

Machine Learning Nano Degree

Capstone Report

Predicting Zoo's animals class

Khlood Ali

July 10, 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.

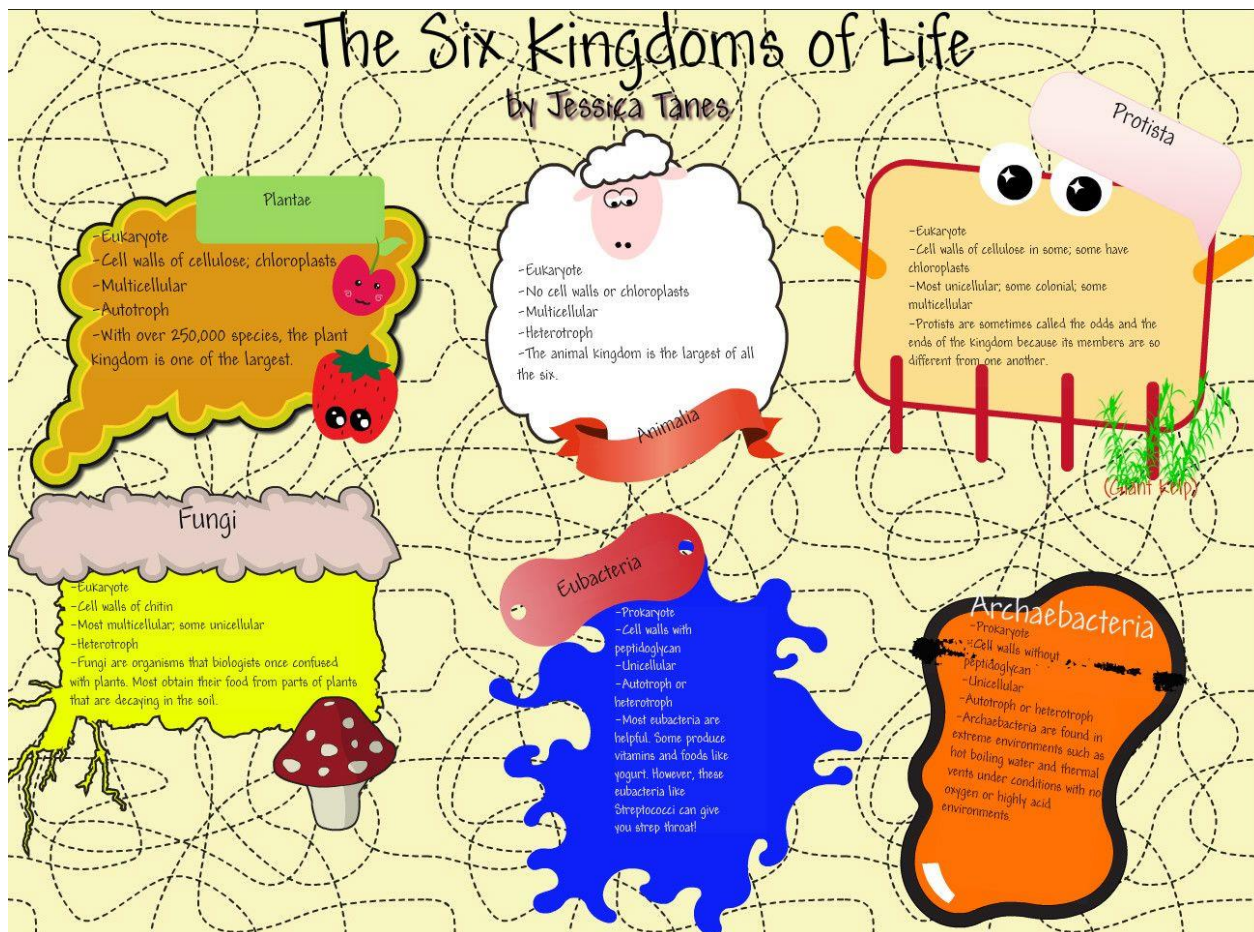
2-Plants

Subgroups: Small, Medium, Large.

Then came Linnaeus, who classified the living organisms according to its traits and called this groups "Kingdoms", and also Linnaeus has divided each 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.

