Mushroom Analysis

Suganya Balasubramanian, Bianca Calin, Emily Costello, Katie Darden, & Sam Farber

Selected Topic

Analyzing features of mushrooms (e.g. cap color, habitat, odor, stalk shape, etc.) to determine if they hold insight into whether a mushroom is edible or poisonous

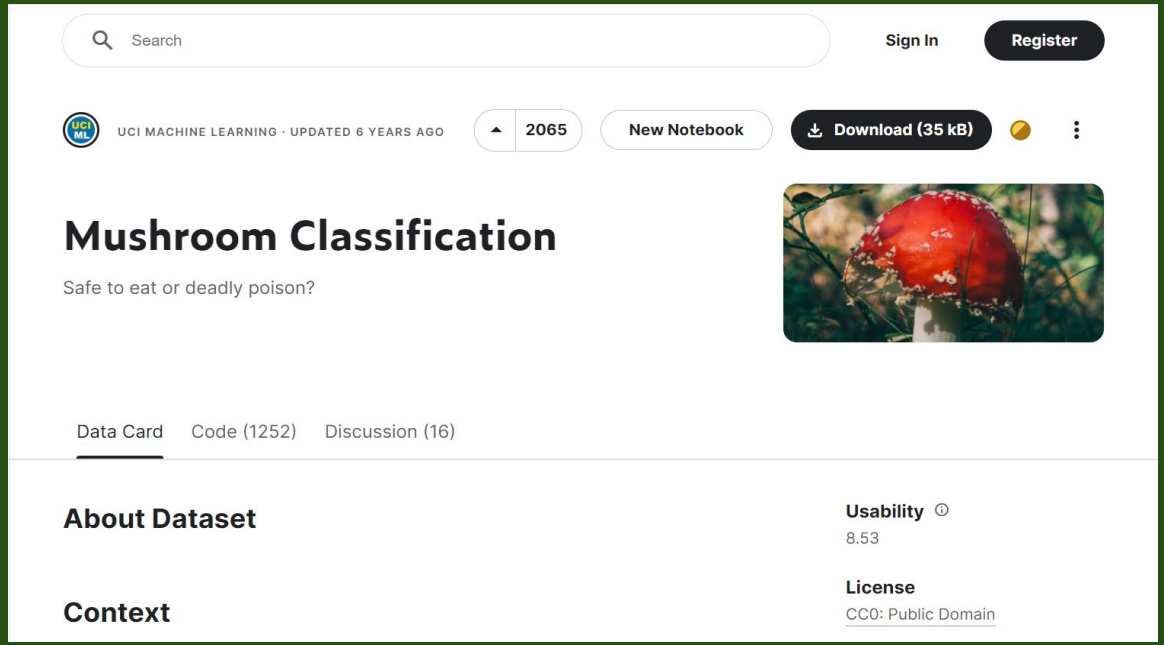
Why did we select this topic?

All five of us are massive mushroom enthusiasts and need to ensure that we can survive the upcoming climate apocalypse in the wild.

Description of Data Source

- CSV file with >8,000 mushrooms and their features
  - Descriptions of hypothetical samples
  - Corresponds to 23 different species
  - Each identified as edible or poisonous

- Furnished by The Audobon Society Field Guide to North American Mushrooms (1981) and contributed to UCI Machine Learning Repository in 1987



Questions we were aiming to answer

1. Can a machine learning model help evaluate whether a mushroom is poisonous or edible?
2. Which features are most indicative of a poisonous mushroom?
3. Which habitat contains the highest percentage of edible mushrooms?
4. What populations contain the most edible and most poisonous mushrooms?

Data Exploration Phase

- Reviewing features like:
  - Cap shape
  - Bruises
  - Odor
  - Gill size

- Determining appropriate code best suited for dataset

Technologies Used

- Python/Jupyter Notebook
  - Libraries include pandas, sqlite3, sklearn
  - Random Forest Classifier
- Tableau
- Google Slides
- Git

```
# Import machine learning and other dependencies

import sqlite3
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix, accuracy_score, classification_report
```

Building the Database

- Sqlite3 and pandas
- Features - mushroom characteristics
- Target - edible or poisonous?
- Use f string syntax
- Separate notebook was key

```
# Create features table

cur.execute(create_features)
```

```
# Separate dataframe into features and target dataframes

df_features = df.drop(['class'], axis=1)
df_target = df[['class']]
```

```
# Create string to use when creating features table

sql_cols = 'create table mushroom_features (id number, '
for col in df_features.columns:
   sql_cols += col + ' varchar,\n'

create_features = f"{sql_cols[:-2]})"
```

```
# Check string created above

print(create_features)
```

```
create table mushroom_features (id number, cap_shape varchar,
cap_surface varchar,
cap_color varchar,
bruises varchar,
odor varchar,
```

