CS505 – Final Project

Binary Classification of Mushrooms

Wiley Hartzog, Sean Durbin, Colton Starkey, Kyle Schwartz, Thabo Adams

Introduction - background

- Mushrooms are an extremely common fungus with over 14,000 unique species across the globe.
- While mushrooms are a very common food, that much biodiversity there are many that are poisonous as well.
- Much of the illness caused by mushrooms is from those picked in the wild that people might not be educated on.
- This leads us to the project…

Introduction – task at hand

- Purpose of project is to classify mushrooms as either poisonous or edible given datasets
  - Binary classification
  - Done using machine learning models
  - Datasets provided via .csv files.

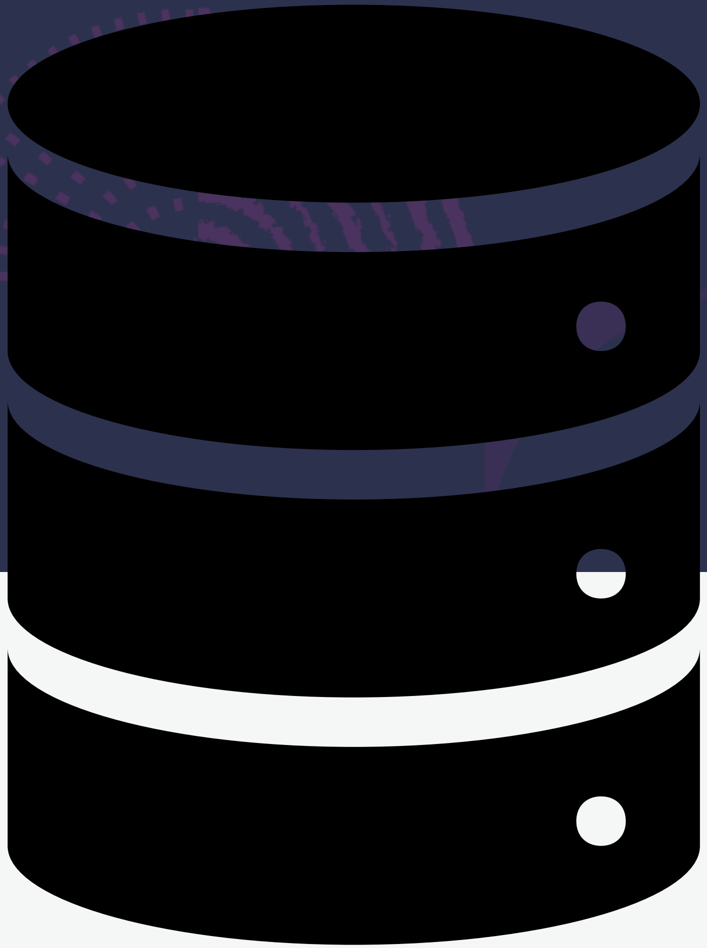
Introduction – business context

- Beneficial to someone that is getting into picking wild mushrooms or just wants some general guidance on what to look for in certain types of mushrooms.
- for liability purposes since this is not 100% accurate would not want someone to put their own live in the hands of our model and fully trust it.
    - Still could be used in research and education.

Dataset Overview

- Sourced from Kaggle
  - Contains over 1,000,000 rows of data
  - 20+ columns of attributes
- All the data in each column besides id of each mushroom was stored as either a float or character/string, or boolean.
- Characters like “a” for autumn and “w” for winter were used for the season attribute as an example

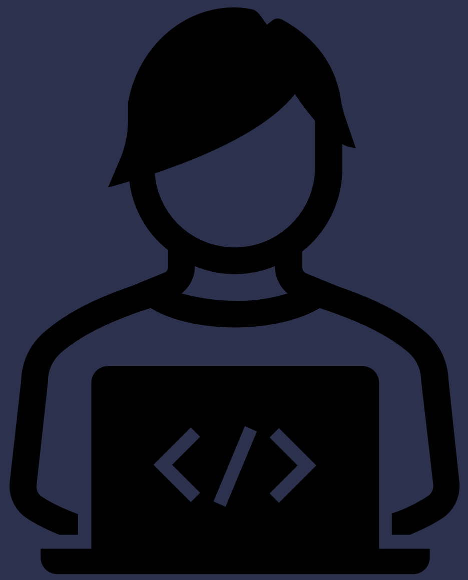
Displaying Data Files



Data Preprocessing

- The data could not be simply uses as provided and get accurate results.
- First began by using pandas to pull in data from the files we downloaded locally.
- Category columns were identified, and data was mapped in and encoded as integer values.
- Data was split into training and test but also dropped the less important columns and normalized
- Null attributes removed via imputation.