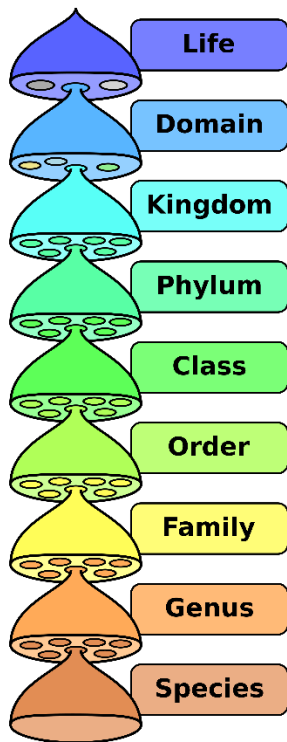
But with the use of microscope leading to the discovery of new organisms the today classification system has developed to include 5 kingdoms which are:

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



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 on the famous seven classes we’re going to classify each zoo animal in our dataset according to the classes they belong to which are :

1-Mammals

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

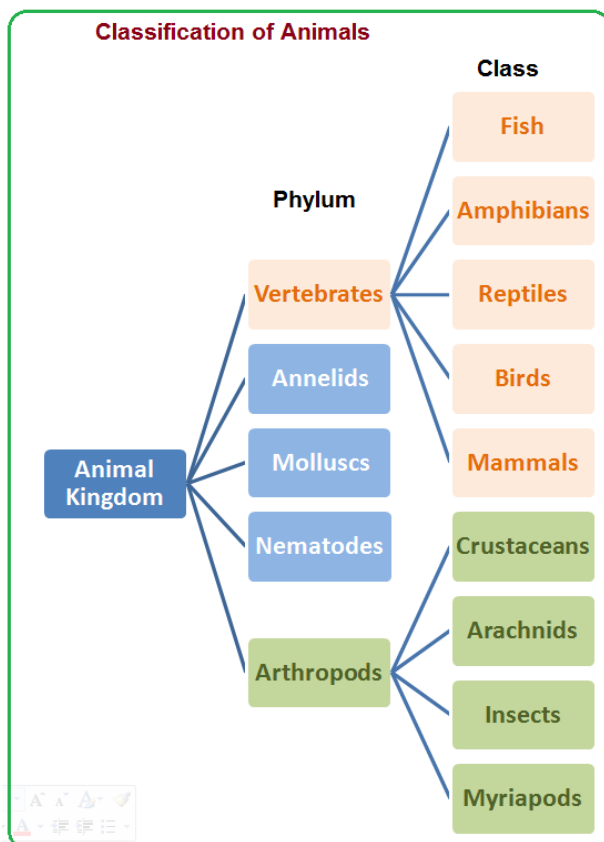
If the animal is born in water and it breaths through gills like fishes, but they develop lungs as they grow and can live on land then it belongs to the amphibian's class.

6-Arthropods "Bugs"

If the animal has more than four jointed legs and it doesn't have a backbone then 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

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 choosing 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.

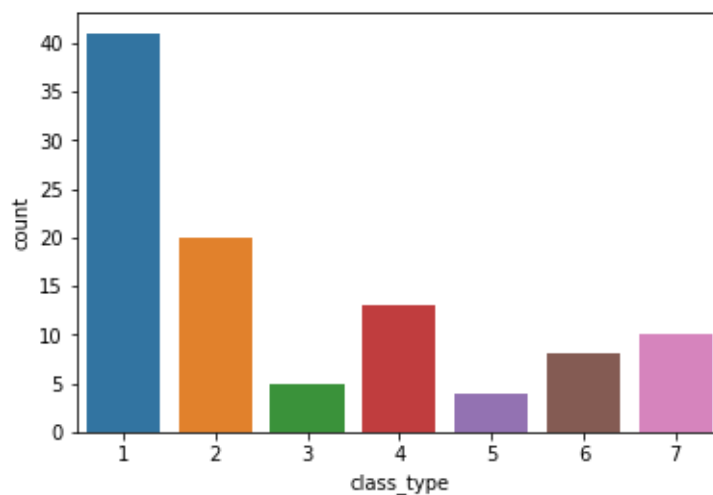
```
In [9]: x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.25,random_state=1)
        print(x_train.shape)
        print(x_test.shape)

(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()

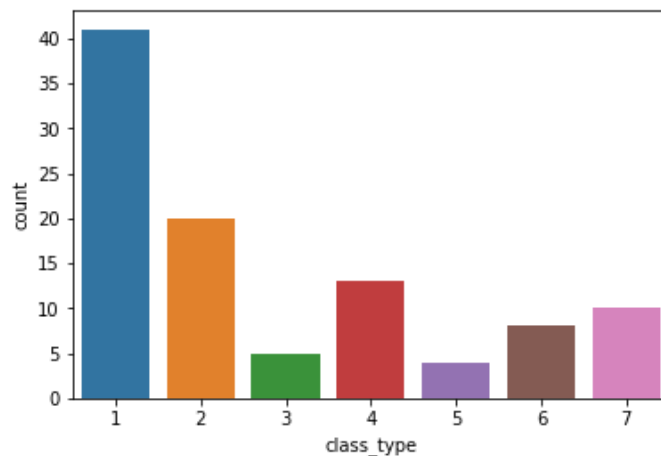
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 101 entries, 0 to 100
Data columns (total 16 columns):
hair          101 non-null int64
feathers       101 non-null int64
eggs          101 non-null int64
milk          101 non-null int64
airborne      101 non-null int64
aquatic       101 non-null int64
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()
```

