Delieverable 3
Feng (Shelley) Xia

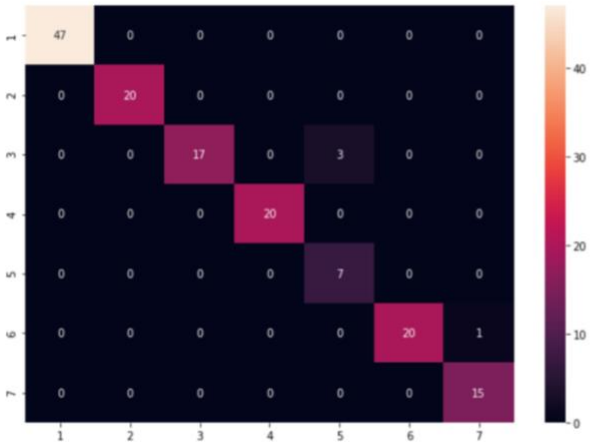
1. Final Training Results:

The training process along with its training results remain unchanged from the last deliverable, as the accuracy is desirable for a ML model, with little to no mistake in classification.

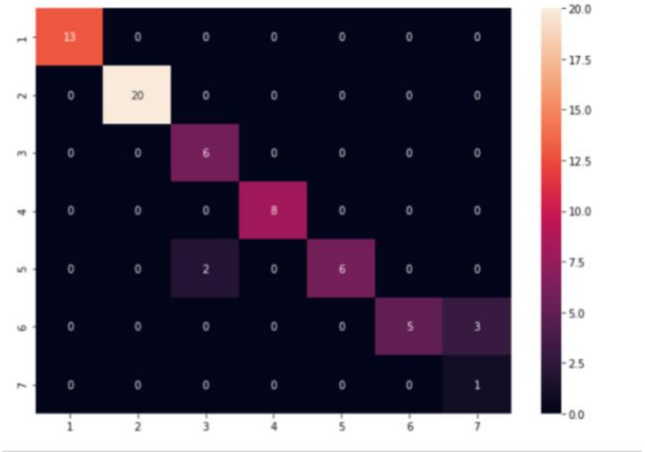
We would use the same results from delieverable 2 as our final results. Here are the accuracies and confusion matrices for train and test datasets respectively, copied and pasted from delieverable 2.

```
print(rdf.score(X_train,y_train))
print(rdf.score(X_test,y_test))
###
```

```
0.9733333333333334
0.921875
```



Train



Test

2. Final Demonstration Proposal

We first import the joblib library which is commonly used to save and load a scikit-learn model. We will save our 'rdf' model for future use, so that we don't have to train it again per user query.

The idea of the final product is a website that asks a user a series of questions about an animal in mind, such as "Does it have hair?", "Does it lay eggs?", each question corresponding to a feature in our model. We expect the user to answer every question, and if they are not sure about a specific question, they will be asked to provide an arbitrary answer. This can ensure that all of our features are not empty so that we could make predictions. We suspect that asking a series of questions may be a challenging job, because it would require some long and complex code in backend files. If this does not go well, we may consider asking the user to provide all the features at once by completing a 'sheet'.

Note that the website inquires the user a question at a time, and only when it receives some answer in the correct format, such as 'yes' or 'no', it would continue to ask the next question until all questions have been asked. The fact that we only accept formatted answers puts on a restriction that may negatively affect user experience, thus this is something we may look into after successfully creating such a website. For example, if a user enters 'I don't know', we may randomly choose an answer in the backend to fill up the blank in that feature.

For the backend implementation, we wish to use Flask as our framework. And if time permits, we would like to explore React for the frontend, to improve user friendliness.

Speaking of real-world application, this website is suitable in various settings: zoos, kindergartens, biology classes in school, museums, etc. It is mainly designed to be used for educational purposes.

With no prior experience in website development, I would like to start learning Flask by viewing the official doc of Flask.

https://flask.palletsprojects.com/en/1.1.x/

Next, I found the below online sources on Flask helptul.

https://blog.miguelgrinberg.com/post/the-flask-mega-tutorial-part-i-hello-world