2.1

Connectionist

I have performed the classification method of **Multilayer Perceptron**, which belongs to the Connectionist tribe of Artificial intelligence.

1.1 Results for Multilayer Perceptron (without alteration):

```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances          97               96.0396 %
Incorrectly Classified Instances         4                3.9604 %
Kappa statistic                          0.9477
Mean absolute error                      0.0199
Root mean squared error                  0.1047
Relative absolute error                  9.0563 %
Root relative squared error             31.7218 %
Total Number of Instances              101

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     mammal
                 1.000    0.012    0.952      1.000    0.976      0.970    0.998     0.992     bird
                 0.600    0.010    0.750      0.600    0.667      0.656    0.938     0.693     reptile
                 1.000    0.011    0.929      1.000    0.963      0.958    1.000     1.000     fish
                 0.750    0.000    1.000      0.750    0.857      0.862    1.000     1.000     amphibian
                 1.000    0.011    0.889      1.000    0.941      0.938    0.997     0.970     insect
                 0.900    0.000    1.000      0.900    0.947      0.944    0.966     0.924     invertebrate
Weighted Avg.    0.960    0.005    0.960      0.960    0.958      0.956    0.993     0.973

=== Confusion Matrix ===

  a  b  c  d  e  f  g   <-- classified as
 41  0  0  0  0  0  0 |  a = mammal
  0 20  0  0  0  0  0 |  b = bird
  0  1  3  1  0  0  0 |  c = reptile
  0  0  0 13  0  0  0 |  d = fish
  0  0  1  0  3  0  0 |  e = amphibian
  0  0  0  0  0  8  0 |  f = insect
  0  0  0  0  0  1  9 |  g = invertebrate
```

1.1.1 Example of Sigmoid Node

```
    Attrib fins=true     0.04604162810362551
    Attrib legs     -0.0058392453575106945
    Attrib tail=true     0.02433761962027571
    Attrib domestic=true     0.025615262801286273
    Attrib catsize=true     -0.019449302038311713
Sigmoid Node 10
    Inputs    Weights
    Threshold     0.04620689182075387
    Node 7     0.022429503357886876
    Node 8     0.01773541612498744
    Node 9     0.030439541722463576
```

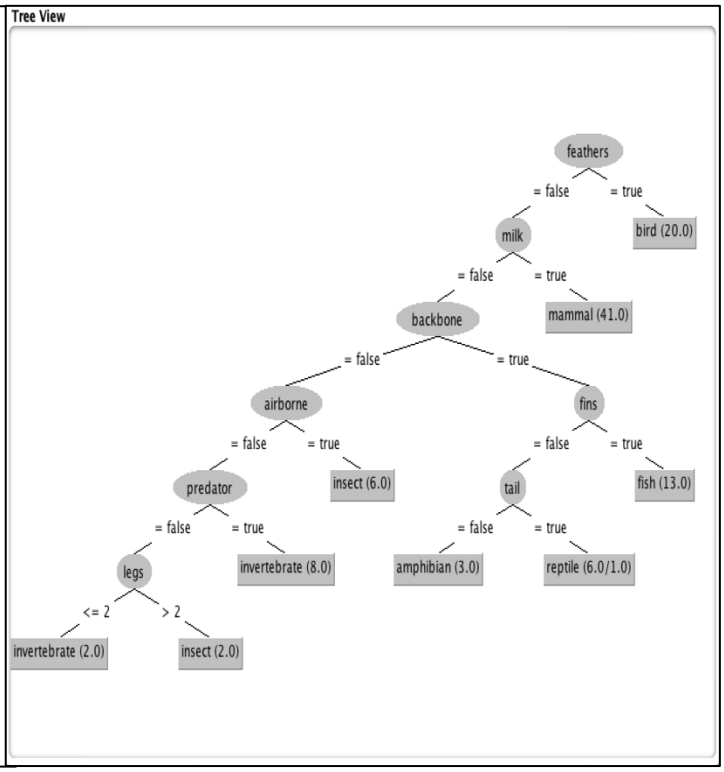
Symbolic

I have performed the classification method of **J48** , which belongs to the Symbolic tribe of Artificial intelligence.