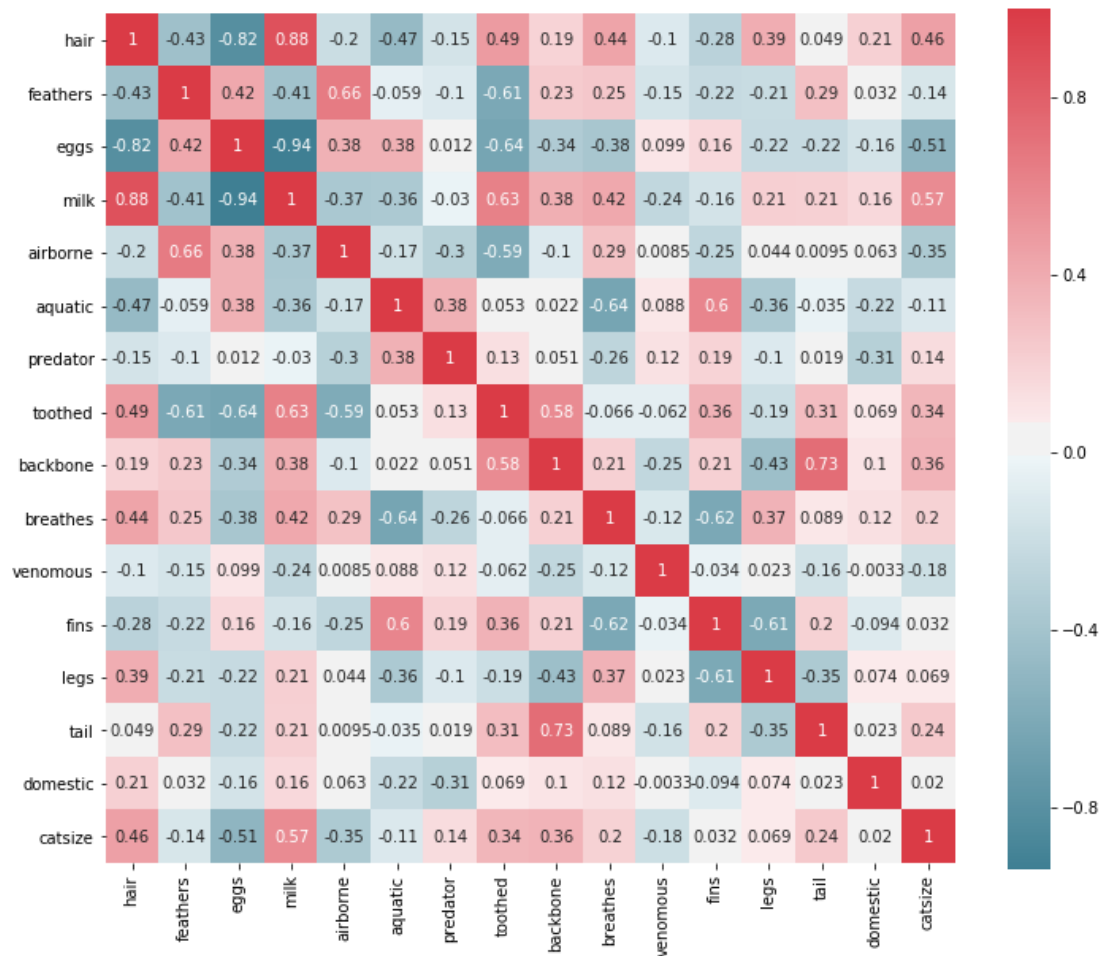
```
Out[16]: 1    41  
        2    20  
        4    13  
        7    10  
        6     8  
        3     5  
        5     4  
        Name: class_type, dtype: int64
```



