# Machine Learning Nano Degree
# Capstone Report
# Predicting Zoo's animals class
Khloud Ali
July 6, 2018

# I. Definition

## 1.1 Project Overview

**Domain Background**

Taxonomy is a branch of science that includes the identification, nomenclature, description and classification of the organisms.

classification is the science of how living organisms are grouped together.

It started with Aristotle who developed the first classification system, which divided all known organisms into two groups then dividing each of these main groups into three smaller subgroups:

**1-Animals**

**Subgroups:** Land, Water, Air.

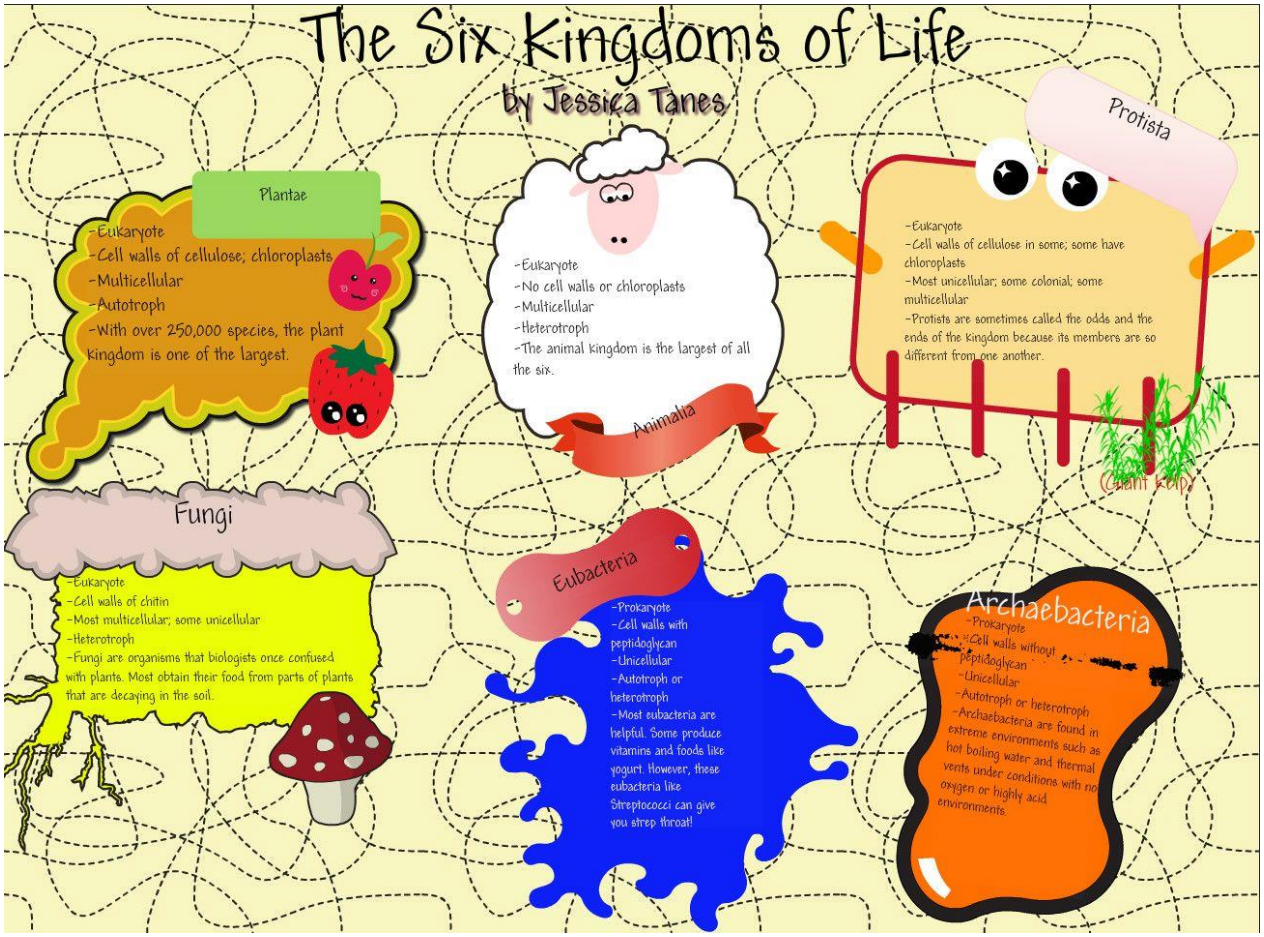
**2-Plants**

**Subgroups**: Small, Medium, Large.

Then came Linnaeus, who classified the living organisms according to its traits and called this groups "Kingdoms",and also Linnaeus hasdividedeach kingdom into five levels: class, order, genus, species, and variety. Each living organism was placed in those levels based on its traits, including similarities of physical body parts such as size, shape, and the way they feed.

