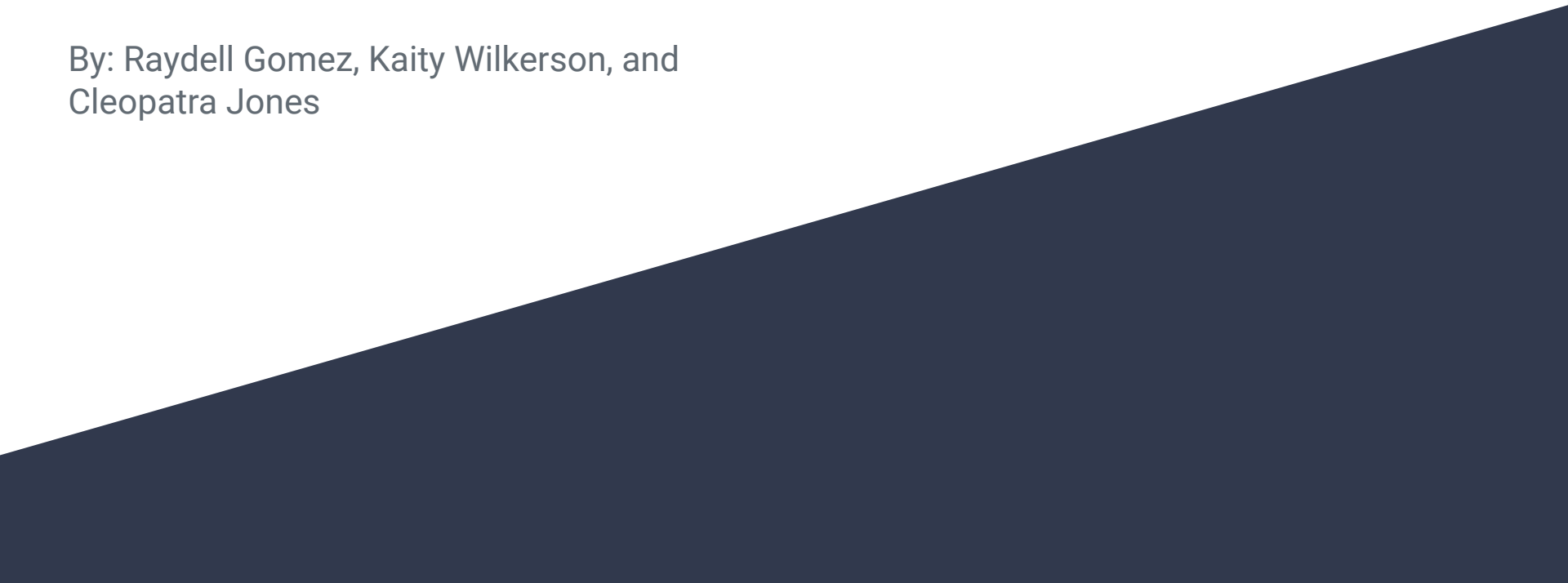


Predicting German Dog Breeds:

By: Raydell Gomez, Kaity Wilkerson, and
Cleopatra Jones

A dark blue diagonal gradient bar that starts from the bottom left corner and extends towards the top right corner, covering the lower half of the slide.

Overview

Summary:

This project aims to predict 5 German dog breeds by using the average height, weight, and lifespan of the following dogs; German Longhaired Pointer, German Pinscher, German Shepherd, German Shorthaired Pointer, and German Wirehaired Pointer.

The approach:

We found a Dog Breed csv file that listed every dog breed along with multiple columns of data for each breed. We decided to take only the German breed and create a smaller csv file using only the average height, weight and lifespan.

https://docs.google.com/spreadsheets/d/1ZxRr8xbqVMIOzJztBSHwLayaSR0C-ji_kRIgEF-pPW0/edit#gid=1236799813

From this data, we used the Random Forest Classifier from SKLearn to model, fit, and predict the data.

A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. Trees in the forest use the best split strategy, i.e. equivalent to passing `splitter="best"` to the underlying `DecisionTreeRegressor`. The sub-sample size is controlled with the `max_samples` parameter if `bootstrap=True` (default), otherwise the whole dataset is used to build each tree.

The Results:

Random Forest Classifier

```
X = breed_df.drop('Breed', axis=1)
Y = breed_df['Breed']

clf = RandomForestClassifier()
clf.fit(X, Y)

y_pred = clf.predict(X)

pred = "Accuracy of the Model:", metrics.accuracy_score(Y, y_pred)
from sklearn import metrics
print(pred)

RandomForestClassifier()
('Accuracy of the Model:', 1.0)
```

Components of the Streamlit App

We then created a Streamlit app to visualize the outcome of that data: <http://localhost:8501/> . We used a couple of components to personalize the front end of our Streamlit app to make it more visually appealing.