Analysis and Data Prep

- SQL query to get both tables
- Join tables to create mushroom_df
- Encode data using LabelEncoder
- Define features and target (mushroom characteristics and edibility)

	id	class	id	cap_shape	cap_surface	cap_color	bruises	odor
0	0	p	0	x	s	n	t	p
1	1	e	1	x	s	y	t	a
2	2	e	2	b	s	w	t	l
3	3	p	3	x	y	w	t	p
4	4	e	4	x	s	g	f	n

	id	class	id	cap_shape	cap_surface	cap_color	bruises	odor
0	0	1	0	5	2	4	1	6
1	1	0	1	5	2	9	1	0
2	2	0	2	0	2	8	1	3
3	3	1	3	5	3	8	1	6
4	4	0	4	5	2	3	0	5

Overview of Machine Learning Model

- Data was split into training and testing samples using “train_test_split” from sklearn
- Scaled the data using StandardScaler
- Create and train Random Forest Classifier model
- Make predictions based on the dataset
- Determine accuracy of the model

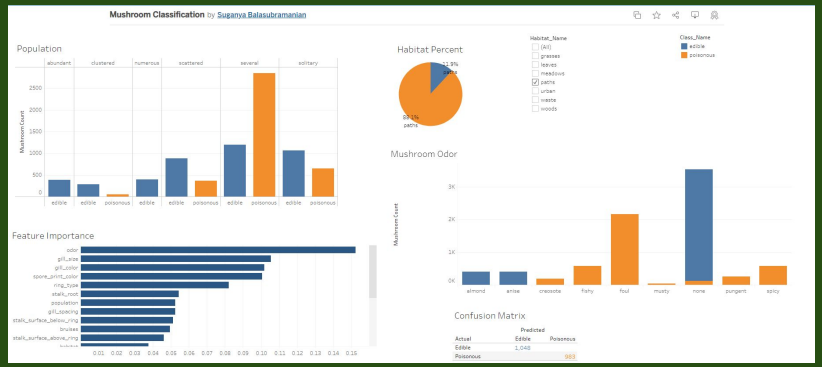
```
# Create a random forest classifier

rf_model = RandomForestClassifier(n_estimators=128, random_state=78)
```

```
# Fit the model

rf_model = rf_model.fit(X_train, y_train)
```

```
# Make predictions using the testing data

predictions = rf_model.predict(X_test)
```

Dashboard

- Our dashboard displays the results of our analysis of mushroom features
- Using Tableau, we illustrate the features that best indicate an edible v. poisonous mushroom
- Interactive elements will include:
  - Chart to display results
  - Graph that displays edible v. poisonous elements



Results

We visualized our dataset and the results of our learning model using a Tableau Dashboard

- Question 1: Confusion matrix
- Question 2: Bar graph of most important features
- Question 3: Pie chart of edible/poisonous results with filter for habitat
- Question 4: Bar graph of edible/poisonous results with filter for population

Accuracy Score

- **Accuracy Score** assesses the performance of our Random Forest Classifier
  - Percentage of correct predictions made by our model

$$Accuracy = \frac{TrueNegatives + TruePositive}{TruePositive + FalsePositive + TrueNegative + FalseNegative}$$

- **Results:** Based on our analysis, there is a high accuracy score of over 90% (Accuracy Score: 1.00)
  - Features are great predictors of our classes!

Precision and Recall

- In a model with discrete outcomes, a **Confusion Matrix** evaluates precision and sensitivity/recall.
- **Results:** extremely precise and extremely sensitive.
  - **Low F1 Scores** supports this analysis.

Confusion Matrix

	Predicted edible	Predicted poisonous
Actual edible	1048	0
Actual poisonous	0	983

Accuracy Score : 1.0

Classification Report

	precision	recall	f1-score	support
0	1.00	1.00	1.00	1048
1	1.00	1.00	1.00	983
accuracy			1.00	2031
macro avg	1.00	1.00	1.00	2031
weighted avg	1.00	1.00	1.00	2031

Challenges/Lessons

- Coping with branches and merge conflicts in GitHub
  - Push and pull; don't ignore .gitignore.
- Trusting results of our machine learning model
  - First time's a charm ?
- Achieving the visualizations we wanted in Tableau
  - Trial and error!

Future Analysis

- Notes on suspiciously high accuracy:
  - Previous examples on Kaggle and Youtube
  - Pros/cons of one hot encoding vs. label encoding
  - Overfitting: May need to run model with more mushroom data
- Using another MLM or Neural Network to test Random Forest results
  - Best model to use?