Data Preprocessing - code explanation



Model Development - XGBoost

- Model we selected in the proposal document.
- Hypothesized to be the most accurate.
- XGBoost algorithms automatically preprocess the data internally after it transitioned to a DMatrix object.
- Utilizes gradient boosting to minimize loss function, and builds decision trees as base models.
- A moderately complex model with the max depth of each tree being set to 6.
- Uses a moderate learning rate of 0.1.
- Optimized for binary classification.
- Uses an AUC metric to access model performance.
- Set to 100 boosting rounds.
- Automatically stops training to prevent overfitting if validation metric doesn't improve after 10 rounds.

Model Development - XGBoost

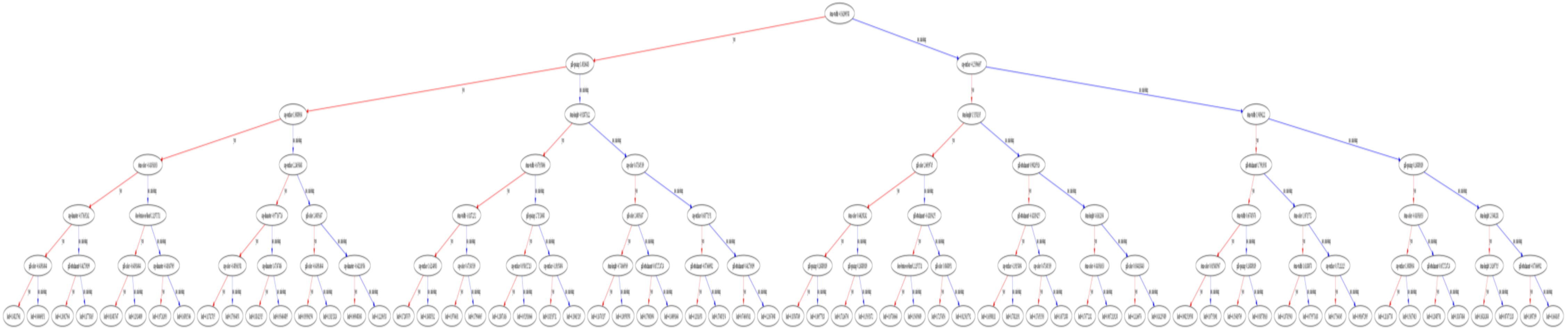
```python
# Prepare data for XGBoost
dtrain = xgb.DMatrix(X_train, label=Y_train)
dval = xgb.DMatrix(X_val, label=Y_val)
dtest = xgb.DMatrix(X_test, label=Y_test)

# Define XGBoost model
params = {
    'max_depth': 6,
    'eta': 0.1,
    'objective': 'binary:logistic',
    'eval_metric': 'auc'
}

# Train the XGBoost model
print("Training XGBoost model...")
bst = xgb.train(
    params=params,
    dtrain=dtrain,
    num_boost_round=100,
    evals=[(dtrain, 'train'), (dval, 'validation')],
    early_stopping_rounds=10,
    verbose_eval=True
)

print("\nXGBoost model training complete.")
```

Model Development – XGBoost – tree diagram



Model Development - logistic regression

- Second model type chosen for experiments.
- Less hyperparameters to define compared to XGBoost.
- Data preprocessing such as handling missing values and mapping non-number values to an integer value was required for this model.
- Makes predictions based a probabilistic framework.
- The model was set to train for 1000 iterations.

Model Development – logistic regression

```
print("Training Logistic Regression model...")
lr = LogisticRegression(max_iter=1000)
lr.fit(X_train, Y_train)

# Confirm training completion
print("\nLogistic Regression model training complete.")
```

Model Development - random forest classifier

- Third and final model type chosen for experiments.
- Only used one hyperparameter.
- Data preprocessing such as handling missing values and mapping non-number values to an integer value was required for this model.
- Data is split based on features at each node.
- Combines the predictions from multiple trees to improve performance and reduce overfitting.
- Set to create 100 decision trees.