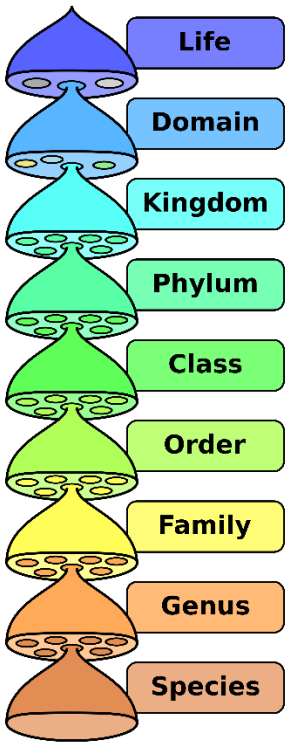
But with the use of microscope leading to the discovery of new organisms the today classification system has developed to include 5 kingdoms which are:

1-Plantae.
2-Animalia.
3-Fungi.
4-Protista.
5-Monera (includes Eubacteria and Archaebacteria).




But in this domain we're going to focus on the Animals kingdom "word animal derived from a latin word called animalis meaning …Having Breath…"where the animalia "kingdom" is divided into 40 smaller groups each is known as a "Phylum" which is divided into further groups each is known as a "Class" and animals in a class have so

much more in common more than the whole entire phylum, then class is divided into further more groups each is known as an “order” and in each order there’s different types of families with similar features that’s further more divided into “genus” where each genus contain the animals with very similar features that’s closely related.




| Levels (from the highest to the lowest) | Example |
|---|---|
| Kingdom | Kingdom Animalia is the broadest category of all in the animal classification system. It includes every animal. |
| Phylum (plural: Phyla) | Phylum Chordata includes all animals of the Kingdom Animalia that have spinal cords. |
| Class | Class Mammalia includes all warm-blooded animals of the Phylum Chordata that have hair and feed their young with milk. |
| Order | Order Artiodactyla includes all animals of the Class Mammalia that have an even number of toes in their hooves. |
| Family | Family Giraffidae includes all animals of the Order Artiodactyla that have long legs, a long narrow head with small horns, thin lips, and long tongues. |
| Genus (plural: Genera) | Genus Okapia and Genus Giraffa |
| Species | Species Camelopardalis, also known as giraffes in English. |

focusing onthe famous sevenclasses  we’re going to classify each zoo animal in our dataset according to the classes they belong towhich are :

**1-Mammels**

If the animal’s body is covered with hair and it drinks milk when it’s a baby then it belongs to the mammal’s class.

**2-Birds**

When the animal has feather and lays eggs as it’s born out of an egg then it belongs to the bird’s class.

**3-Fish**

If the animal is vertebrate that lives in the water and have fins, scales and gills on their body then it belongs to the fishes’ class.

**4-Reptiles**

If the animal has a scaly skin, born in land with tail and are cold blooded then it belongs to the reptile's class.

**5-Amphibians**

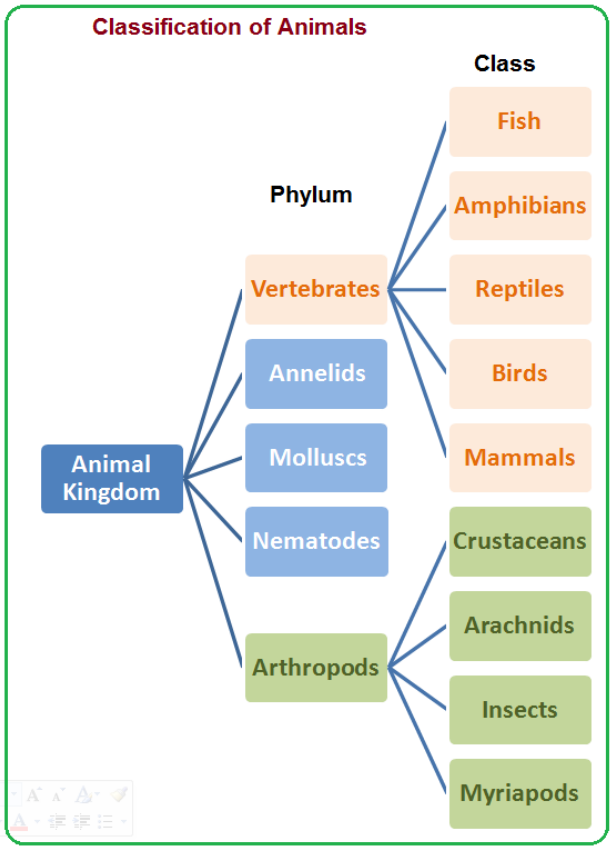
If the animal is born in water and it breaths through gills like fishes, but they develop lungs as they grow and can live on land then it belongs to the amphibian's class.