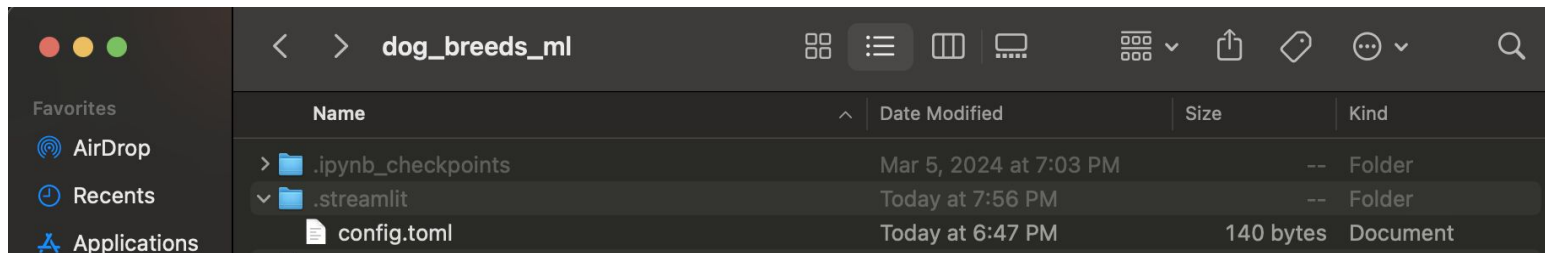
- 1.) From the `streamlit_extras` library, we utilized a colored header (for extra flair!)
- 2.) We also changed the background color to a nice, cool blue hue.

Components of the Streamlit App Con't

Below is a snippet showing the simple code we used for the header. You also need to pip install streamlit-extras

```
51
52 #This line of code adds a header, divided with a rainbow line (because everyone loves a pop of color!)
53 st.header('This is app will predict the breed of a dog!', divider = 'rainbow',)
```

In order to change the background color, we needed to create a toml file. The toml file needs to be saved to an invisible file, where it will then be called as an environment file. This means the file name needs to be “.streamlit” and your toml file will be named “config.toml.” The screenshots below show how to set it up in your project file.



Components of the Streamlit App Con't

Below is a snapshot showing the contents of the toml file. HTML color hash codes are used to determine the colors you choose for your app.

Users > kaitywilkerson > Desktop > Project_3 > dog_breeds_ml > .streamlit >  config.toml

```
1  [[theme]]
2  primaryColor = "#f63366"
3  backgroundColor = "#327ba8"
4  secondaryBackgroundColor = "#dde5eb"
5  textColor = "#edf4f5"
6  font = "sans serif"
7
```

The Conclusion:

In the end, our Streamlit application deployed beautifully. We even added an option for users to add a CSV file of their choosing to make their own predictions.

×

Upload you input CSV file

Drag and drop file here
Limit 200MB per file • CSV

Browse files

average height

18.5023.50

average weight

35.1067.50

average lifespan

1113

Deploy

This is app will predict the breed of a dog!

User Input features


Awaiting CSV file to be uploaded. Currently using example input parameters

	average height	average weight	average lifespan	Unnamed: 0
0	18.5	35.1	11	None

Breed Prediction

value

German Pinscher



New Technology:

1. TOML file, which was used to customize our Streamlit Interface(background and header).
2. Image.open; PIL is the Python Imaging Library which provides the python interpreter with image editing capabilities. The Image Module provides a class with the same name which is used to represent a PIL image. The module also provides a number of factory functions, including functions to load images from files, and to create new images.**PIL.Image.open()** Opens and identifies the given image file.

```
img0 = Image.open('DogImages/German-Longhaired-Pointer.jpeg')
img1 = Image.open('DogImages/German-Longhaired-Pointer.jpeg')
img2 = Image.open('DogImages/German-Shepherd.jpeg')
img3 = Image.open('DogImages/German-Shepherd.jpeg')
img4 = Image.open('DogImages/German-Wirehaired-Pointer.jpeg')
```

Challenges: Pickle

We initially wanted to use a Pickle file to file and load our German Dog breed, but we had issues running the file.

Pickle allows for flexibility when deserializing objects. You can easily save different variables into a Pickle file and load them back in a different Python session, recovering your data exactly the way it was without having to edit your code.

Advantages of Pickle:

- Unlike serialization formats like JSON, which cannot handle tuples and datetime objects, Pickle can serialize almost every commonly used built-in Python data type. It also retains the exact state of the object which JSON cannot do.
- Pickle is also a good choice when storing recursive structures since it only writes an object once.
- Pickle allows for flexibility when deserializing objects. You can easily save different variables into a Pickle file and load them back in a different Python session, recovering your data exactly the way it was without having to edit your code.

Disadvantages of Pickle:

- Pickle is unsafe because it can execute malicious Python callables to construct objects. When deserializing an object, Pickle cannot tell the difference between a malicious callable and a non-malicious one. Due to this, users can end up executing arbitrary code during deserialization.
- As mentioned previously, Pickle is a Python-specific module, and you may struggle to deserialize pickled objects when using a different language.
- According to multiple benchmarks, Pickle appears to be slower and produces larger serialized values than formats such as JSON and ApacheThrift.

Questions/Concerns:

As mentioned prior, we initially started off attempting to use the Pickle file. This took up about 3 days. We realized that we couldn't use the Pickle method, and switched to a completely different project. Eventually we figured out a way to make our initial German Dog breed project work, without the Pickle file.

With more time, we would have utilized more Machine Learning capabilities to add different dog breeds to the Streamlit App.

We also realized last minute that the "Upload your CSV file" did not function properly. With more time we would have worked allowing users to successfully upload their own CSV file.

Resources

<https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html>

<https://www.geeksforgeeks.org/python-pil-image-open-method/>

<https://www.datacamp.com/tutorial/pickle-python-tutorial#>