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

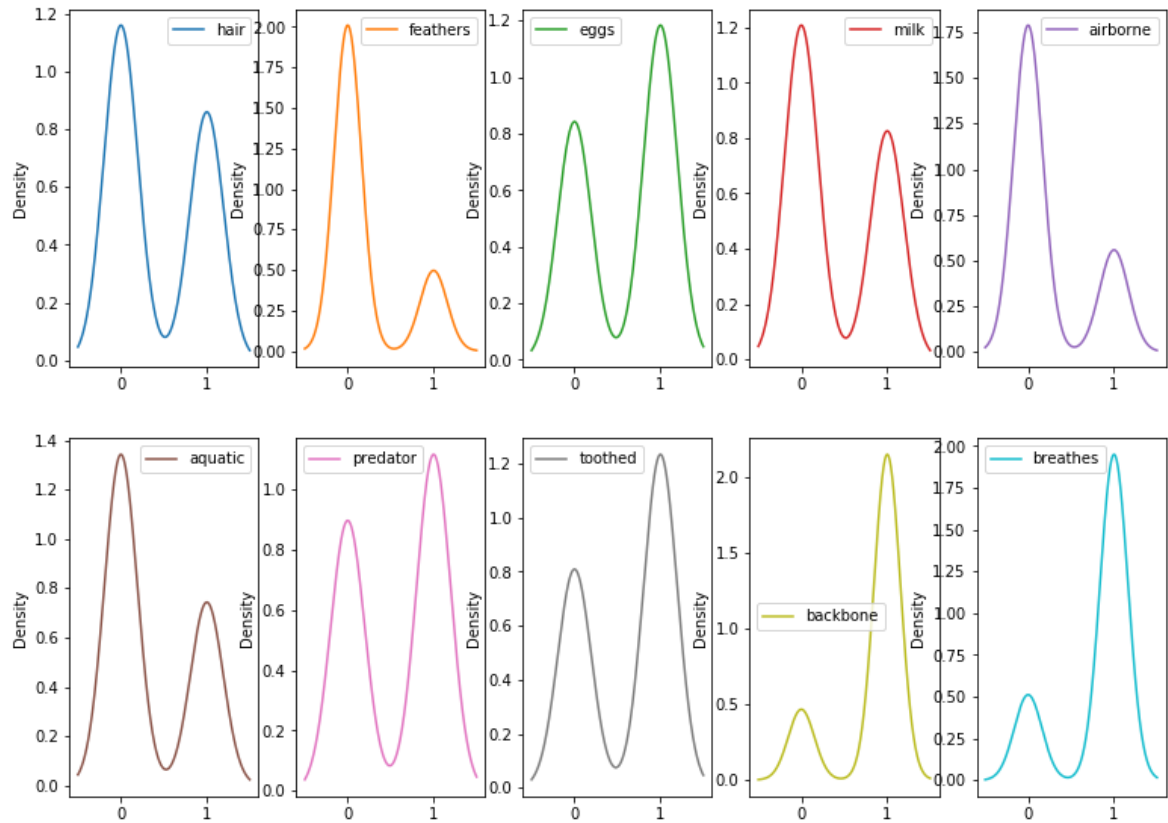
```
In [117]: hmap=x.corr()
_,ax=plt.subplots(figsize=(12,10))
cmap=sns.diverging_palette(220,10,as_cmap=True)
sns.heatmap(hmap,cmap=cmap,ax=ax,square=True,annot=True)
```

