), potassium (K), calcium(Ca), barium(Ba) and iron(Fe). We notice that there are no missing values.

**Key Points**

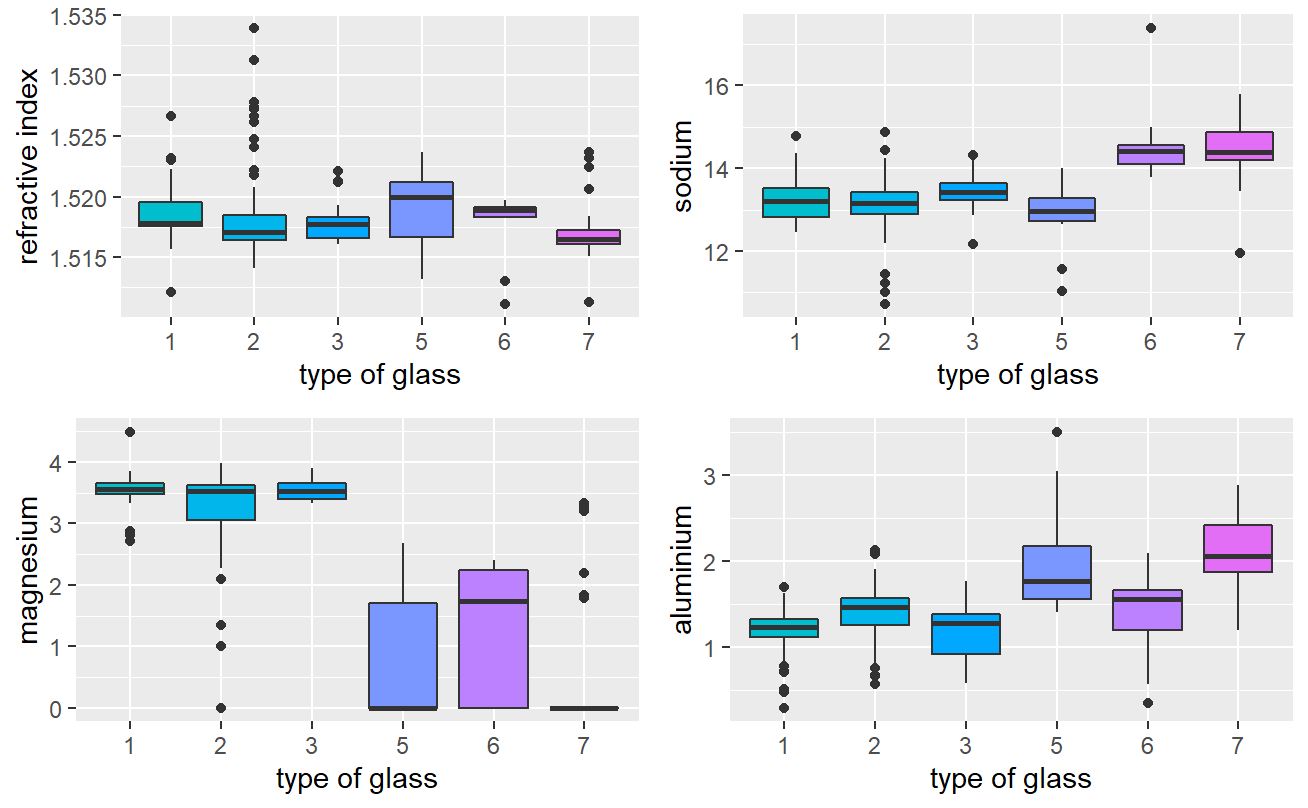
***Key Points - EDA***

Looking at Figure 1, 2 and 3 we notice that .