1.2 Results for J48 (without alteration):

```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances          93               92.0792 %
Incorrectly Classified Instances         8                7.9208 %
Kappa statistic                          0.8955
Mean absolute error                      0.0225
Root mean squared error                  0.14
Relative absolute error                 10.2478 %
Root relative squared error             42.4398 %
Total Number of Instances              101

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     mammal
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     bird
                 0.600    0.010    0.750      0.600    0.667      0.656    0.793     0.420     reptile
                 1.000    0.011    0.929      1.000    0.963      0.958    0.994     0.929     fish
                 0.750    0.000    1.000      0.750    0.857      0.862    0.872     0.760     amphibian
                 0.625    0.032    0.625      0.625    0.625      0.593    0.920     0.677     insect
                 0.800    0.033    0.727      0.800    0.762      0.735    0.986     0.812     invertebrate
Weighted Avg.    0.921    0.008    0.922      0.921    0.920      0.914    0.976     0.908

=== Confusion Matrix ===

  a  b  c  d  e  f  g   <-- classified as
 41  0  0  0  0  0  0 |  a = mammal
  0 20  0  0  0  0  0 |  b = bird
  0  0  3  1  0  1  0 |  c = reptile
  0  0  0 13  0  0  0 |  d = fish
  0  0  1  0  3  0  0 |  e = amphibian
  0  0  0  0  0  5  3 |  f = insect
  0  0  0  0  0  2  8 |  g = invertebrate
```

1.2.1 Decision Tree produced by J48:



Bayesian

I have performed the classification method of **NaiveBayes**, which belongs to the Bayesian tribe of Artificial intelligence. Below are the results which I believe are the important aspects of the result.

1.3 Results for NaiveBayes (without alteration)

```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances          96               95.0495 %
Incorrectly Classified Instances         5                4.9505 %
Kappa statistic                          0.9352
Mean absolute error                      0.0153
Root mean squared error                  0.098
Relative absolute error                  6.9784 %
Root relative squared error             29.693  %
Total Number of Instances              101

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.951    0.000    1.000      0.951    0.975      0.959    1.000     1.000     mammal
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     bird
                 0.400    0.000    1.000      0.400    0.571      0.623    0.994     0.925     reptile
                 1.000    0.034    0.813      1.000    0.897      0.886    1.000     1.000     fish
                 1.000    0.021    0.667      1.000    0.800      0.808    1.000     1.000     amphibian
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     insect
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     invertebrate
Weighted Avg.    0.950    0.005    0.963      0.950    0.947      0.943    1.000     0.996

=== Confusion Matrix ===

  a  b  c  d  e  f  g   <-- classified as
 39  0  0  2  0  0  0 |  a = mammal
  0 20  0  0  0  0  0 |  b = bird
  0  0  2  1  2  0  0 |  c = reptile
  0  0  0 13  0  0  0 |  d = fish
  0  0  0  0  4  0  0 |  e = amphibian
  0  0  0  0  0  8  0 |  f = insect
  0  0  0  0  0  0 10 |  g = invertebrate
```

Analogizers

I have performed the classification method of **SMO**, which belongs to the Analogizer tribe of Artificial intelligence. Below are the results which I believe are the important aspects of the result.

1.4 Results for SMO (without alteration)

```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances          97               96.0396 %
Incorrectly Classified Instances         4                3.9604 %
Kappa statistic                          0.9477
Mean absolute error                      0.2049
Root mean squared error                  0.3023
Relative absolute error                 93.4608 %
Root relative squared error             91.6223 %
Total Number of Instances              101

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     mammal
                 1.000    0.012    0.952      1.000    0.976      0.970    0.994     0.952     bird
                 0.600    0.010    0.750      0.600    0.667      0.656    0.765     0.472     reptile
                 1.000    0.011    0.929      1.000    0.963      0.958    0.994     0.929     fish
                 0.750    0.000    1.000      0.750    0.857      0.862    0.988     0.827     amphibian
                 1.000    0.011    0.889      1.000    0.941      0.938    0.995     0.889     insect
                 0.900    0.000    1.000      0.900    0.947      0.944    0.985     0.934     invertebrate
Weighted Avg.    0.960    0.005    0.960      0.960    0.958      0.956    0.984     0.933

=== Confusion Matrix ===

  a  b  c  d  e  f  g   <-- classified as
 41  0  0  0  0  0  0 |  a = mammal
  0 20  0  0  0  0  0 |  b = bird
  0  1  3  1  0  0  0 |  c = reptile
  0  0  0 13  0  0  0 |  d = fish
  0  0  1  0  3  0  0 |  e = amphibian
  0  0  0  0  0  8  0 |  f = insect
  0  0  0  0  0  1  9 |  g = invertebrate
```