**6-Arthropods "Bugs"**

If the animal has more than four jointed legs and it doesn't have a backbonethen it belongs to the arthropod's class.

**7- Invertebrates**

If an animal doesn't have a backbone or vertebra then it belongs to the invertebrates' class.



## 1.2 Problem Statement

- *Is the problem statement clearly defined? Will the reader understand what you are expecting to solve?*
- *Have you thoroughly discussed how you will attempt to solve the problem?*
- *Is an anticipated solution clearly defined? Will the reader understand what results you are looking for?*

It's a multi-class classification problem, where the problem is finding to which animal kingdom class each zoo animal belongs.

The goal is to predict which zoo animal is from a certain class from the provided classes, seven target class types are provided in this dataset:
Mammal, Bird, Reptile, Fish, Amphibian, Bug and Invertebrate

**Solution Statement**

Machine learning algorithm can learn from given data and make classification with a good accuracy giving a trusted output.

I am going to make a prediction using different models that either linear model, non-linear models or even ensemble models

then i am going to feed the model with the animal's name and its features making it predict to which class this given animal belongs to and the model with best score would be the final solution.

## 1.3 Metrics

The Zoo's animal class prediction is a classification problem therefore the optimal metric must be one that rates the model's ability to classify zoo animals across the 7 classes correctly.
The accuracy approach is simply measuring the percentage of animals that are correctly classified.
But since the dataset is imbalanced therefor the accuracy is no longer an option as it could be Misleading.
As a result, I'll be using the F1 Score, Also I will use auc-roc in chossing the best chosen model that will be used in the final result.
F1 score presents an optimal alternative that works well when the classes are imbalanced and it ensures that:

⇒ The animals selected of a certain class are relevant. (Precision)
⇒ relevant animals are selected in the same class. (Recall)

The formulas for Precision, Recall and F1 score where"TP" means True Positives,
"FP" means False Positives and "FN"means False Negatives.

$$\text{Precision} = \frac{tp}{tp + fp}$$

$$\text{Recall} = \frac{tp}{tp + fn}$$

| | | Actual | |
|---|---|---|---|
| | | Positive | Negative |
| Predicted | Positive | **True Positive** | **False Positive** |
| | Negative | **False Negative** | **True Negative** |

$$F_1 = \frac{2}{\frac{1}{\text{recall}} + \frac{1}{\text{precision}}} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}.$$

# II. Analysis

## 2.1 Data Exploration

### Datasets and Inputs

I used a dataset that features 7 different classes of animals and a 100 different zoo animal type with 16 features each is a Boolean provided by Kaggle…
https://www.kaggle.com/uciml/zoo-animal-classification#zoo.csv
All the features have a data type "Boolean".
The distinguishing features included in the dataset:

| No. | Feature | Description |
|---|---|---|
| 1. | Hair | Is its body is covered with hair? |
| 2. | Feather | Is its body is covered with feathers? |
| 3. | Eggs | Does it lay eggs or born out of an egg? |
| 4. | Milk | Does it drink milk when it's a baby? |
| 5. | Airborne | Does it fly? |
| 6. | Aquatic | Does it live in water? |
| 7. | Predator | Does it feed on row meat? |
| 8. | Toothed | Does it have teeth? |
| 9. | Backbone | Is it vertebra or invertebrate? |
| 10. | Breathes | Does it have a nose or breath on land? |
| 11. | Venomous | Is it poisonous? |
| 12. | Fins | Does it have fins? |
| 13. | Legs | Does it have legs and how many legs it has? |
| 14. | Tail | Does it have a tail? |
| 15. | Domestic | Is it housebroken? |
| 16. | CatSize | Is it in a cat size or not? |
| 17. | Class type | To which animal's class does it belong |

***statistics about the dataset***

I have used the print(x.shape) to display how many rows “animals” and how many columns “features” are there in the dataset, and the print(y.shape) to display  how many rows are there In the dataset.

```
In [7]: print(x.shape)
        print(y.shape)

        (101, 16)
        (101,)
```

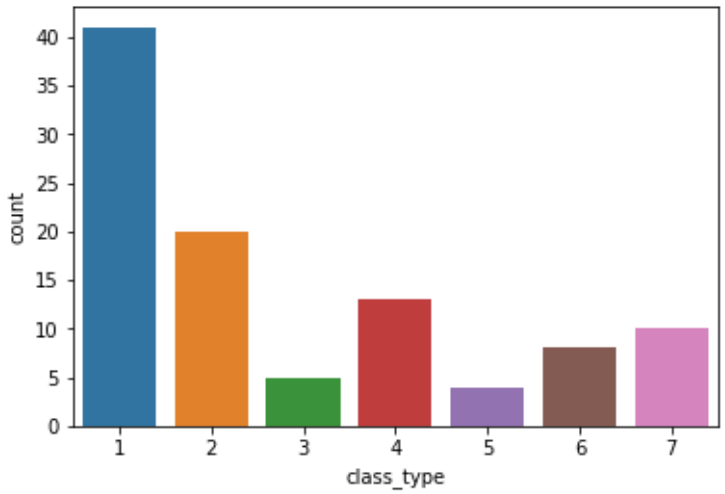
And then I’ve put the data into training and testing and then used the print(x_train.shape) To display number of rows and features used in the training set and the print(y_train.shape) To display number of rows and features in the testing set.

```
In [9]: x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.25,random_state=1)
        print(x_train.shape)
        print(x_test.shape)

        (75, 16)
        (26, 16)
```

