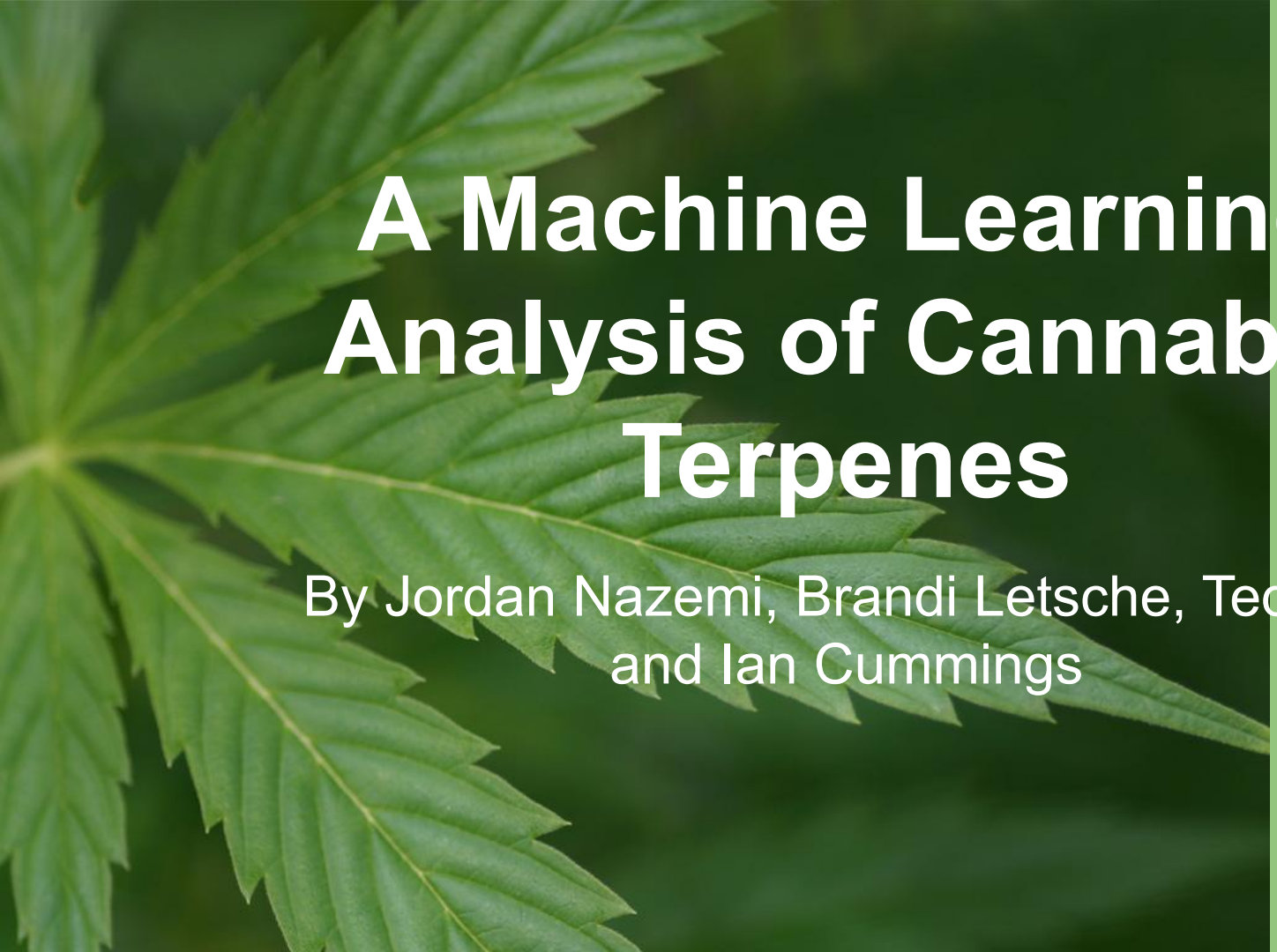

A Machine Learning Analysis of Cannabis Terpenes

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

Intro

- With the growing legalization of the cannabis industry more data is becoming available on the genetic makeup of different marijuana strains

- Terpene makeup has been proven in studies to be, in many cases, noticeably distinguishable to a consumer.

- The question at the core of this project is this: **Can you predict subjective strain labels from a terpene analysis?**

Quick Facts:

- α-Pinene, a chemical in the terpene group, has proven to have a noticeable pine flavor and be instrumental to weed's anti-inflammatory effects

- Terpenes are found in a wide range of flora, and so any discovery found regarding marjuana terpene effects may extend to other plants

- There are over 4 million Americans using medical marjuana and yet science is just beginning to understand the effects terpenes have on the body

Dataset

- Our feature dataset was taken from a Github that had skimmed the data from SClabs, a chemical lab that uses gas chromatography to identify 35 terpenes for over 5,000 strains
  - Of those 35, we removed 6 inconsistently reported terpenes for a total of **29 features** ranging from 0 to 1 (percentage makeup)
- The label data was skimmed via a Python application from Allbud, which has user-submitted reviews for strains containing aroma, taste, and effect
  - The skimmer was only able to match up some of the strains, so our final dataset was **~500 entries with 29 features each**

Entry Example:

Name - Gorilla Glue
3-Carene - 0
Camphene - 0.003
Eucalyptol - 0.002
beta-Myrcene - 0.299
…
Aromas - 'Berry', 'Fruity', 'Pine', 'Sweet'
Flavors - 'Berry', 'Citrus', 'Lemon', 'Sweet', 'Tangy'
Effects - ‘Body High', ‘Cerebral

Baseline Approach

- Our baseline model 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.

