Comp 379 - Terpenes Classification

by Jordan Nazemi, Brandi Letsche, Ian Cummings, and Ted Fu

I - Introduction

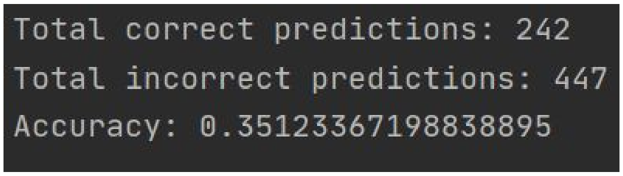
Continued legalization of cannabis has made the industry grow exceptionally fast in the past 10 years, which also provided more data of the genetic makeup of different marijuana strains. Terpenes, which are the aromatic compounds found in marijuana, has been proven in many studies to be noticeably distinguishable to consumers. Terpenes also may play a key role in differentiating the effects of various cannabis strains. Under this circumstance we started this project trying to answer this question: “Can you predict subjective strain labels from a terpene analysis?” By studying this question, we are taking several different approaches of models, analyzing the different results and finding out what we should do with the data and how to improve the accuracy of the prediction.

II - Dataset Description

We sourced our data from a GitHub repository that skimmed reports from SClabs, a lab that uses gas chromatography to identify the terpene contents of cannabis strains with over 5,000 samples. The dataset contained the composition of strains using 35 different terpenes (so values are between 0 and 1 representing percent composition). We removed 6 terpene features containing sparse data, so we ultimately had 29 features to our dataset. We then scraped AllBud, a website containing reviews for different cannabis strains on subjective measures such as aroma, flavor, and effect, for matching strains that were contained in the SClabs report. From this we were able to find around 500 matching strains, which we used to append the subjective labels to the terpene contents of each respective strain. We decided to focus on classifying strictly using the aroma data, which we simplified from 47 different aromas down to 10: “diesel”, “earthy”, “pine”, “citrus”, “fruity”, “skunky”, “nutty”, “sweet”, “spicy”, and “herbal”.

III - Baseline Approach

The baseline model we developed uses a very simple weighted random-guess for each aroma in each entry. For example, if 33 out of 100 aromas were “pine” then “pine” would be randomly predicted for an entry 33% of the time. These percentages are not exclusive for each aroma. Using this method the accuracy seemed to around ~35%

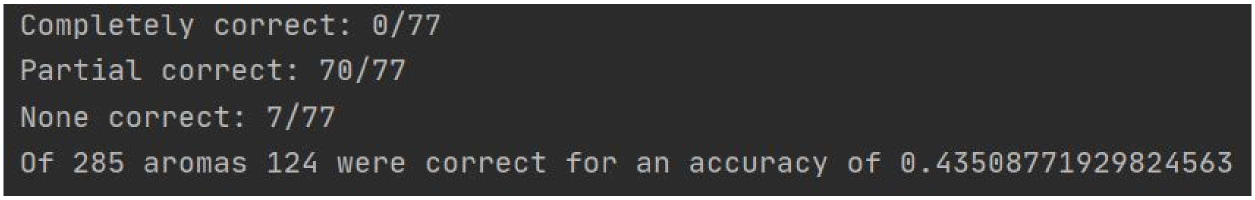


```
Total correct predictions: 242
Total incorrect predictions: 447
Accuracy: 0.35123367198838895
```

IV - Method description

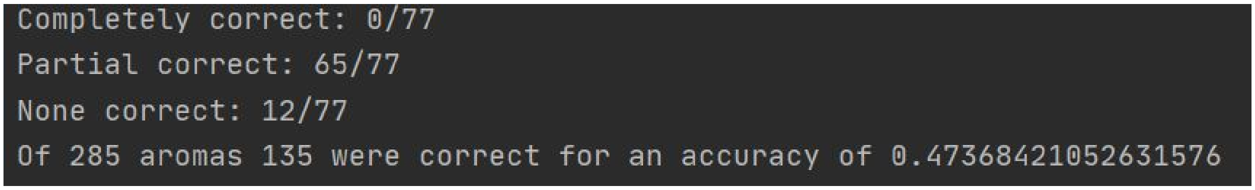
Our team tried numerous methods on the dataset and compared results. For each model the data was trained in an all-vs-one method since multi-label classification proved difficult to do in SKlearn (it required a poorly documented MultilabelBinarizer we couldn’t get working).

Our first model was trained originally using Logistic Regression, min-max regularization, and an all-vs-one approach. The ensemble classification was then evaluated on the development set. We did a manual grid search and tried a number of different hyperparameters. Our best results were using the *liblinear* solver and *L1* penalty. However, as you can see below, early results were not promising with an overall accuracy in the mid-40s.