then I used the sns.countplot(y) and y.value_counts() to display how many animal in each class then plot the results in a histogram.

```
In [16]: sns.countplot(y)
         y.value_counts()
```

```
Out[16]: 1    41
         2    20
         4    13
         7    10
         6     8
         3     5
         5     4
         Name: class_type, dtype: int64
```




Then I've have used the describe() function to count the number of rows for each feature and to calculate their Mean, standard deviation, the 3 quartiles and the minimum and maximum value for each feature.

```
In [67]: x.describe()
```

Out[67]:

| | hair | feathers | eggs | milk | airborne | aquatic | predator | toothed | backbone | breathes | venomous |
|---|---|---|---|---|---|---|---|---|---|---|---|
| **count** | 101.000000 | 101.000000 | 101.000000 | 101.000000 | 101.000000 | 101.000000 | 101.000000 | 101.000000 | 101.000000 | 101.000000 | 101.000000 |
| **mean** | 0.425743 | 0.198020 | 0.584158 | 0.405941 | 0.237624 | 0.356436 | 0.554455 | 0.603960 | 0.821782 | 0.792079 | 0.079208 |
| **std** | 0.496921 | 0.400495 | 0.495325 | 0.493522 | 0.427750 | 0.481335 | 0.499505 | 0.491512 | 0.384605 | 0.407844 | 0.271410 |
| **min** | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |
| **25%** | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 | 1.000000 | 1.000000 | 0.000000 |
| **50%** | 0.000000 | 0.000000 | 1.000000 | 0.000000 | 0.000000 | 0.000000 | 1.000000 | 1.000000 | 1.000000 | 1.000000 | 0.000000 |
| **75%** | 1.000000 | 0.000000 | 1.000000 | 1.000000 | 0.000000 | 1.000000 | 1.000000 | 1.000000 | 1.000000 | 1.000000 | 0.000000 |
| **max** | 1.000000 | 1.000000 | 1.000000 | 1.000000 | 1.000000 | 1.000000 | 1.000000 | 1.000000 | 1.000000 | 1.000000 | 1.000000 |

| fins | legs | tail | domestic | catsize |
|---|---|---|---|---|
| 101.000000 | 101.000000 | 101.000000 | 101.000000 | 101.000000 |
| 0.168317 | 2.841584 | 0.742574 | 0.128713 | 0.435644 |
| 0.376013 | 2.033385 | 0.439397 | 0.336552 | 0.498314 |
| 0.000000 | 0.000000 | 0.000000 | 0.000000 | 0.000000 |
| 0.000000 | 2.000000 | 0.000000 | 0.000000 | 0.000000 |
| 0.000000 | 4.000000 | 1.000000 | 0.000000 | 0.000000 |
| 0.000000 | 4.000000 | 1.000000 | 0.000000 | 1.000000 |
| 1.000000 | 8.000000 | 1.000000 | 1.000000 | 1.000000 |

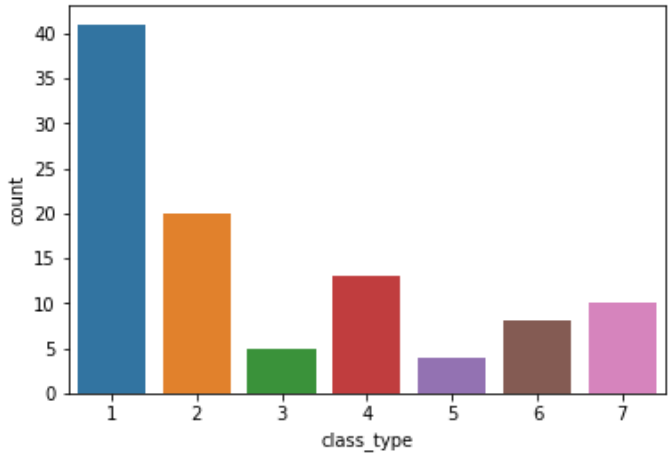
Then I used x.info() function to get information about the data set.

```
In [68]: x.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 101 entries, 0 to 100
Data columns (total 16 columns):
hair        101 non-null int64
feathers    101 non-null int64
eggs        101 non-null int64
milk        101 non-null int64
airborne    101 non-null int64
aquatic     101 non-null int64
predator    101 non-null int64
toothed     101 non-null int64
backbone    101 non-null int64
breathes    101 non-null int64
venomous    101 non-null int64
fins        101 non-null int64
legs        101 non-null int64
tail        101 non-null int64
domestic    101 non-null int64
catsize     101 non-null int64
dtypes: int64(16)
memory usage: 12.7 KB
```

## 2.2 Exploratory Visualization

I used the sns.countplot(y) and y.value_counts() to display how many animal in each class then plot the results in a histogram.