Important aspects of the results:

The correctly classified instances and incorrectly classified instances respectively are the most important results for us to consider when asking how well a technique classifies the 'zoo' data set. This is because these metrics tell us how many instances are correctly classified by the prediction model produced by the respective algorithm. We can see the respective metrics below in Table 1, which will be a useful tool when comparing the performance of each method.

Table 1.

Method	Multilayer Perceptron	SMO	NaiveBayes	J48 (C4.5)
Correctly Classified (%)	96.0396	96.0396	95.0495	92.0792
Incorrectly Classified (%)	3.9604	3.9604	4.9505	7.9208

Incorrectly classified instances are visualised in the confusion matrix. The importance of this as a result can be seen through an example; we can see that an amphibian was incorrectly classified as a reptile. This represents a false positive and is found in all the prediction models used. This information can be used in the future to perhaps train the models to remove this error. Furthermore the confusion matrix is also a good tool for summarizing accuracy when the data set is imbalanced, which in this case it is.

Because we want to classify as many animals as possible correctly for the 'zoo' data set, the recall metric (sensitivity) is a useful result. This is because the metric answers the question of what proportion of true positives are correctly classified. It is important to note that more often than not improving the recall metric will be detrimental to the precision metric. However, given the nature of the classification problem of the 'zoo' data set this is not a trade of we are concerned with. This is because improving predicted positives equating to actual positives is not entirely useful when trying to correctly classify as many true positives as possible (which we are trying to do with the 'zoo' data set).

F1 measure is useful because it takes both false negatives and false positives into account. It is the 'weighted' average of recall and precision. It will be especially useful because the zoo data is imbalanced with 40% of instances lying with the class 'mammal'.

Table 2.

Method	Multilayer Perceptron	SMO	NaiveBayes	J48 (C4.5)
Recall	0.96	0,96	0.95	0.921
F measure	0.958	0.956	0.947	0.920

The decision tree produced in 1.2.1 is a valuable way of visualizing how the J48 algorithm has predicted the class (type) of the animal. This is because it easy to understand the models prediction methods and trace errors with the models logic. I also think another useful form of presentation is the actual display of the neural network model which the Multilayer Perceptron creates. However because the data set 'zoo' has 101 instances of animals the neural network created was an undiscernible web, which gives no value to analyse. Instead we can look to the sigmoid nodes (1.1.1) to understand how the neural network has been put together and thus we can see how it predicts the types of the animals for classification and potentially where the model is making mistakes.

Why each stated technique belongs to a given tribe:

The technique **Multilayer Perceptron** involves creating artificial neurons (sigmoid nodes) and connecting them in a neural network. This aligns with the connectionist tribe as the perceptron is essentially trying to reverse engineer the brain. From a representation stand point both the perceptron and the connectionist tribe use a neural network to visualise the reimagined neural connections for the given data set. When the perceptron predicts some animal type (class) for output, it compares this with the actual animal type and will send the error back through the nodes and will adjust the weights accordingly. This process is common among connectionist algorithms and is known as back propagation. It optimizes the lest errors by using gradient decent. This calculates the gradient of the error function with regard to the networks weights and will adjust the weight in the direction of less error. Furthermore, the perceptron evaluates the models errors for classification using mean squared error, which aligns with the connectionist tribe evaluation methods.

J48 belongs to the Symbolist tribe firstly because they both use logic as the basis for representation. The tree structure which is created using J48 provides a series of paths which give you binary choice as to which path you will take. Using logic each choice will diminish the given data's possible outcomes and consequentially push the data to the correct classification. The evaluation of how well the model's logic works depends on its accuracy and is another reason why the tool J48 belongs to the symbolist tribe. This is because if the accuracy of classification is low then the choices are not likely to be that logical or useful, consequentially evaluating to bad performance. J48 also uses the symbolist tribe's optimization driver of inverse deduction to optimize the accuracy of classification.