Model Development - random forest classifier

```
print("Training Random Forest model...")
print("This will take about 5.5 minutes.")
rf = RandomForestClassifier(n_estimators=100)  # Set the number of trees
rf.fit(X_train, Y_train)

print("\nRandom Forest model training complete.")
```

Performance Evaluation - metrics

- Multiple metrics used to gauge performance:
  - Accuracy: Percentage of total predictions made that were correct.
  - Precision: measured by true positives / (true positives + false positives)
  - Recall: measures by true positives / (true positives + false negatives)
  - F1 Score: used in binary classification and is derived from both precision and recall. Generally considered a better metric than accuracy.
  - ROC-AUC: stands for receiver operator characteristic area under curve. Represents the probability of a model given randomly picked positive and negative examples that it will rank positive higher than negative

Performance Evaluation - Results

- Best performance is Random Forest Classifier
- Worst performance is logistic regression

Test Results Comparison with Detailed Metrics:

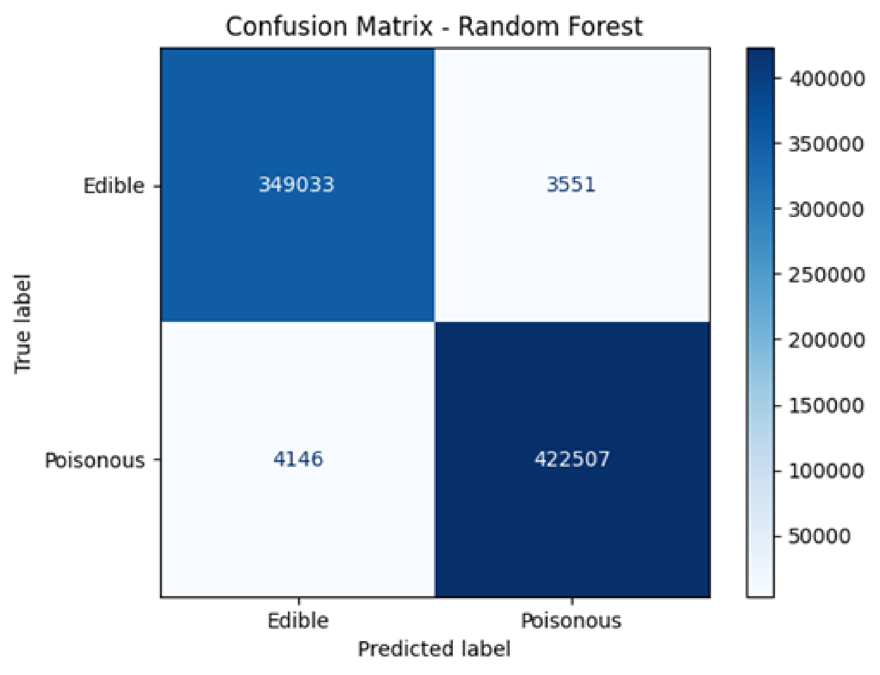
	Model	Accuracy	Precision	Recall	F1 Score	ROC-AUC
0	XGBoost	0.981167	0.98	0.98	0.98	0.994639
1	Logistic Regression	0.630299	0.63	0.63	0.63	0.683139
2	Random Forest	0.990122	0.99	0.99	0.99	0.995617

Confusion Matrix

- Random forest model confusion matrix
  - The top left is true positive.
  - Top right is false negative.
  - Bottom left is false negative.
  - Bottom right is true negative.
  - Accurately predicted 349,033 true edible
  - Accurately predicted 422,507 true poisonous
  - Formula to calculate accuracy:

    Accuracy = (TP + TN) / (TP + TN + FP + FN)

    Accuracy = (349033 + 422507) / (349033 + 422507 + 4146 + 3551)
  - The matrix depicts a 99.5% accuracy.