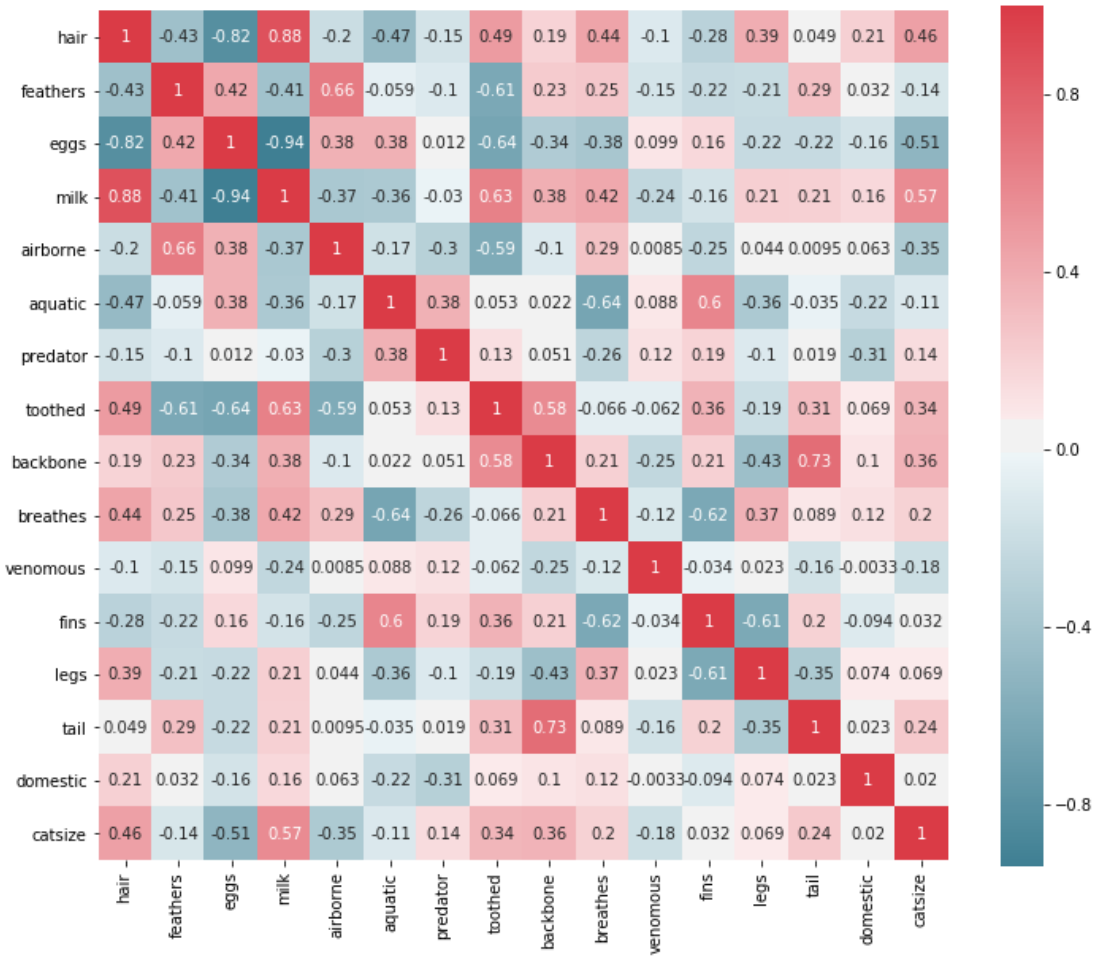
```
In [16]: sns.countplot(y)
         y.value_counts()

Out[16]: 1    41
         2    20
         4    13
         7    10
         6     8
         3     5
         5     4
         Name: class_type, dtype: int64
```




And then used the hmap=x.corr() to create a heat map to show the degree of correlation between the features of the dataset.