The technique **NaiveBayes belongs to the Bayesian tribe** because they use probabilistic inference to optimize the performance of the algorithm. NaiveBayes uses Bayes theorem to infer classification by updating the probability of a hypothesis for classification when more data becomes available. The probabilistic models performance is evaluated against posterior probability, which is the probability that a prediction for classification is true when it has regard to relevant observations. This is the primary evaluating metric for the symbolist tribe and therefore is another reason why NaiveBayes belongs to them. NaiveBayes similarly to other Bayesian tribe techniques is represented through a graphical model, NaiveBayes uses a tree with a single root. This expresses conditional dependency between certain attributes and will help visualise the optimized paths to each classification.

The technique **SMO belongs to the Analogizer tribe** because it is represented by a support vector. The technique SMO is evaluated by looking at how large the margin is between two classes in the support vector representation. This is done by finding the hyperplane which maximizes the distances between classes. This type of evaluation which the SMO technique is using is considered the standard for the Analogizer tribe. SMO like any other technique in the Analogizer tribe uses constrained optimization to acquire the best results for the given data set.

Important differences which contribute to performance:

All the important aspects (representation, evaluation method and optimization driver) of each technique differ. This is because they each belong to different tribes of AI. However there are some differences which are more important to draw attention to than others such as why the particular techniques were conceived. The NaiveBayes technique was created to implement Bayes theorem (a probabilistic model) to help predict outcomes in machine learning, whereas the MultilayerPerceptron was conceived in order to try and recreate the brain and in doing so predict and classify instances. J48 was conceived just so that a simple decision tree algorithm could be applied in machine learning and . Furthermore, by having regard to why the techniques were created we can see the far reaching and diverse field of machine learning and that there are vastly different routes you can take for the same objective of classification. So ultimately we must find which technique fits our data set the best as all of them have the same objective.

This raises the question of which technique is suited to the given 'zoo' data set. I think that NaiveBayes is suited to the given data set as the classification problem seems to be one which would find the Bayes theorem useful. This is because the Bayes theorem allows someone to correctly classify an instance such a mammal with regard to other factors which will point you in that direction e.g. does the animal produces milk. I also think that the MultilayerPerceptron is suited to this data as it can give greater weight to relationships of attributes and the class they are likely to correspond to. For example the classification of a mammal is not dependent on the length of their legs and therefore the technique would give little weight to it. However the relationship between a mammal and the production of milk is incontrovertible and the technique will therefore give large weight to it, which consequentially allow for greater classification of the given data set 'zoo'.

Another important difference which may not have been articulated previously is the speed at which the algorithms performs. Given the nature of my data set and how little instances I have, it is important to note which techniques may be processing slower than others. This is because if we were to acquire more data we want the results to return in a practical time. With the current data set all techniques except MultilayerPerceptron gave results instantaneously. This allows us to deduce that the neural networks requires a lot more machine processing than the other techniques. This may be something you want to consider when dealing with larger data sets and depending on what goal you have you may weigh whether you value accuracy (MultilayerPerceptron) over speed more (other techniques).

2.2

Pipeline created:

I will create a pipeline which applies some feature tuning by normalizing the legs attribute so that the data is rescaled to between 0 and 1. The pipeline will also apply instance selection and partitioning which will help deal with the class imbalance by 'minority class oversampling'. Furthermore I will implement a pipeline which applies data cleansing before the algorithms are run (adding missing values or removing duplicated instances). Lastly I will implement a pipeline which uses 10 k-fold cross validation to try and remove the threat of overfitting, which we would likely see occurring if we applied any new data to algorithms produced above in 1.1/1.2/1.3/1.4. I chose to use this instead of the 'hold-out' technique as cross validation allows the model to train on multiple test sets. I will also apply some dimensionality reduction (feature selection) as we have some redundant attributes such as the 'name' of the animal which gives no value to the relevant method when trying to create a prediction model.