- This required first counting the number of aromas in each category to find the percentage likelihood of each one appearing

Percent likelihood:

```
diesel: 4.0%
earthy: 19.0%
citrus: 11.0%
pine: 8.0%
fruity: 11.0%
nutty: 2.0%
sweet: 16.0%
spicy: 8.0%
skunky: 13.0%
herbal: 8.0%
```

Results:
Total correct: 242
Total incorrect: 447
Accuracy: ~35%

Logistic Regression

- **Pre-processing the data:** To simplify the problem, this model focused solely on Aroma of which there were **48 categories.** To reduce this number even further, a number of similar categories were combined
  - For example: ‘piney’, ‘woody’, and ‘sawdust’ were simplified to ‘pine’

- **Regularizing the data:** We first attempted to use Sklearn’s min-max regularization

- **Training:** Data was split using the holdout method and the model trained using a custom all-vs-one implementation
  - A manual grid-search was done, the following results were the best achieved with **penalty = l1** and **solver = liblinear**

Results:

```
diesel - accuracy: 0.8701298701298701
earthy - accuracy: 0.8051948051948052
citrus - accuracy: 0.6753246753246753
pine - accuracy: 0.7402597402597403
fruity - accuracy: 0.5714285714285714
nutty - accuracy: 0.974025974025974
sweet - accuracy: 0.5064935064935064
spicy - accuracy: 0.6883116883116883
skunky - accuracy: 0.45454545454545453
herbal - accuracy: 0.6493506493506493
```

Overall Accuracy:
124 aromas out of 285 were predicted correctly for an accuracy of **~43%**

- 0 of 77 fully correct
- 70/77 partially correct
- 7/77 none correct

K Nearest Neighbor

- **Pre-processing the data:** The data was pre-processed identically to the Logistic Regression model

- **Regularizing the data:** After doing more research, we came to the conclusion L1 is a more appropriate regularization for our problem since the dataset includes a large number of low-value, low-priority features

- **Training:** Data was split using the holdout method and the model trained using the same all-vs-one implementation
  - A manual grid-search was done, the following results were the best achieved with **n_neighbor = 5**

Results:

```
diesel - accuracy: 0.8441558441558441
earthy - accuracy: 0.7402597402597403
citrus - accuracy: 0.6233766233766234
pine - accuracy: 0.6883116883116883
fruity - accuracy: 0.6103896103896104
nutty - accuracy: 0.961038961038961
sweet - accuracy: 0.36363636363636365
spicy - accuracy: 0.6753246753246753
skunky - accuracy: 0.5064935064935064
herbal - accuracy: 0.6233766233766234
```

Overall Accuracy:
135 aromas out of 285 were predicted correctly for an accuracy of **~47%**

- 0 of 77 fully correct
- 65/77 partially correct
- 12/77 none correct

Reduced-Feature KNN

- **Pre-processing the data:** The data was pre-processed identically to the first Logistic Regression model. To simplify the model further, however, we limited our model to make predictions based upon only four out of the original label attributes.
  - Linaloo, beta-Pinene, Limonene, and alpha-Pinene
  - This was done based on recommendation from the original Github repository we skimmed for the SCLabs data

- **Regularizing the data:** the model reflects L1 regularization.

- **Training:** Data was split using the holdout method and the model trained using the all-vs-one implementation. The results were achieved using weighted distance with n_neighbors = 5.

Results:

```
diesel - accuracy: 0.8181818181818182
earthy - accuracy: 0.7662337662337663
citrus - accuracy: 0.5324675324675324
pine - accuracy: 0.6753246753246753
fruity - accuracy: 0.5454545454545454
nutty - accuracy: 0.974025974025974
sweet - accuracy: 0.5584415584415584
spicy - accuracy: 0.6753246753246753
skunky - accuracy: 0.42857142857142855
herbal - accuracy: 0.6233766233766234
```

Overall Accuracy:
141 aromas out of 285 were predicted correctly for an accuracy of **~49%**

- 0/77 fully correct
- 70/77 partially correct
- 7/77 none correct

Decision Tree

- **Pre-processing the data:** The data was pre-processed identically to the first Logistic Regression model

- **Training:** The model was trained using the holdout method, and hyperparameters were tuned using a grid search. The grid search resulted in a max tree depth of 17 and and a maximum of 22 features evaluated at each split.

- **Results:**

```
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
```

Evaluation

- We first evaluated our best model, the decision tree, on the test set as it had the highest accuracy during development
  - **BUT** it proved to have widely different performance when faced with the test set, perhaps due to an overfitting of hyperparameters

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

- We decided to try and evaluate on our next best model, the feature-reduced KNN.
  - Performance turned out to much more in line with what we got from the development, with a final accuracy of **~49%**

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

Conclusion

- There are a number of possible issues that could explain our low accuracy:
  - Biased/inaccurate label data (it was sourced from un-scientific sources)
  - Too much categorial simplification, we may have last nuance when reducing aroma from 48 to 10
  - A relatively small dataset with only around 500 examples

- In conclusion, our model was not exceptionally accurate at predicting aroma via terpene makeup **HOWEVER** it was a significant improvement from the baseline and seems to imply an, atleast, weak correlation