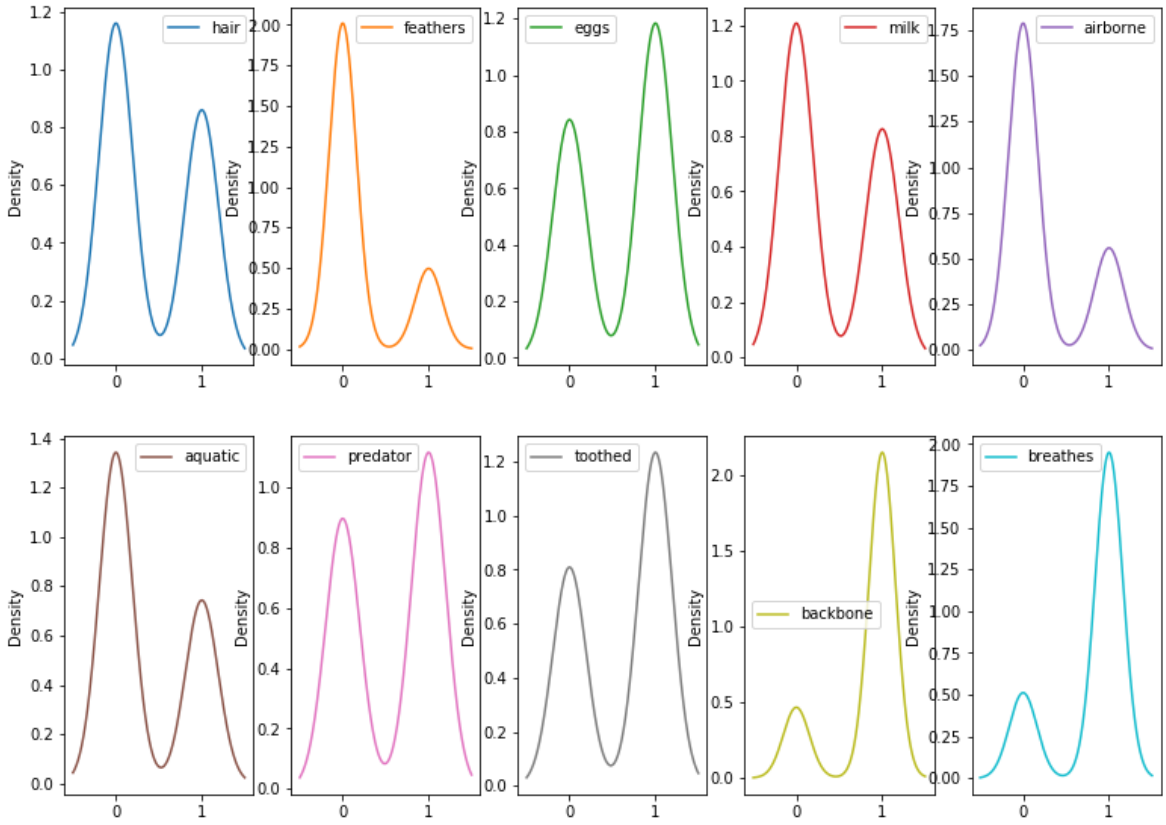
In [117]:
```
hmap=x.corr()
_,ax=plt.subplots(figsize=(12,10))
cmap=sns.diverging_palette(220,10,as_cmap=True)
sns.heatmap(hmap,cmap=cmap,ax=ax,square=True,annot=True)
```