Business understanding:

One of the potential reasons why the 'zoo' data set was collected is for any person who does not know the type of an animal. They could work out what type the animal is by looking at its attributes e.g. does it produce milk? Perhaps it was gathered for a zoo when they discover new animals and want to be able to classify the animals type. Zoos could look at the attributes of any type of animal to work out what type the newly discovered animal is likely to be. Above are some useful and practical reasons for gathering the data, however the data may have just been collected simply because of curiosity.

Data understanding:

The zoo data set is characterised as multivariate. This means that it has many variables for us to consider, the variables are portrayed in the form of attributes. The data set has 17 of these attributes, which are categorical by nature and hold integer values. Interestingly the data set has two of the same instance (animal) "frog" which is considered 'inconsistent data'. This is likely the consequence of there being many species of the same animal and is therefore something we have to regard to understand the nature of the data set 'zoo'. Apart from the inconsistent data the data set is very clean and contains no obvious outliers, missing data values or mixed attribute types (all categorical). The data was donated in 1990-05-15 by Richard S. Forsyth, which has since had 272,397 web hits. Another thing to note about the data which may cause it to struggle with some algorithms is that it is heavily imbalanced, there are 7 classes and one of them has 41% of the instances. There is also an instance of "girl" which is quite unusual.

Data preparation:

A pipeline can assist preparation of the data prior to the technique being applied by removing any problems the data set may present to the technique. A pipeline can help 'data cleanse' by removing invalid values from the raw data. In the case at hand we would want our pipeline to remove the duplicate instance described above. A pipeline is also useful for 'instance selection and partitioning' where we can diminish the current classification problem which is imposed by an imbalanced data set by minority class oversampling. Furthermore a pipeline can select data points to create test, validation and training sets, which are useful tools when trying to improve the algorithm.

Modelling and Evaluation:

I would say this pipeline will likely suit and improve the MultilayerPerceptron technique which belongs to the Connectionist tribe. This is because the categorical data which has been rebalanced via 'minority class over sampling' will allow the algorithm to process and manipulate data more logically and thus form a more accurate neural network. Furthermore gradient descent as a method to optimize the classification performance will also flourish because the preprocessed data will be able to give better indicators as to what the weights for each node should be. Therefore the pipeline preprocessing should only support and help the MultilayerPerceptron method more. I also think that the NaiveBayes technique which belongs to the Bayesian tribe will benefit from the rebalanced data set. This is because probability based algorithms will be able to make better predictions when they have evenly spread data as inputs. Furthermore, NaiveBayes method will be supported by the pipeline passing it preprocessed data which consequentially will make it easier for Bayes theorem to work its magic.

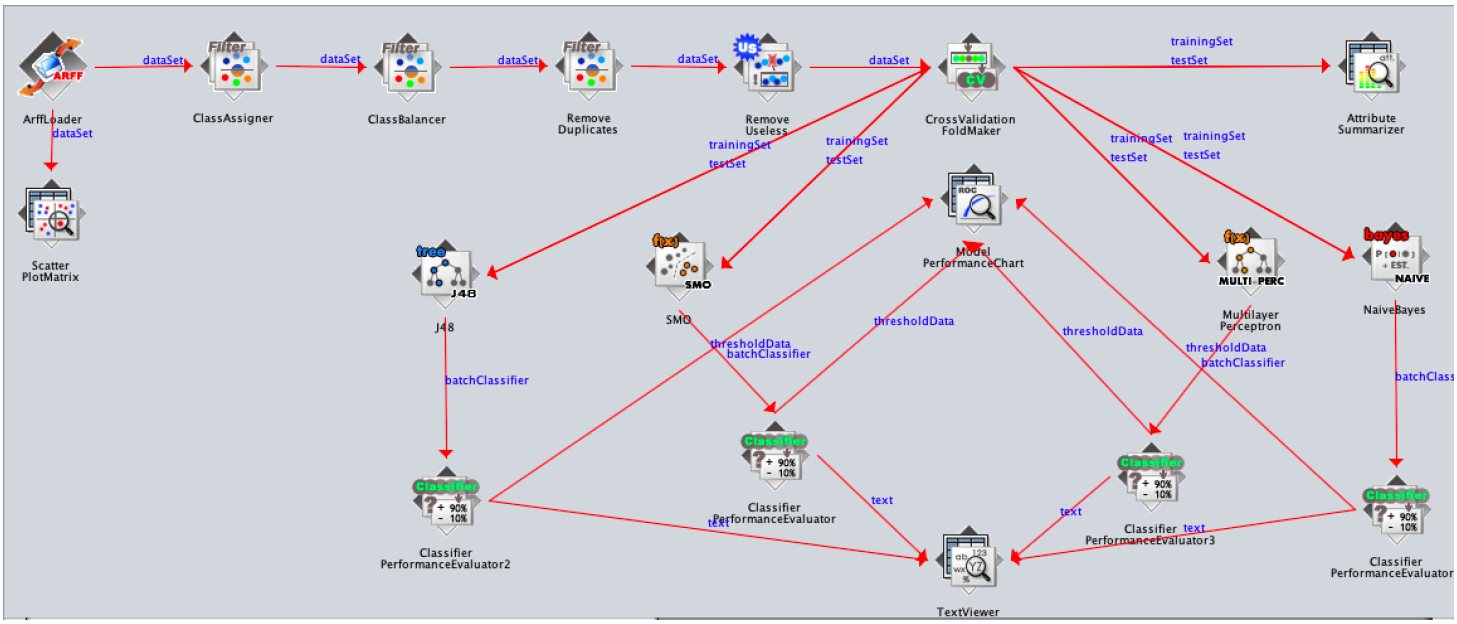
Deployment:

The model described will likely require additional effort to deploy relative to the models produced in 1.1/1.2/1.3/1.4. This is because I will be subject to human error, which will arise when I play a more involved role in creating the machine learning model. I predict that when I attempt the deploy my models I will encounter errors which have arisen from simple mistakes. Furthermore you cannot easily save the model to a local folder when using knowledge flow, this is unlike the explorer. This in turn would make it harder to deploy as you have to create your pipeline in knowledge flow and then you will have to apply all the filters.. etc. in the explorer to save a model.

2.3

Program code for classifiers

2.1 Visualization of the pipeline using knowledgeFlow



Snapshot of learned classifiers and other results

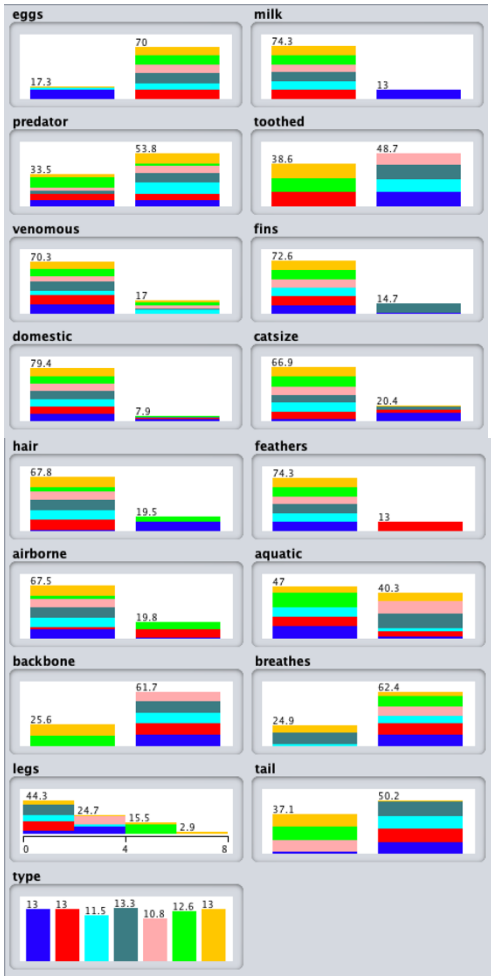
2.2 MultilayerPerceptron results (after pipeline approach)

```
Correctly Classified Instances          90.1786          89.2857 %
Incorrectly Classified Instances        10.8214          10.7143 %
Kappa statistic                          0.875
Mean absolute error                      0.0388
Root mean squared error                  0.1528
Relative absolute error                 15.7498 %
Root relative squared error             43.3602 %
Total Number of Instances              101

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     mammal
                 1.000    0.033    0.833      1.000    0.909      0.898    0.997     0.983     bird
                 0.600    0.042    0.706      0.600    0.649      0.598    0.905     0.767     reptile
                 1.000    0.033    0.833      1.000    0.909      0.898    1.000     1.000     fish
                 0.750    0.000    1.000      0.750    0.857      0.849    1.000     1.000     amphibian
                 1.000    0.017    0.909      1.000    0.952      0.945    1.000     1.000     insect
                 0.900    0.000    1.000      0.900    0.947      0.941    0.943     0.923     invertebrate
Weighted Avg.    0.893    0.018    0.897      0.893    0.889      0.875    0.978     0.953

=== Confusion Matrix ===

  a     b     c     d     e     f     g      <-- classified as
 14.43  0     0     0     0     0     0    |     a = mammal
  0    14.43  0     0     0     0     0    |     b = bird
  0     2.89  8.66  2.89  0     0     0    |     c = reptile
  0     0     0    14.43  0     0     0    |     d = fish
  0     0     3.61  0    10.82  0     0    |     e = amphibian
  0     0     0     0     0    14.43  0    |     f = insect
  0     0     0     0     0     1.44 12.99 |     g = invertebrate
```