* O1: (Fig.1) the dataset does not contain any glass of type 4. Moreover the dataset is rather unbalanced. Indeed, the dependent variable (glass type) has an unequal distribution tending to have more of Type 1 and Type 2. We predict that using oversampling could increase the performance of the algorithms.
* O2: (Fig.2) Type 6 and 7 tend to have more Na, type 5 and 7 tend to have more Aluminum. Barium is only present in type 7 and potassium is not present in type 6. Types1 2 and 3 tend to have more Mg and Iron. Overall, Silicon seems to be the main component in all glass types, with Silicon Sodium and Calcium accounting for most of the composition. The refractive index tends to vary from 1.51-1.54. Moreover, we see that there are some outliers, especially so in Barium, Calcium and Magnesium. Overall the data is skewed, we will therefore refrain from using algorithms such as Naïve Bayes.
* O3: (Fig 3) There is a strong positive correlation between the refractive index and calcium and an intermediate positive correlation between Barium and Aluminium and an intermediate negative correlation between Magnesium and Aluminium and Magnesium and Barium. Moreover, the dataset specifies that there is a high correlation between the variables and each other and the variables and the response variable, leading us to think that we may have some issues of collinearity with some algorithms.



Figure 1: Glass Type Plot



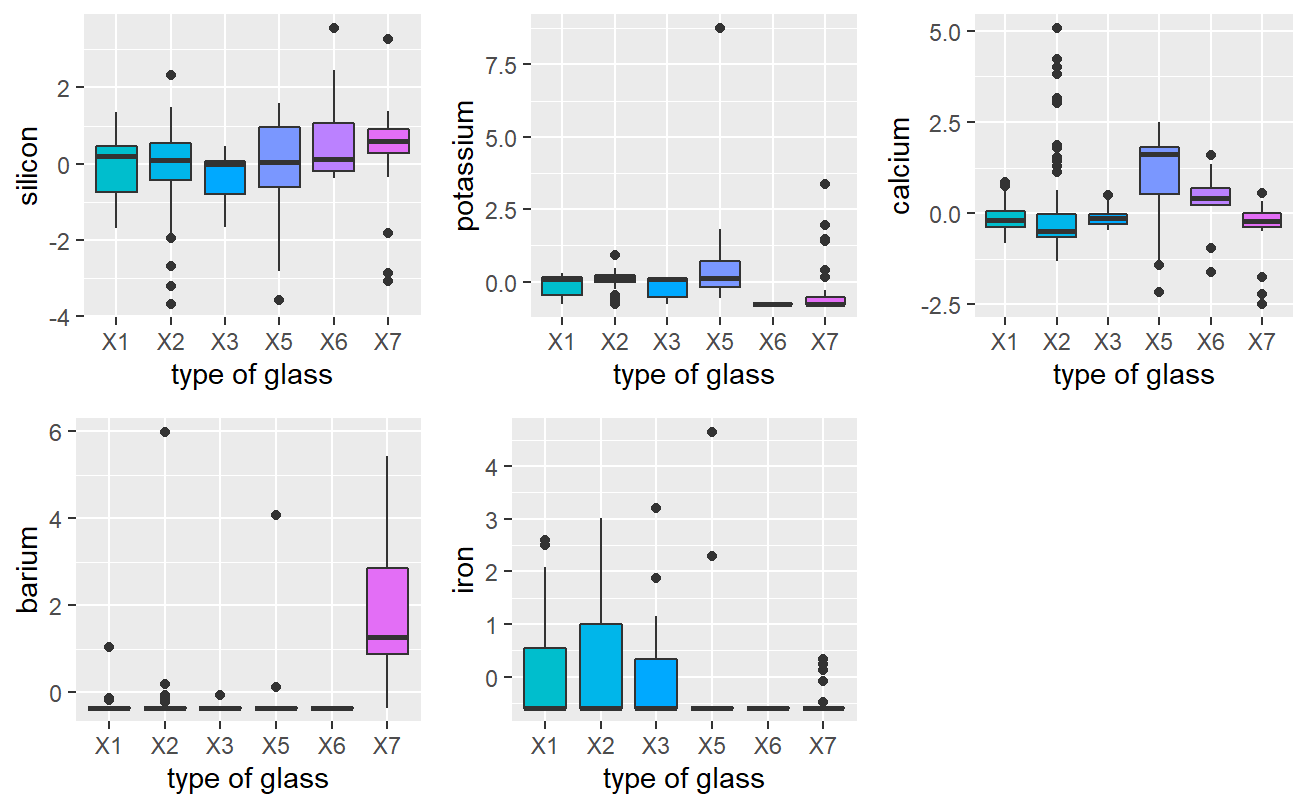


Fig 2 – Box and Scatter Plots

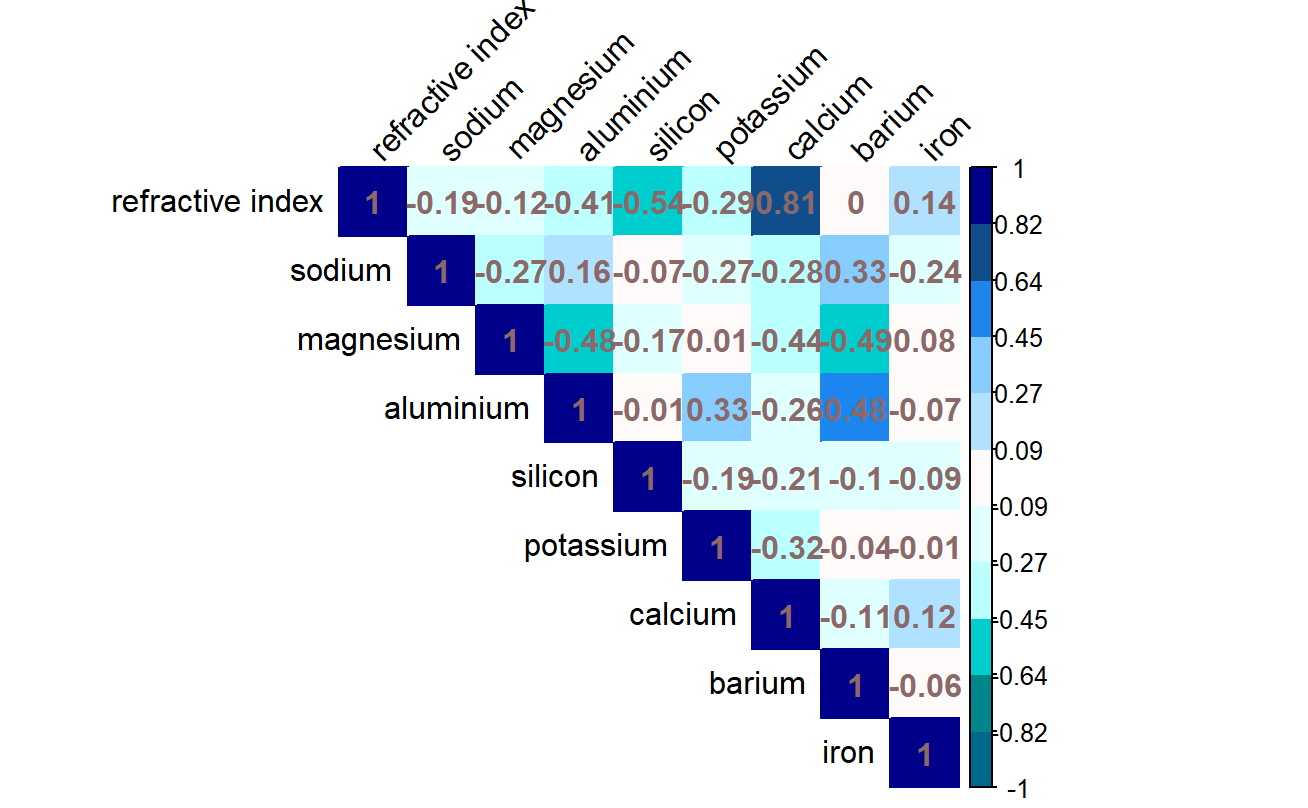


Figure 3 – Correlation

***Key Points and Analysis - Supervised Learning***

For each supervised technique, we choose to normalize the data and split it into a train/test data set (70%/30%). Seeing as the dataset is small, we do not opt for PCA. To reduce the possibility of overfitting, we perform cross-validation (using 5 folds). We end up with 5 samples of similar size (around 118 each).