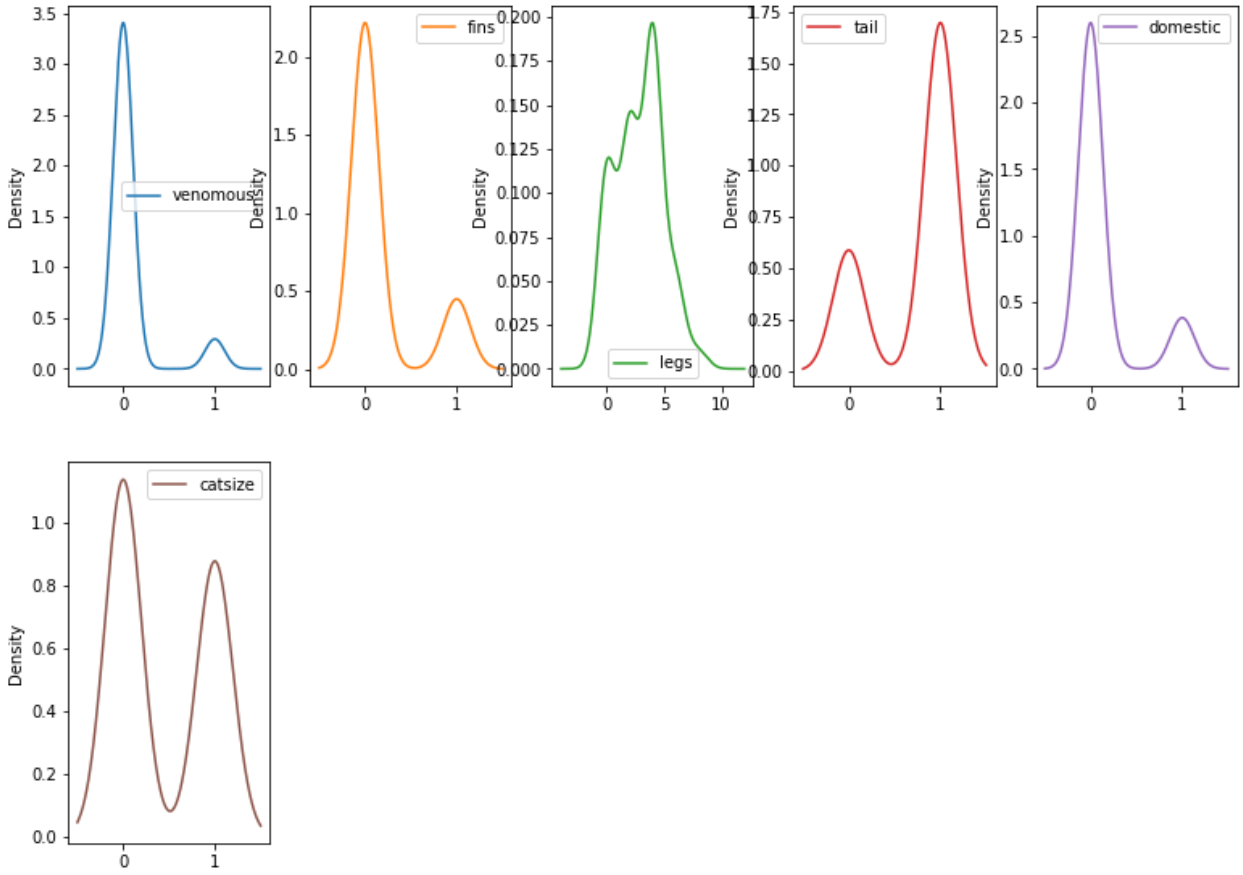
Out[117]: <matplotlib.axes._subplots.AxesSubplot at 0x1acc3ea3cc0>



Then used the plot() function to display the density of each value in each feature.

```
In [65]: x.plot(kind='density', subplots=True, layout=(4,5), figsize=(13,20), sharex=False, sharey=False)
         plt.show()
```




venomous
fins
legs
tail
domestic
catsize
Density

## 2.3 Algorithms and Techniques

I've used different models to make a prediction:

**"LR" LogisticRegression()**

Logistic Regression is a classification algorithm used in predicting the probability of a categorical dependent variable which is a binary variable containing data coded a Boolean predicting P(y=1) as a function of x.

**"KNN" KNeighborsClassifier()**

The model of the KNN is the whole training dataset, so in the case of predicting an unseen data Instance the KNN algorithm'll search the training dataset for the K-most similar instances, And in the case if regression problems the average of the predicted attributes is returned and For the classification problems the most relevant class is returned.

**"DT" DecisionTreeClassifier()**

It's an easy straight forward idea for solving the classification problem, it organizes a series of Test questions and conditions about the attributes of test record in a tree structure and in each time it receives an answer a follow up question is asked until a conclusion about the class label of the record is reached.

**"NB" GaussianNB()**

The Naïve Bayes is a classifier which uses the Bayes theorem where it predicts the probability for each class such as the probability for a given record belonging to a particular class.

**"SVM" SVC()**

The support vector classifier that's used for classification and regression problems, it fits the data provided and returns a best fit that categorizes the data, and see what is the predicted class.

**"RF" RandomForestClassifier()**

It's a classification algorithm that creates a number of decision trees, and the higher the number of trees given in the forest the higher the accuracy results are.

**“AB” AdaBoostClassifier()**

It’s a meta-estimator that begins with fitting the dataset to the classifier then fitting an additional copies of the classifier on the same dataset but with the adjusted weights of the Incorrectly classified instances so the classifiers can focus more on the difficult cases.

**“GBM” GradientBoostingClassifier()**

It’s a machine learning technique that train a group of weak learners in order to carry out a machine learning task (Classification/Regression), It depends on some techniques like boosting and gradient descent in order to reduce the bias and variance

And the accuracy each model scored is :

```
LR: 93.214%, 0.068
KNN: 81.429%, 0.083
DT: 93.393%, 0.066
NB: 94.464%, 0.068
SVM: 84.107%, 0.079
RF: 94.821%, 0.064
AB: 73.571%, 0.122
GBM: 93.393%, 0.066
```

So, I’ve chosen the model with the best score which is the RandomForestClassifier() with an Accuracy score of 94.8% .

## 2.4 Benchmark

I’ve used kaggle F1 scores that I found on the kernels (since there was no completion on that data so there’s no leaderboard), The highest F1 score I have seen till now on kaggle for this problem is 95% and I’ve reached an F1 score equals 96% using the random forest classifier.