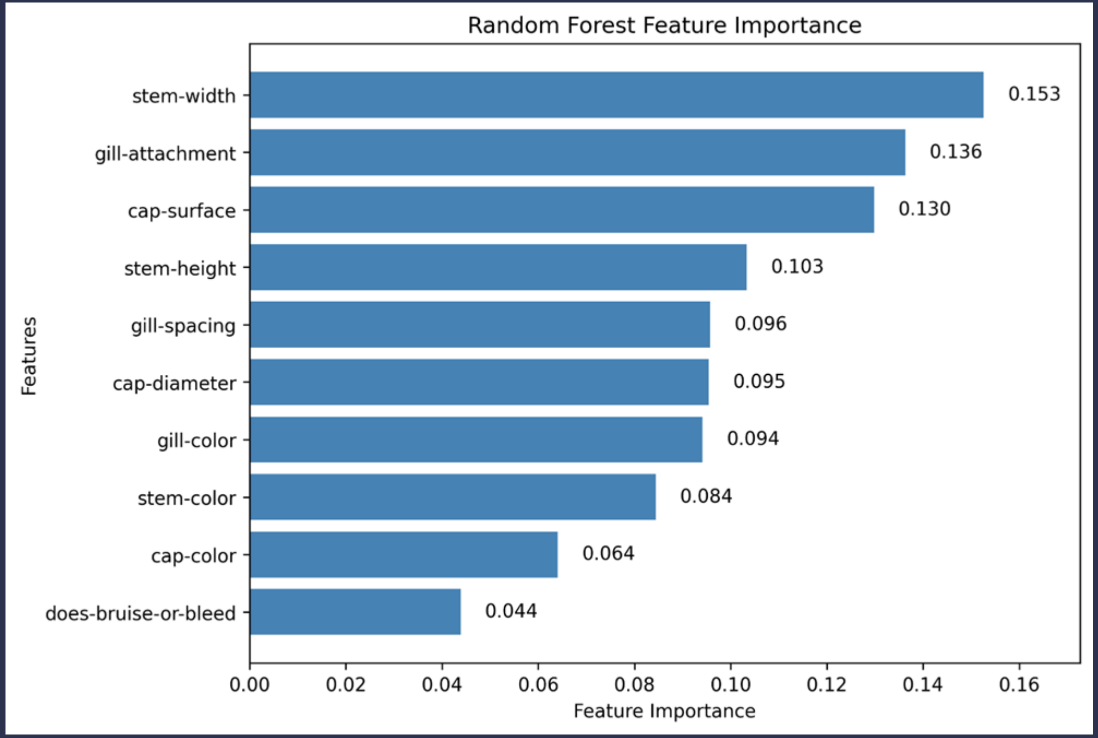


Feature Importance Analysis

- **Most Influential Features**
  - Stem Width
  - Gill Attachment
  - Cap Surface

- **How Features Were Analyzed**
  - Random Forest ranks features based on their impact on decisions.
  - Importance derived from decision tree splits.

- **Practical Insights**
  - Highlights traits that separate poisonous mushrooms from edible ones.

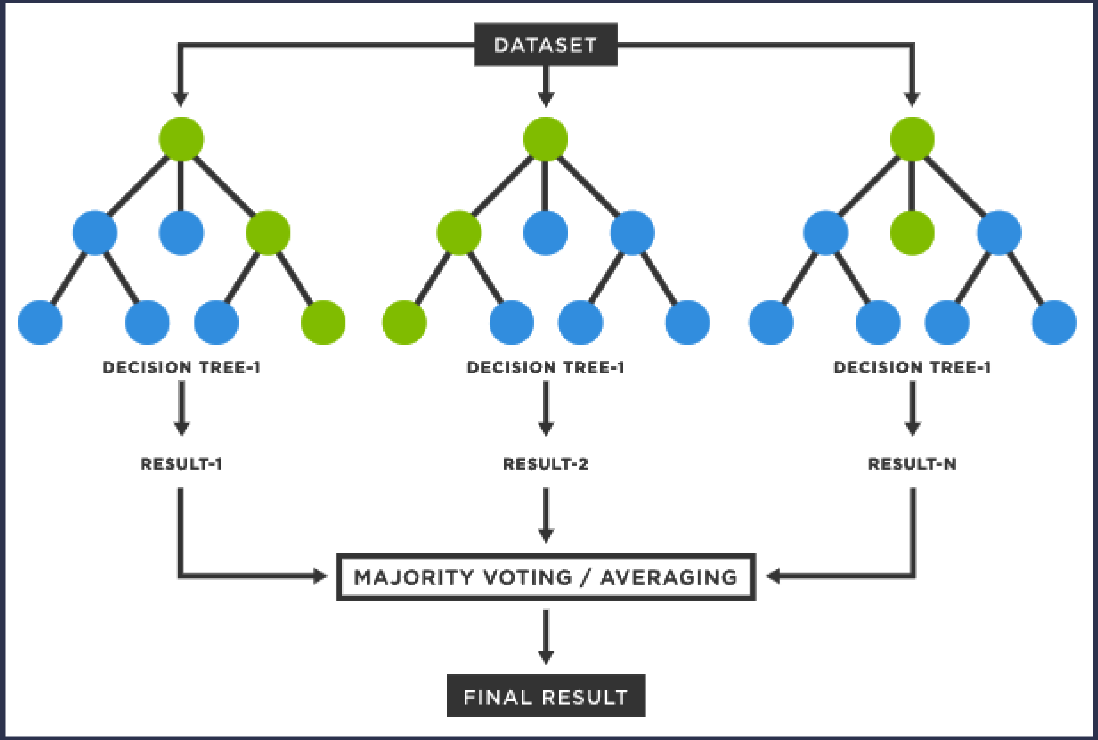


Model Interpretation

- **How Random Forest Works**
  - Combines predictions from multiple decision trees.
  - Each tree trains on random subsets of data and features.
  - Uses majority voting or averaging for final classification.