2.3 J48 (decision tree) results (after preprocessing)

```
Correctly Classified Instances          83.325           82.5    %
Incorrectly Classified Instances        17.675           17.5    %
Kappa statistic                          0.7958
Mean absolute error                      0.0487
Root mean squared error                  0.2139
Relative absolute error                 19.7381 %
Root relative squared error             60.7058 %
Total Number of Instances              101

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     mammal
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     bird
                 0.600    0.042    0.706      0.600    0.649      0.598    0.779     0.481     reptile
                 1.000    0.033    0.833      1.000    0.909      0.898    0.983     0.833     fish
                 0.750    0.000    1.000      0.750    0.857      0.849    0.875     0.786     amphibian
                 0.625    0.067    0.610      0.625    0.617      0.553    0.900     0.640     insect
                 0.800    0.063    0.681      0.800    0.736      0.691    0.973     0.776     invertebrate
Weighted Avg.    0.825    0.029    0.833      0.825    0.824      0.798    0.930     0.788

=== Confusion Matrix ===

  a     b     c     d     e     f     g      <-- classified as
 14.43  0     0     0     0     0     0    |     a = mammal
  0    14.43  0     0     0     0     0    |     b = bird
  0     0     8.66  2.89  0     2.89  0    |     c = reptile
  0     0     0    14.43  0     0     0    |     d = fish
  0     0     3.61  0    10.82  0     0    |     e = amphibian
  0     0     0     0     0     9.02  5.41 |     f = insect
  0     0     0     0     0     2.89 11.54 |     g = invertebrate
```

2.4 Attribute summarizer (pipeline approach)



2.5 SMO results (after pipeline approach)

```
Correctly Classified Instances          85.4893          84.6429 %
Incorrectly Classified Instances        15.5107          15.3571 %
Kappa statistic                          0.8208
Mean absolute error                      0.2068
Root mean squared error                  0.3051
Relative absolute error                 83.8797 %
Root relative squared error             86.5854 %
Total Number of Instances              101

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     mammal
                 1.000    0.033    0.833      1.000    0.909      0.898    0.983     0.833     bird
                 0.600    0.042    0.706      0.600    0.649      0.598    0.805     0.501     reptile
                 1.000    0.033    0.833      1.000    0.909      0.898    0.983     0.833     fish
                 0.750    0.000    1.000      0.750    0.857      0.849    0.961     0.836     amphibian
                 0.875    0.050    0.745      0.875    0.805      0.772    0.964     0.711     insect
                 0.700    0.021    0.848      0.700    0.767      0.737    0.947     0.719     invertebrate
Weighted Avg.    0.846    0.026    0.852      0.846    0.842      0.822    0.949     0.776

=== Confusion Matrix ===

  a     b     c     d     e     f     g      <-- classified as
 14.43  0     0     0     0     0     0    |     a = mammal
  0    14.43  0     0     0     0     0    |     b = bird
  0     2.89  8.66  2.89  0     0     0    |     c = reptile
  0     0     0    14.43  0     0     0    |     d = fish
  0     0     3.61  0    10.82  0     0    |     e = amphibian
  0     0     0     0     0    12.63  1.8  |     f = insect
  0     0     0     0     0     4.33 10.1  |     g = invertebrate
```