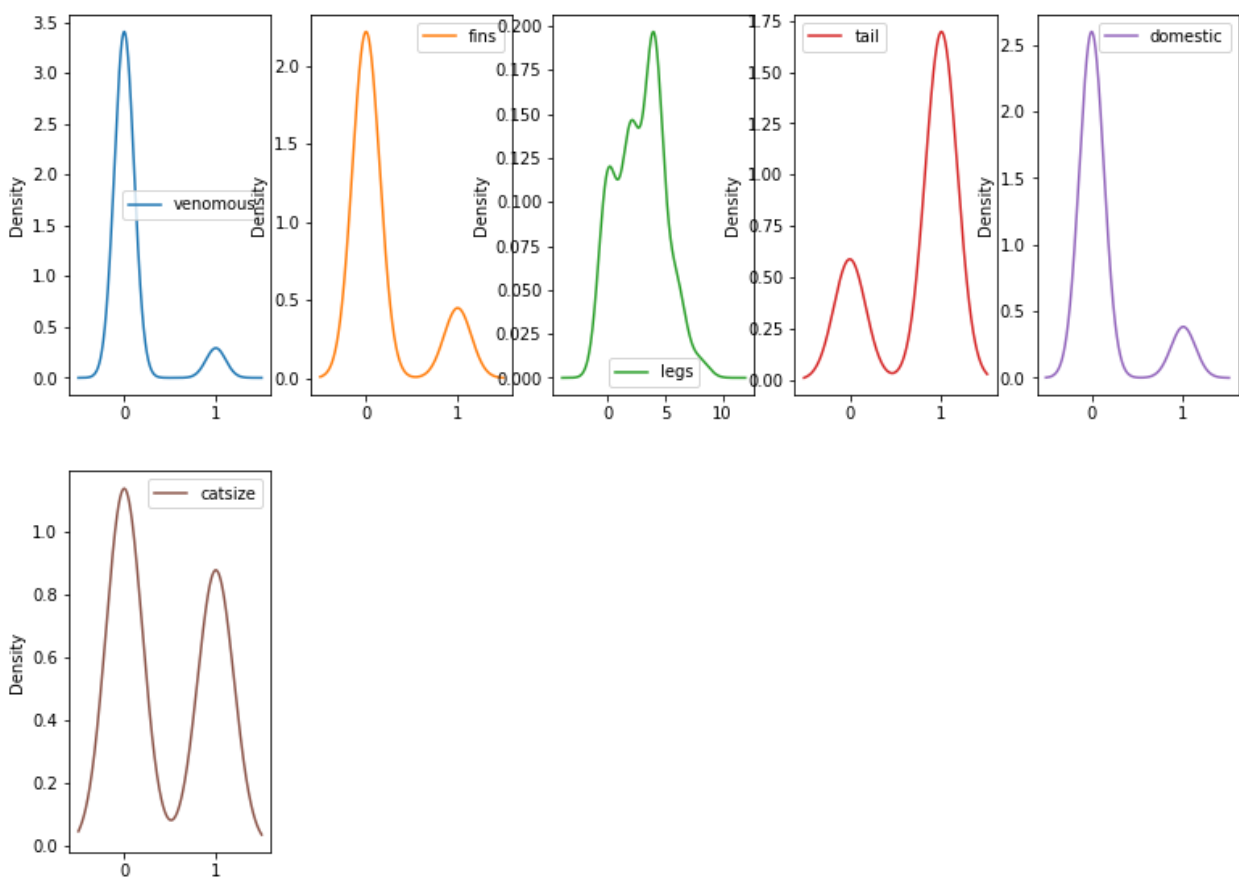
Out[117]: <matplotlib.axes._subplots.AxesSubplot at 0x1acc3ea3cc0>