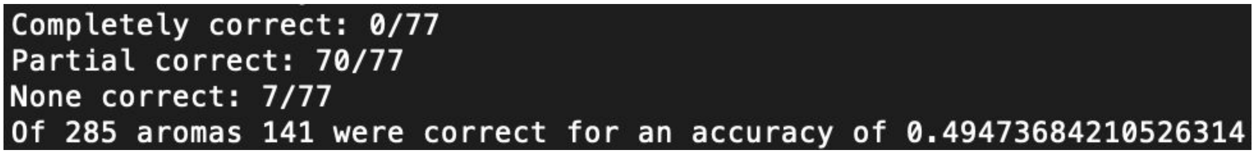
```
Completely correct: 0/77
Partial correct: 70/77
None correct: 7/77
Of 285 aromas 124 were correct for an accuracy of 0.43508771929824563
```

After many failures to improve the Logistic Regression accuracy, we changed the classifier to a KNN and the standardization to L1. L1 was decided after looking over our dataset and noticing a large number of low-value and low-importance features, something L1 is useful for dealing with. After testing different n_neighbor values (settling finally on n_neigbors = 5) we were able to achieve the highest accuracy yet, a still measly 48% on the development set.



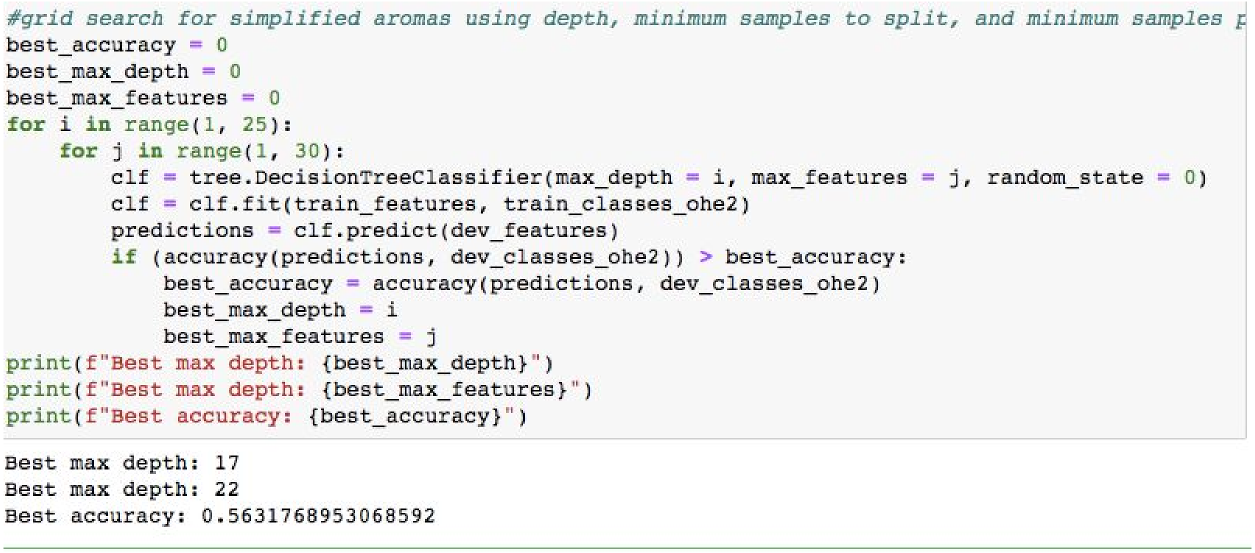
```
Completely correct: 0/77
Partial correct: 65/77
None correct: 12/77
Of 285 aromas 135 were correct for an accuracy of 0.47368421052631576
```

Our next model was trained using the same KNN, L1 regularization, and an all-vs-one approach. A unique aspect of this model is that it was modified to only evaluate a select few of more relevant strains of Terpene: Linalool, beta-Pinene, Limonene, and alpha-Pinene. This was done based on recommendation from the original Github repository we skimmed for the SCLabs data. The ensemble classification was then evaluated on the development set. Once again, we did a manual grid search while tuning hyperparameters. Our best results were k_neighbors = 5 and weight = ‘distance.’ As you can see below, we were able to achieve an increased accuracy of approximately 49%, in comparison to the previous model. .



```
Completely correct: 0/77
Partial correct: 70/77
None correct: 7/77
Of 285 aromas 141 were correct for an accuracy of 0.49473684210526314
```

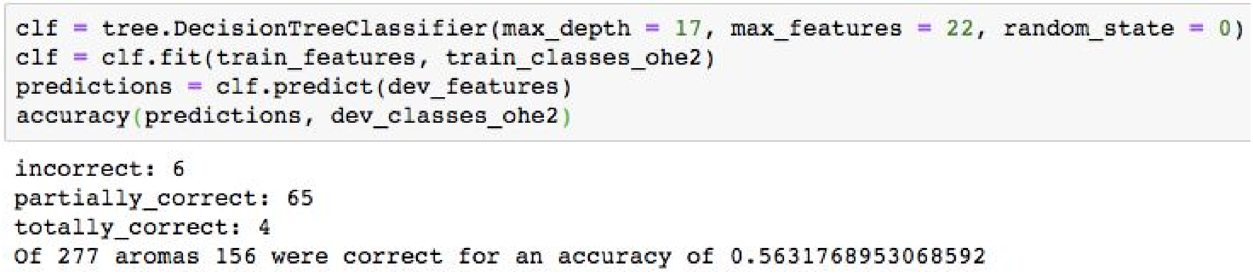
Finally, we implemented a decision tree. The tree was hyperparameter optimized using a grid search over maximum tree depth, from 1 until a pure depth, and over maximum number of features used to evaluate each split, from 1 until 30.



```
#grid search for simplified aromas using depth, minimum samples to split, and minimum samples p
best_accuracy = 0
best_max_depth = 0
best_max_features = 0
for i in range(1, 25):
    for j in range(1, 30):
        clf = tree.DecisionTreeClassifier(max_depth = i, max_features = j, random_state = 0)
        clf = clf.fit(train_features, train_classes_ohe2)
        predictions = clf.predict(dev_features)
        if (accuracy(predictions, dev_classes_ohe2)) > best_accuracy:
            best_accuracy = accuracy(predictions, dev_classes_ohe2)
            best_max_depth = i
            best_max_features = j
print(f"Best max depth: {best_max_depth}")
print(f"Best max depth: {best_max_features}")
print(f"Best accuracy: {best_accuracy}")
```

```
Best max depth: 17
Best max depth: 22
Best accuracy: 0.5631768953068592
```