***Logistic Regression***

We find the following numbers w.r.t. the algorithm’s performance (see table 1)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 5 | 6 | 7 | AVG |
| SENSITIVITY/TRUE POSITIVE RATE | 0.73 | 0.68 | 0.12 | 0.84 | 1 | 0.97 |  |
| SPECIFICITY/TRUE NEGATIVE RATE | 0.78 | 0.84 | 0.98 | 0.98 | 1 | 1 |  |
| PRECISION | 0.62 | 0.71 | 0.33 | 0.73 | 1 | 1 |  |
| ACCURACY |  |  |  |  |  |  | 0.718 |

Table 1 – Results LR

We notice that logistic-regression tends to perform rather badly on Type 3, with points being Predicted as Class 3 although they aren’t (leading to a low TPR and precision). Class 7 is well-predicted although we notice that we had more points available in this class compared to Class 3 and 5. Class 6 is also well predicted although we notice that Class 6 distinctly did not have potassium. We believe that the presence of outliers on Class 1 2 would have contributed to the algorithm’s performance on these classes.

***Random Forest***

Using mtry=5, we find the following numbers w.r.t. the algorithm’s performance (see table 2)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 5 | 6 | 7 | AVG |
| SENSITIVITY/TRUE POSITIVE RATE | 0.96 | 0.95 | 0.82 | 0.92 | 1 | 0.97 |  |
| SPECIFICITY/TRUE NEGATIVE RATE | 0.96 | 0.99 | 0.99 | 1 | 0.99 | 1 |  |
| PRECISION | 0.94 | 0.95 | 0.79 | 1 | 1 | 0.97 |  |
| ACCURACY |  |  |  |  |  |  | 0.94 |

Table 2- Results RF

Here we see that random forest performs quite well overall, with an average accuracy of 0.94, and relatively high values in all classes. We notice again that the lowest value (which is still good) occurs in Glass Type 3 (which again could be because we have scarcity in the dataset). The predictions of Type 1 and Type 2 are quite good as Random Forest tends to be rather robust to noise.

***KNN***

Using k=7, we find the following numbers w.r.t. the algorithm’s performance (see table 3)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 5 | 6 | 7 | AVG |
| SENSITIVITY/TRUE POSITIVE RATE | 0.81 | 0.75 | 0 | 0.53 | 0.33 | 0.86 |  |
| SPECIFICITY/TRUE NEGATIVE RATE | 0.75 | 0.83 | 0.99 | 0.98 | 1 | 0.99 |  |
| PRECISION | 0.61 | 0.71 | 0 | 0.64 | 1 | 0.96 |  |
| ACCURACY |  |  |  |  |  |  | 0.70 |

Table 3 Results KNN

We notice that the KNN tends to have a similar performance to Logistic Regression. However although it tends to outperform KNN on the first two classes (as it is slightly more robust to noise than LR), it is not able to correctly classify Type 3 glass. Moreover it tends to perform less well on the remaining 3 classes (which have relatively few samples when compared to the dataset). If we would have removed the outliers from the dataset” (e.g. removing the points with z score >3), we therefore believe that Logistic Regression would have outperformed KNN in all aspects. As we have seen that KNN seems to have a problem with datasets with few points, we wish to see the performance of the algorithm if we oversample the dataset before applying the algorithm.

***KNN with Oversampling***

By oversampling we create a fictitious dataset with 53 samples of each of the 6 types. Reapplying the random forest algorithm, we get 5 classes of approximately 254. With k=2 (fewer neighbors when compared to the previous run) we reach the optimal model and list the results in Figure 4

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 5 | 6 | 7 | Average |
| SENSITIVITY/TRUE POSITIVE RATE | 0.72 | 0.55 | 0.82 | 1 | 1 | 0.90 |  |
| SPECIFICITY/TRUE NEGATIVE RATE | 0.90 | 0.94 | 0.85 | 0.95 | 0.99 | 1 |  |
| PRECISION | 0.78 | 0.86 | 0.33 | 0.60 | 0.75 | 1 |  |
| ACCURACY |  |  |  |  |  |  | 0.71 |

Table 4 Results – RF with Oversampling

Although we notice that the accuracy is relatively the same, we notice significant improvements for class 3 – 7 (the classes which tended to be oversampled).

**Conclusion**

Overall, we notice that the random forest algorithm tends to perform best on this dataset. We believe that the reasons are as follows:

* The data is categorical
* RF tends to be robust to outliers and can handle multicollinearity (e.g. correlated variables and data correlated with the response variable).
* RF tends to perform decently even when we do not have a large number of points in the training set (e.g. when compared to KNN).