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()
```





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 . It uses a curve that divides the points into classes and minimizes the errors using log loss function by assigning large penalty to the misclassified points and small to correct points.

I have used the logistic regression function from sklearn library from [here](#)

"KNN" KNeighborsClassifier()

The model of the KNN is non-parametric lazy-learner model, It uses database in which the data points are separated into several classes in order to predict the class of any new data point, The object is classified by majority vote of its neighbors data points with the object being assigned to the class with the most votes among its k neighbors, 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.

I've used KNN model from sklearn [here](#)

"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. So it uses simple true or false questions to determine the decision.

I've used the Decision Tree model from sklearn library [here](#)

"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 by combining the most rule features to detect the decision.

I've used Gaussian Naïve bayes model from sklearn library [here](#)

“SVM” SVC()

It's a technique that modifies the curve of separation that classifies the points in order to have the largest distance between the points and the curve, the support vector classifier uses kernel trick which is a way to separate classes using 3d technique by adding dimension on 2d features as an equation to separate classes.

I've used the svm classifier model from sklearn library [here](#)

“RF” RandomForestClassifier()

It's a supervised classification algorithm that creates a number of decision trees or forest of decision trees, in which each tree used to predict the output class and combine the output using bagging and the higher the number of trees given in the forest the higher the accuracy results are.

I've used random forest model from sklearn library [here](#)

“AB” AdaBoostClassifier()

Adaptive boosting or Adaboost is a Machine learning algorithm that use a bunch of weak learners and train them repeatedly and each time the total cost function (error function) decreases a bit till it reach global minimum, then it combines the output of all weak learners to form a strong valid rule, and through this process the new rule will distinguish between the individual points.

I've used adaboost model from sklearn library [here](#)

“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.

I've used GBM model from sklearn library [here](#)

I also used some techniques:

“GSCV” GridSearchCV()

It's a model hyperparameter optimization technique iterate on several values for given parameters to find the model with the highest accuracy(or another scoring function)

I've used grid search function found in sklearn library [here](#)

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% using Artificial neural network and single layer perceptron and I've reached an F1 score equals 96% using the random forest classifier.

III. Methodology

III. I Data Preprocessing

The data didn't need any preprocessing as the dataset was clean enough with clear features, none of the cells had a null value, the whole data was represented through Boolean data type and every feature had a value of 0 or 1, every row contained the animal name with its features along and the class type number in the last column, the data was actually clean and clear and just ready to be used.

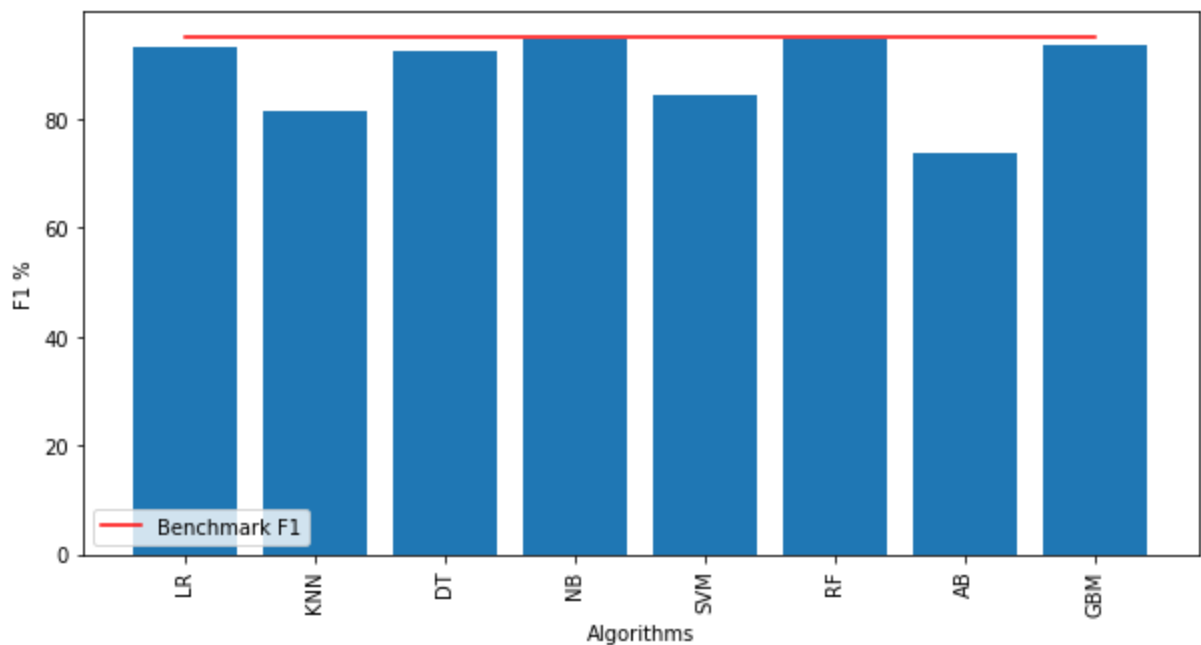
I have just splitted it into training data and testing data while each part of the data was splitted into y target feature which is the class here and X input data which is all features except the class column and the names of the animals

III. II Implementation

Programming language used: python 3

Libraries used: Pandas, sklearn, seaborn, matplotlib, numpy.

1. First of all I've uploaded the data & libs.
2. I presented the whole dataset by using the `full_data.head(10)` to show the first 10 rows of the dataset.
3. I prepared the data by showing how many rows and features it included and how many of them were in both the training and testing phases.
4. Analyzed the data to see how many animals each class had, and showed the results numerically and plotted it for more visualization.
5. I described the data to calculate their Mean, standard deviation, the 3 quartiles, the minimum and maximum value for each feature.
6. Then used the heatmap to show the correlation among the features.
7. And also used the `x.plot()` function to see the density of each feature.
8. I've prepared the variables and records list needed, by renaming each of them and used the cross validation setting it to the default number of folds = 10 as it gave the best result.
9. I then used 8 different algorithms to train the data to see which algorithm will return the highest accuracy giving the best result.
10. I trained each model using the cross-validation data to test various parameters tuning to choose the best and Kfolds technology and the highest accuracy score was for the Random Forest classifier = 94.6%.
11. Then I used the `plot.figure()` function to compare my result to the benchmark results where the RF & NB both fitted the benchmark very well.



12. Then I fine tuned the chosen models to find the best estimator of them.

13. Then I trained the Random Forest Classifier where its F1 score exceeded the initial result by about 2% giving the best accuracy = 96.1%.

I have found no complications executing this part as the data was clean and it all came to finding The best model with the highest accuracy.

III. III 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

IV. I Model Evaluation and Validation

After training the 8 models on the data I've found that the Random Forest Classifier is the most accurate as it outperformed all the other classifiers.