2.6 NaiveBayes results (after pipeline approach)

```
Correctly Classified Instances          91.639           90.7317 %
Incorrectly Classified Instances         9.361            9.2683 %
Kappa statistic                          0.8919
Mean absolute error                      0.0249
Root mean squared error                  0.1236
Relative absolute error                 10.105  %
Root relative squared error             35.08   %
Total Number of Instances              101

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.951    0.000    1.000      0.951    0.975      0.971    0.999     0.996     mammal
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     bird
                 0.400    0.000    1.000      0.400    0.571      0.603    0.988     0.948     reptile
                 1.000    0.041    0.801      1.000    0.889      0.876    1.000     1.000     fish
                 1.000    0.033    0.833      1.000    0.909      0.898    1.000     1.000     amphibian
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     insect
                 1.000    0.033    0.833      1.000    0.909      0.898    1.000     1.000     invertebrate
Weighted Avg.    0.907    0.015    0.924      0.907    0.893      0.892    0.998     0.992

=== Confusion Matrix ===

  a     b     c     d     e     f     g      <-- classified as
 13.72  0     0     0.7   0     0     0    |     a = mammal
  0    14.43  0     0     0     0     0    |     b = bird
  0     0     5.77  2.89  2.89  0     2.89 |     c = reptile
  0     0     0    14.43  0     0     0    |     d = fish
  0     0     0     0    14.43  0     0    |     e = amphibian
  0     0     0     0     0    14.43  0    |     f = insect
  0     0     0     0     0     0    14.43 |     g = invertebrate
```

Comparison of results between the tools

Table 3. Accuracy in terms of correctly classifying instances

Method	Multilayer Perceptron	SMO	NaiveBayes	J48 (C4.5)
Correctly Classified (%)	89.2857	84.6429	90.7317	82.5
Incorrectly Classified (%)	10.7143	15.3571	9.2683	17.5

From these classification results we can deduce that NaiveBayes and MultilayerPerceptron are the best models to help solve this particular data sets classification problem. They are better at classifying a balanced multiclass data set than J48 and SMO. We saw the largest decrease in correct classification in the classification methods SMO and J48. This is because the models could not process and manipulate the rebalanced ‘zoo’ data set as well as the other two. Consequentially they struggled to find patterns, which allowed them to classify a wide range of classes as opposed to just the mammal class, which was heavily favored during the naïve approach.

Comparison of results between 'naïve approach' and 'pipeline approach'

Firstly we can see that there is an overall decrease in the correctly classified instances by comparing the results in 2.2/2.3/2.5/2.6 with 1.1/1.2/1.3/1.4. This is likely the consequence of ridding ourselves of overfitting, and replacing it with real pattern recognition models which will not struggle on new data when it tries to predict its class. Furthermore, the decrease in accuracy is also the likely result of minority class oversampling. During the naïve approach the majority of instances were mammals and constituted 41% of the entire dataset. These high classification results misrepresented how well the model could actually predict new classes which had fewer instances, such as amphibians. However, now (after the pipeline approach) that we have balanced the instances of classes, the models have been able to holistically search for patterns which help solve our classification problem instead of only solving one classes (mammals). We can see that the class has been balanced through result 2.4. Also it is also important to note that all instances of the mammal class held true for the attribute milk (no other instance did). Because of this and the imbalance in the naïve approach, it made it extremely easy for the model to predict a mammal, which made up the majority of the data set. It is therefore a likely reason why the models classified so many correctly during the naïve approach in comparison to the pipeline approach.

I have confidence that the results given using the pipeline approach will be able to replicated on a new set of data. I do not think that the naïve approach models would be able to give the same great classification accuracy on a new set of data. This is because the prediction models created are overfitted and only predict the classes well with that specific data set. The models created using the pipeline approach have general pattern recognition for all classes in the multiclass set and resultingly I have confidence in the models to succeed with new data.