- **Decision-Making Process**
  - Each tree predicts based on individual thresholds (e.g., stem width, gill attachment).
  - Aggregates predictions from all trees for a final result.

- **Advantages**
  - Robust against overfitting due to randomization.
  - Transparent and interpretable with features importance metrics.



Problem Solving Steps

- **Key Issue**:
  - Model misclassified 4,146 poisonous mushrooms as edible (<1%).
- **How We Can Improve Accuracy**:
  - **Collaborate with experts**: Validate data and identify overlooked patterns.
  - **Add new data points**: Use environmental factors like soil type or region.
- **Safety First**:
  - Focus on reducing false negatives to prioritize user safety, even if it slightly increases false positives.

Projected Business Outcomes

- **Improved Safety**:
  - Builds trust and prevents dangerous misclassifications.
- **Stronger Customer Retention**:
  - Reliable results keep users engaged and reduce churn.
- **Educational Impact**:
  - Helps educate users and communities on safe mushroom identification.
- **Competitive Advantage**:
  - Focus on safety and reliability differentiates the tool from competitors.
- **Future Growth**:
  - Opportunity to partner with educators, conservation groups, and others to expand the tool's reach.

Limitations

- Our model was limited by the species of mushrooms included in the UCI dataset
- Each characteristic is prone to the interpretation of whoever entered the feature data
- This tool <u>must</u> be cross-referenced by somebody with mushroom identification skills to be used safely
    - This could have legal consequences that must be considered
    - This also makes the tool less useful for the average user in its current form

Impacts

- Not every mushroom is described in the dataset and therefore cannot be assumed to be accurately classified by our model
- Tabular data, although great for training data with our model, is limited in usefulness. Each entry must be classified and the data must be manually entered for each key feature

Future Work

- The key to future works is image analysis:
  - This would greatly expand the generalizability of the model
  - Beyond binary classification, image analysis could open the door for species identification as well
  - A hybrid approach of tabular and image analysis could prove to be a more complete method
  - Image analysis could allow the development of web and mobile applications devoted to mushroom identification
- Incorporating image data would require vastly more complicated models, and it would likely sacrifice efficiency and a slight amount of accuracy.

Improvements

- The first and most impactful improvement would be better data and more of it.
  - Incorporating more species of mushrooms and more features would create a more generalizable model
  - This additional data could be crowdsourced from mycology groups willing to share expert identification data
- More experimentation of hybrid model approaches could allow improvement either in efficiency or accuracy, although our results are quite high in both.

Conclusion

- **Best Performing Model**
  - Random forest achieved 99% accuracy and was cross-validated.
- **Insights from Feature Importance**
  - Stem width, gill attachment, and cap surface were the most influential traits in determining toxicity.
- **Limitations**
  - A small percentage of false negatives (poisonous classified as edible).
  - Requires detailed feature input which limits it's use for non-technical users.
- **Future Potential**
  - Integration of image analysis for broader accessibility.
  - Relevant applications in education, food safety, and research.

References

- Colorado State University. (2024, June 19). *Mushrooms*. Food Source Information. https://www.chhs.colostate.edu/fsi/food-articles/produce/mushrooms/#:~:text=There%20are%2C%20however%2C%20many%20morphological,on%20some%20type%20of%20substrate
- GeeksforGeeks. (2024b, June 20). *Logistic Regression in Machine Learning*. GeeksforGeeks. https://www.geeksforgeeks.org/understanding-logistic-regression/
- GeeksforGeeks. (2024a, January 31). *Random Forest classifier using Scikit-learn*. GeeksforGeeks. https://www.geeksforgeeks.org/random-forest-classifier-using-scikit-learn/
- Google. (n.d.). *Classification: Roc and AUC | machine learning | google for developers*. Google. https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc#:~:text=The%20area%20under%20the%20ROC,random%20positive%20and%20negative%20example.
- Reade, W., & Chow, A. (2024). *Binary prediction of poisonous mushrooms*. Kaggle. https://kaggle.com/competitions/playground-series-s4e8