```
LR: 93.214%, 0.068
KNN: 81.429%, 0.083
DT: 92.143%, 0.064
NB: 94.464%, 0.068
SVM: 84.107%, 0.079
RF: 94.643%, 0.066
AB: 73.571%, 0.122
GBM: 93.393%, 0.066
```

And for the validation I've used the `sklearn's GridSearchCV` function to perform cross validation 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 that the dataset is relatively small and to get the best parameters, and the final F1 of the Random Forest Classifier score was 96.2% where the score variation from the initial result "94.6" was about 2%.

```
model f1 score using RandomForestClassifier is 0.9615384615384616
```

And from the screenshot above it's clear that there isn't much variance in scores.

IV. II 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

V. I Free-Form Visualization

This's a visual showing the weight and importance of each feature of the following:

Hair, feathers, eggs, milk,
airborne, aquatic, predator, toothed,
backbone, breaths, Venomous, fins,
legs, tail, domestic, cat size.

```
#The weight of each feature  
(model_rforest.feature_importances_)
```

```
array([9.56467157e-02, 1.30476398e-01, 1.52515340e-01, 1.12662609e-01,  
       4.11476689e-02, 3.40744903e-02, 1.56525494e-02, 3.49571556e-02,  
       6.76488168e-02, 6.11842867e-02, 1.21900709e-02, 7.48728059e-02,  
       1.10579955e-01, 5.03672993e-02, 4.71027620e-05, 5.97673488e-03])
```

V. II Reflection

At first the problem I was set to solve was a multi-class classification problem, where the problem is finding to which animalia kingdom class each zoo animal belongs, and the approach I took to solve the problem was finding the best classifying model which can classify the animals to its right classes with the highest accuracy, and the final solution I've found was the Random Forest Classifier which gave me the highest accuracy exceeding my expectations.

The really interesting aspects of this project were first the domain which reminded me how much I 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.

V. III 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-](https://github.com/udacity/machine-learning/blob/master/projects/capstone/capstone_proposal_template.md)

[learning/blob/master/projects/capstone/capstone_proposal_template.md](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-Mayeasha/udacity-mlnd-deeplearning-](https://github.com/Tahsin-Mayeasha/udacity-mlnd-deeplearning-capstone/blob/master/capstone%20proposal.md)
[capstone/blob/master/capstone%20proposal.md](https://github.com/Tahsin-Mayeasha/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-learning/>

[http://scikitlearn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.h
tml](http://scikitlearn.org/stable/modules/generated/sklearn.ensemble.AdaBoostClassifier.html)