The optimized model performed as follows on the development set:



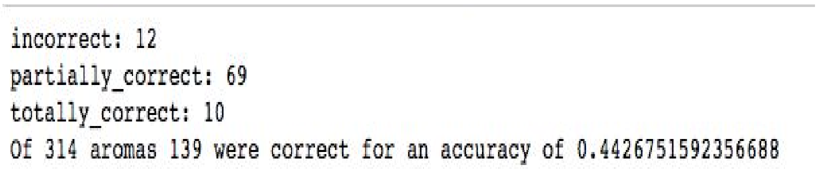
```
clf = tree.DecisionTreeClassifier(max_depth = 17, max_features = 22, random_state = 0)
clf = clf.fit(train_features, train_classes_ohe2)
predictions = clf.predict(dev_features)
accuracy(predictions, dev_classes_ohe2)
```

```
incorrect: 6
partially_correct: 65
totally_correct: 4
Of 277 aromas 156 were correct for an accuracy of 0.5631768953068592
```

The decision tree was able to perform the best of any model on the development set with a whopping 56%, a significant improvement from the first model.

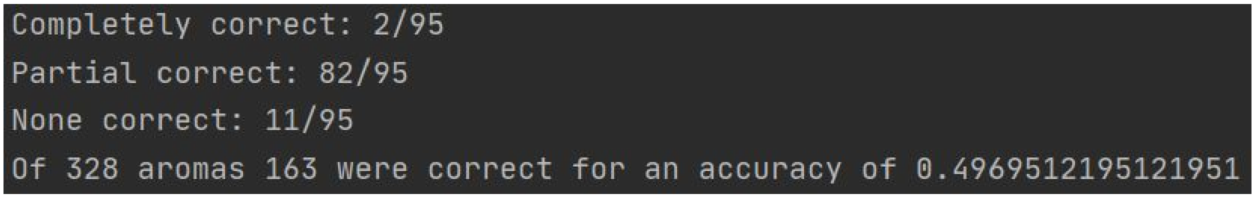
V - Evaluation

As described above, our best results on the development set were using a decision tree. However, when we used that model on the test data the accuracy we got was much lower than expected by almost 10% (44% accuracy). .



```
incorrect: 12
partially_correct: 69
totally_correct: 10
Of 314 aromas 139 were correct for an accuracy of 0.4426751592356688
```

We theorize this is due to an overfitting of hyperparameters during the grid search which did not translate well to the test set. In light of this, we tried our other most successful model: the feature reduced KNN. The results for which were far more consistent with performance on the development set, reaching the same 49% accuracy as it had before. This is our best overall accuracy and represents the final result of our “completed” model.



```
Completely correct: 2/95
Partial correct: 82/95
None correct: 11/95
Of 328 aromas 163 were correct for an accuracy of 0.4969512195121951
```

VI - Conclusion

Overall, our models were not very accurate indicators of aroma labels in the dataset. This could be due to a number of possible factors. The first of which relates to our models themselves; because we opted for an all-vs-one approach I believe we lost a lot of the nuance that a multi-label model could have offered. It could also have been an issue with the dataset. The feature data, as described above, is sourced from laboratory tests but the label data is from publicly available reviews. As such, these reviews may not be accurately labelling the aroma as smell is widely subjective. It could also have been due to the small size of our dataset and the relatively large number of features. Regardless, it does seem to imply at least some kind of correlation between terpene percentages and these subjective labels as we far outperformed our random baseline.

Reducing the number of features, and thus the dimensionality of data, seems to be the best way to achieve greater accuracy and would likely be instrumental in a more complex model. Overall there appears to be a correlation between terpene and aroma that may eventually be accurately predictable from similar terpene lab reports.

Appendix

Dataset creation, logistic regression model, all-vs-one implementation, and KNN model by **Jordan Nazemi**

Reduced-feature KNN by **Brandi Letsche**

Decision tree model by **Ian Cummings**

Attempted Multi-label Binarizer model by **Ted Fu**

Presentation, final code compilation, and report by **The Entire Team**