# III. Methodology

## III. IImplementation

**Programming languageused:** python 3
**Libraries used:** Pandas, sklearn, seaborn, matplotlib, numpy.
I've start by

- Uploading the data and exploring it by presenting the data.
- Exploring the data much further by showing its shape and how the classes are distributed.
- Through data visualization, we could gain more insight from the data and figure out how we can deal with it.
- Train the models mentioned in the "Analysis" part above on training data and using cross-validation data to test various parameters tuning to choose the best.
- Test the models using test data then choosing the best model with the highest accuracy then applying the best model on the data to get the best score which turned out to be the Random Forest Classifier with an F1 score equal 96%.
- Finally Evaluating data through metrics and visuals.

## 3.2 Refinement

For the final model (RandomForestClassifier) I've tuned some parameters which appeared to increase the F1 score efficiently

| Parameter | Values | Description |
|---|---|---|
| n_estimators | From 25 to 75 with step of 2 | The total number of trees in the model |
| criterion | gini / entropy | The function used to measure how good the split is |
| max_depth | From 10 to 50 with step of 3 | The maximum depth of the tree |

By fine tuning the model the best mixture of parameter values was n_estimators of 25, max_depth of 10 and gini function, and with these values the F1 score increased from 94% to

over 96%.

# IV. Results

## 4.1 Model Evaluation and Validation

The final model I used Random Forest classifier as I stated before, I used K-fold cross-validation on the training data to make sure we had the actual F1 score of each model to choose the best from them, also to avoid any kind of overfitting especially the dataset is relatively small

The final F1 score was 96.2%

## 4.2 Justification

I've mentioned before in the benchmark my intention to use F1 score as the highest score It got on Kaggle as I've seen so far was 95% but my model has beaten that it by scoring 96% which is better and even much better than what I've expected it to be in predicting the right animal's class.

# V. Conclusion

## 4.1 Free-Form Visualization

I've visualized my data in many parts such as the heatmap and the density plotting parts and I've Included the images in the "Analysis" section above.

## 4.2 Reflection

The problem solution I used was making predictions using different models that either linear model, non-linear models or even ensemble models then feeding the model with the animal's name and its features making it predict to which class this given animal belongs to and the model with best score would be the final solution.

The really interesting aspects of this project were first the domain which reminded me how muchI used to love all branches of science but didn't get the chance to specialize in this field until now, when I got the chance to do something related to it even if it's on the surface, the second thing was my ability to write such a long report and proposal as i get bored so quickly, third thing actually was my friends' help, they really saved no effort helping me whenever I asked, they were great and so was my mentor Fernando, he was such a great help and always to be found around he's really great.

The difficult I found was a bit in coding and the forever writing of the report and proposal but Everything was solved with a bit more effort.

The finale model exceeded my expectations beating the highest score in Kaggle and I see that It could be used for solving such issues.

## 4.3 Improvement

To improve the score I could:

- Oversampling my data to increase the size of my dataset.
- Remove the outliers (if they existed) due to their disastrous effect especially with small dataset .
- Try other preprocessing techniques that could decrease the variance and avoid overfitting.

## References


https://a-z-animals.com/reference/animal-classification/
http://www.kidzone.ws/animals/animal_classes.htm
https://museumsvictoria.com.au/bugs/aboutbugs/types.aspx
https://biology.tutorvista.com/organism/kingdom-animalia.html
http://scienceprojectideasforkids.com/2010/history-of-classification/
http://www.softschools.com/science/biology/classification_of_living_things/
http://www.softschools.com/science/biology/classification_of_living_things/
http://www.softschools.com/science/biology/the_five_kingdoms/
https://www.mrsd.org/cms/lib/NH01912397/Centricity/Domain/245/animal%20classification%20

https://github.com/udacity/machine-learning/blob/master/projects/capstone/capstone_proposal_template.md
https://github.com/davidrobles/mlnd-capstone-proposal/blob/master/proposal.pdf
https://github.com/Tahsin-Mayeesha/udacity-mlnd-deeplearning-capstone/blob/master/capstone%20proposal.md
https://github.com/mvirgo/MLND-Capstone-Proposal
https://github.com/sgreenberg/seedling-classification/blob/master/proposal.pdf
https://github.com/sgreenberg/seedling-classification/blob/master/proposal.pdf
https://en.wikipedia.org/wiki/Taxonomy
https://towardsdatascience.com/building-a-logistic-regression-in-python-step-by-step-becd4d56
https://machinelearningmastery.com/tutorial-to-implement-k-nearest-neighbors-in-python-from

http://mines.humanoriented.com/classes/2010/fall/csci568/portfolio_exports/lguo/decisionTree.

https://dataaspirant.com/2017/02/06/naive-bayes-classifier-machine-learning/

https://pythonprogramming.net/linear-svc-example-scikit-learn-svm-python/
https://dataaspirant.com/2017/05/22/random-forest-algorithm-machine-learing/

http://scikitlearn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html